Unveiling Privacy Risks in Quantum Optimization Services

Mateusz Leśniak
mateusz.lesniak@nask.pl
NASK National Research Institute
Warsaw, Poland

Michał Wroński
michal.wronski@nask.pl
NASK National Research Institute
Warsaw, Poland

Ewa Syta
ewa.syta@trincoll.edu
Trinity College
Hartford, CT, USA

Mirosław Kutyłowski
miroslaw.kutylowski@nask.pl
NASK National Research Institute
Warsaw, Poland

Abstract

As cloud-based quantum computing services, such as those offered by D-Wave, become more popular for practical applications, privacy-preserving methods (such as obfuscation) are essential to address data security, privacy, and legal compliance concerns. Several efficient obfuscation methods have been proposed, which do not increase the time complexity of solving the obfuscated problem, for quantum optimization problems. These include *sign reversing*, *variable permutation*, and the combination of both methods assumed to provide greater protection. Unfortunately, sign reversing has already been shown to be insecure.

We present two attacks on variable permutation and the combined method, where it is possible to efficiently recover the deobfuscated problem, particularly when given access to the obfuscated problem and its obfuscated solution, as a cloud-based quantum provider would have. Our attacks are in the context of an optimization problem of cryptanalysis of the Trivium cipher family, but our approach generalizes to other similarly structured problems.

Our attacks are efficient and practical. Deobfuscating an optimization problem with n variables obfuscated with the combined method has a complexity of $O(n^2)$ compared to the complexity of $O\left(n \cdot n! \cdot 2^n\right)$ of the brute force attack. We provide an implementation of our attack; using a commodity laptop, our attack using the full Trivium cipher takes less than two minutes if optimized. We also present possible countermeasures to mitigate our attacks and bring attention to the need for further development in this area.

CCS Concepts

• **Security and privacy → Cryptanalysis and other attacks**.

Keywords

Privacy preserving techniques, attacks, obfuscation, quantum computing and annealing, quantum optimization

ACM ASIACCS 2025, August 25–29 , 2025, Hanoi, Vietnam

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Mateusz Leśniak, Michał Wroński, Ewa Syta, and Mirosław Kutyłowski. 2025. Unveiling Privacy Risks in Quantum Optimization Services. In *Proceedings of The 20th ACM ASIA Conference on Computer and Communications Security (ACM ASIACCS 2025).* ACM, New York, NY, USA, 17 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The ever increasing scale and complexity of computational problems require significantly greater computational power [48], driving the search for alternative solutions like those offered by quantum computing. While classical (von Neumann) solutions have been widely and effectively employed, quantum solutions present an appealing alternative, offering far greater computational capabilities, particularly for specific classes of problems such as cryptography, complex simulations, machine learning, and more.

Gate-based quantum computers are the most widely recognized type of quantum computer. Operating on the principles of quantum gates to manipulate quantum bits (qubits), they are considered general purpose as they can solve a wide range of problems and run any quantum algorithm. Although these computers hold long-term promise as universal solutions for quantum computation, they are still experimental and face significant research and deployment challenges, particularly in scalability (increasing qubit counts) and accuracy (reducing error rates) [32].

In contrast, quantum annealing computers are specialized quantum computers designed for specific types of computational tasks. They leverage the principles of quantum physics to identify the minimum energy state of a system, making them particularly effective for problems such as optimization and probabilistic sampling. Quantum annealing computers have scaled up to thousands of qubits, with D-Wave's Advantage system featuring over 5,000 qubits [45]. In contrast, state-of-the-art gate-based quantum computers have reached over 1,000 qubits with IBM's 1,121-qubit quantum chip [23].

Quantum annealing is increasingly being applied in real-world scenarios in diverse industries [56], in part due to the fact that access to quantum annealing is commercially available [47]. In transportation, scheduling and logistics, it has been used to optimize vehicle routing, fleet management and driver scheduling [4, 7, 20, 54], including applications such as navigating bus fleets at events [35, 55] and automating weekly driver schedules. In finance, quantum annealing has shown significant potential in portfolio optimization by effectively balancing risk and return, achieving near-optimal solutions for indices like the Nikkei225 and S&P500 [22, 34, 43, 50].

It also holds promise in cryptanalysis, particularly in the context of stream ciphers [28, 52], where it can outperform traditional methods such as brute force or the Grover's algorithm [3], although its application to asymmetric cryptography remains limited [17, 25, 31, 51].

Despite the high computational potential and applicability of quantum annealing to industry problems, building quantum computers is prohibitively expensive and challenging [32]. As a result, organizations are turning to cloud-based quantum computing services, which provide affordable and scalable access to quantum resources without the need for significant infrastructure investments. In fact, many organizations, including organizations such as Johnson & Johnson (healthcare and consumer goods), NTT Docomo Inc. (telecommunications), Recruit Group (HR and staffing solutions), Ferrovie dello Stato Italiane - FS Italiane (transportation), SavantX in cooperation with Fenix Marine Services (AI and logistics), have used D-Wave services [47].

However, this approach presents challenges associated with sharing computational problems with a third-party provider, mirroring concerns commonly seen in traditional cloud computing, such as data security, privacy, and regulatory compliance [41]. Even if we optimistically–and unrealistically, contrary to established security principles–assume a fully trusted provider resistant to all security breaches, regulatory compliance issues still persist. Unlike traditional cloud computing, where local providers can often meet jurisdictional requirements, cloud-based quantum annealing services will, for the foreseeable future, require reliance on organizations operating outside the same legal jurisdiction. This is due to the limited availability and regional presence of those services.

These legal compliance challenges affect companies and service providers alike. Companies are required to comply with relevant laws, such as data protection and privacy regulations, including EU GDPR [38], California CCPA [27], and Brazilian LGPD [8]. At the same time, service providers may prefer not to access sensitive or protected data to mitigate compliance risks. This becomes particularly complex in international applications, where differing legal frameworks add another layer of difficulty. For instance, national security and surveillance laws, such as the U.S. CLOUD Act [15], China's Cybersecurity Law [9], and the U.S. Patriot Act [12], can impose obligations on providers to disclose customer data, sometimes conflicting with local laws in other jurisdictions. Additionally, cybersecurity laws like the EU Cybersecurity Act [39], U.S. HIPAA [11], and U.S. FISMA [13] impose strict requirements for securing sensitive data, while intellectual property and industry-specific regulations, such as the U.S. Dodd-Frank Act [14] for financial services and FERPA for educational data [10], add further compliance obligations. See Section 2 for further discussion.

A straightforward solution to mitigate these issues, often employed in traditional cloud computing [5, 21, 42], is to apply privacy-preserving techniques to obfuscate input and output data, a strategy also applicable in the context of cloud-based quantum computation.

In this paper, we focus on privacy-preserving techniques for quantum optimization problems that can be solved using either quantum annealing [26, 33] or gate-based quantum methods such as the Quantum Approximate Optimization Algorithm (QAOA) [2, 18, 19]. However, we are primarily concerned with the impact of the attacks we present on quantum annealing, as it is currently the most practical quantum computing approach for optimization problems and is likely to remain so for the foreseeable future.

Quantum optimization problems are typically represented in the form of Quadratic Unconstrained Binary Optimization (QUBO) or the Ising model, and several obfuscation techniques for those problems exist (see Table 1). A task to be solved is first transformed into an optimization problem (either QUBO or Ising model), after which a selected obfuscation technique is applied. The obfuscated optimization problem is then sent to the cloud-based quantum provider to solve. Once the provider returns the obfuscated solution, it is deobfuscated to yield the solution to the original task (see Figure 1).

One such obfuscation technique is *sign reversing* [37], which involves reversing the sign of selected coefficients in the Ising model representation of the optimization problem to disguise its structure. Another method is *variable permutation*, partially derived from [1], where both a vector of variables and the QUBO matrix are multiplied by a permutation matrix. These two methods can also be combined to improve the level of obfuscation. Unfortunately, sign reversal (spin reversal transformation [37] and Enigma-I [1]) has been shown to be insecure [29]. It is possible to efficiently deobfuscate the obfuscated optimization problem and its solution by arranging and solving a system of equations.

In this paper, we show that there exist specific problems for which these privacy-preserving techniques are insecure, even when both are applied simultaneously. Our results apply to variable permutation, a component of Enigma-II [1], as well as to the simplified Enigma-II [1] without the inclusion of additional variables (see Table 1), effectively rendering all currently available obfuscation methods that *do not* increase the complexity of the obfuscated optimization problem ineffective. This is a serious concern for practical applications of cloud-based quantum computation.

We present two attacks (see Section 5) in the context of the Trivium cipher family [49], and show that the obfuscation methods mentioned above are insecure. In our attack, we use the cryptanalysis of a cipher as an optimization task to be solved using a cloud-based quantum provider. The task (an algebraic attack against the Trivium cipher [52] with the standard 80-bit key length) is converted to a QUBO optimization problem, obfuscated using the respective methods, and then sent to be solved. We then show how an attacker (who receives the obfuscated optimization problem and/or its obfuscated solution) can extract details about both the optimization problem and its solution. We stress that our attack is against the obfuscation methods and *not* the Trivium cipher family.

The Trivium cipher family, like many hardware-oriented ciphers, is highly structured, making it particularly susceptible to our attacks. Our attack could extend to other classes of optimization problems which are similarly structured. Since we cannot definitively determine whether a given problem is highly structured or not, we should assume that all problems are potentially vulnerable.

The only two other obfuscation methods currently available – the full Enigma-III algorithm [1] and the combination of digit-wise splitting with matrix permutation [53]–increase the complexity of the obfuscated problem. Enigma-III requires adding additional variables and connections, which, at a time of limited resources, may render the available hardware unusable. Even with the planned 7000-qubit annealer [46], a solution that creates a fully connected graph is infeasible. The second method involves solving multiple

optimization problems instead of one, increasing computational costs and time requirements, as each problem requires additional processing. More importantly, however, these methods have been proposed relatively recently and, although no known attacks have been identified so far, they also lack formal proofs of security.

While we provide countermeasures (see Section 7), our work highlights the need to develop more robust obfuscation techniques for quantum optimization problems. These techniques should not only provide security, but also maintain practicality by minimizing computational overhead and preserving compatibility with existing quantum solvers.

In summary, our contributions are as follows.

- We show that common quantum optimization privacy-preserving techniques are insecure if the adversary can access the obfuscated problem represented in any form, as is the case of a cloud-based quantum solver. To the best of our knowledge, we present the first attack against the variable permutation and the combination of sign reversing and variable permutation methods by exploiting the optimization problem's structure. For the combined method, our attack works regardless of the order in which the individual methods are applied. Our attack has a polynomial complexity of $O(n^2)$, which is significantly faster than the superexponential time complexity of the brute force attack of $O\left(n \cdot n! \cdot 2^n\right)$ [1].
- We provide an implementation of our attack in SageMath [44] and evaluate it using Trivium with an 80-bit key. We demonstrate that our attack is highly efficient, taking under 60 minutes when unoptimized and under 2 minutes when optimized. Additionally, we present a pen-and-paper example of our attack using the Univium cipher with a 9-bit key.
- We propose mitigation techniques such as modifying the coefficient in the Rosenberg substitution, forming linear combinations of Boolean equations, and applying affine transformations to Boolean variables before transforming the task into an optimization problem. These techniques can be combined with sign reversing and variable permutation methods. Our countermeasures significantly increase the difficulty of the resulting optimization problem, rendering the attacks we presented ineffective. However, it is unclear whether other types of attack could be applied in such cases, but the complexity of exploiting these mitigated approaches appears to be significantly higher.

This paper is organized as follows. Section 2 explores legal risks in cloud computing and obfuscation as a defense. Section 3 provides an overview of current privacy-preserving methods for quantum optimization, as well as background on the Trivium cipher family. Section 4 describes the attack framework, while Section 5 details our attacks. Section 6 demonstrates their practical feasibility. Section 7 outlines possible countermeasures to defend against our attacks. Finally, Section 8 concludes the paper.

2 Legal Compliance Risks in Cloud Computing and Obfuscation Defense

2.1 Legal compliance risks in cloud computing

Outsourcing a computational task to a service provider is feasible if there is sufficient trust, the service provider is held accountable for their actions and their consequences (under civil, criminal, and administrative law), and/or technical safeguards are in place to limit the potential for misconduct.

When considering the responsibility of cloud services, the general data protection rules from criminal law may apply. For example, German criminal law [40] defines penalties for computer fraud (Section 263a). Among others, "making unauthorized use of data" falls into the category of computer fraud. A precondition is *the intention of obtaining an unlawful pecuniary benefit for themselves or a third party*. In this case, the penalty is up to 5 years' imprisonment. The keywords are "unauthorized use of data" and "benefit." In turn, Polish criminal law [36] states in Article 266 that *whoever, contrary to the provisions of the law or an obligation assumed, discloses or uses information that he or she has become aware of in connection with his or her function, work, ... or scientific activity, shall be subject to a fine, restriction of freedom, or imprisonment for up to 2 years.* There is no condition of obtaining a benefit," while the penalties are substantially lower. In both cases, a contract stating confidentiality conditions is necessary for criminal prosecution. However, it is worth noting that the penalties might be too low to deter cases where the value of the data leaked is very high. This may include, in particular, cryptanalytic tasks and data of strategic importance.

There are situations where specific regulations apply, and additional threats exist for the offender. A prime example is GDPR [38] and violations of personal data protection rules. In such cases, heavy administrative fines can be imposed alongside criminal penalties. For unauthorized personal data transfer, the maximum fine is 10,000,000 EUR or up to 2% of the offender's total worldwide annual turnover (whichever is greater). In the case of cloud-based quantum computing services, the issue might be that the service provider cannot determine beforehand whether the data being processed concerns an *identifiable person* and, therefore, whether processing it falls into the category of personal data protection.

On the other hand, the problem with contractual protection and criminal law is that gathering evidence against a cloud service can sometimes be difficult or even impossible. If the leaked data is a decryption key, the party gaining access to a particular plaintext encrypted with this key may use this knowledge in a covert manner. This, in turn, creates a legal risk for the customer, who could be accused of delivering data to a service provider without ensuring adequate control.

Another practical problem is the applicability of local law. If services are ordered in a third country, the place of contracting and the place where services are provided may, according to the contract, result in the simultaneous application of legal rules from different countries, even if they are incompatible. As a result, the customer might be at a disadvantage if the services are operated in a country with a Common Law system, while the customer originates from a country with a Continental Law system.

The situation becomes complicated when laws such as the U.S. CLOUD Act [15] apply. According to the act, *A provider of ... or*

remote computing service shall comply with the obligations of this chapter to preserve, backup, or disclose the contents of a wire or electronic communication and any record or other information pertaining to a customer or subscriber. This means that for a computing task, a customer's input and output must be retained and disclosed upon demand by relevant authorities. The scope of the CLOUD Act is vast: the information can be requested from U.S. companies and organizations *regardless of whether such communication, record, or other information is located within or outside of the United States.* As a result, if processing takes place on foreign soil and is regulated by regional law, fulfilling the obligations stated in the CLOUD Act may be forbidden by regional law and subject to severe penalties.

The CLOUD Act establishes safeguards that may be applied to protect a provider in specific situations. A provider *may file a motion to modify or quash the legal process* in the U.S.; however, this step is admissible only if the provider *reasonably believes*:

(1) the customer or subscriber is not a United States person and does not reside in the United States; and
(2) the required disclosure would create a material risk of violating the laws of a qualifying foreign government.

It is important to note that the term *United States person* is very broad and includes *unincorporated associations with a substantial number of members* that fall into this category. The critical condition is the second one, which applies only to *qualified* countries. The definition of qualified countries is also provided in the CLOUD Act and specifically requires an executive agreement between the United States and the country in question. As a result, providers offering services in countries without such agreements cannot file a motion to quash the process.

Ongoing disputes and discussions concerning the CLOUD Act persist in Europe, with strong opposition to its admissibility in the region. From the perspective of a service provider, these legal and political disputes create uncertainty and fail to provide a reliable foundation for conducting business.

2.2 Obfuscation as a legal defense

Obfuscating the input and output of a cloud computation is a highly effective strategy to mitigate the business risks faced by a cloud operator. Obfuscation serves multiple purposes:

(1) The input and output data, being obfuscated, can be disclosed to authorities in compliance with regulations like the CLOUD Act without violating the rights and interests of the customer, as the data are rendered useless in this form.
(2) At the same time, the provider is protected from accusations of illegal data transfer, as any leaked data are unintelligible and of no value. In cases of potential criminal prosecution, the act would be considered to have *minor social harmfulness*, leading to the dismissal of the case. For personal data protection, effective pseudonymization of the input addresses concerns related to GDPR [38].
(3) Obfuscation reduces the effort required for cybersecurity protection of services. Even if a security breach occurs and the fault is attributed to the provider, civil claims would reflect the actual damages suffered by the customer, which should be minimal or negligible if the data are well-obfuscated.
(4) It simplifies risk analysis and facilitates contractual agreements with service providers. In many cases, *due diligence* is required, and failure to meet this obligation is punishable. Obfuscation reduces this burden by mitigating associated risks.

If the computation input is encrypted using a homomorphic encryption scheme, the provider performing computations on the encrypted data is safeguarded in the ways discussed above. The same applies to end-to-end encryption for communication services. For tasks like quantum optimization involving sensitive information, strong privacy guarantees are essential for both the customer and the service provider.

3 Background and Related Work

In this section, we provide an overview of optimization problem representations, existing privacy-preserving methods for quantum optimization, and the Trivium cipher family.

3.1 Representation of optimization problems

Many optimization problems can be represented as either a QUBO problem or an Ising model, both of which are required formats for quantum solvers. These two representations are equivalent and can be easily transformed into one another.

QUBO Problem. A QUBO problem is an optimization problem that uses binary variables, defined in Expression (1).

$$\min_{x\in\{0,1\}^n} x^T Q x. \tag{1}$$

Problem can also be represented as minimizing the folowing expression: $f(x) = \sum_{i=0}^{n-1} Q_{i,i}x_i + \sum_{i,j=0}^{n} Q_{i,j}x_i x_j$. The vector x contains n binary variables, and Q is the matrix representing the problem:

$$Q = \begin{bmatrix} Q_{0,0} & Q_{0,1} & \cdots & Q_{0,n-1} \\ Q_{1,0} & Q_{1,1} & \cdots & Q_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{n-1,0} & Q_{n-1,1} & \cdots & Q_{n-1,n-1} \end{bmatrix}. \tag{2}$$

Ising Model. The Ising model is an alternative representation of an optimization problem. It can be viewed as minimizing the expression: $f(s) = \sum_{i=0}^{n-1} h_i s_i + \sum_{i,j=0}^{n-1} J_{i,j} s_i s_j$. The vector s is called the state, and each variable $s_i \in \{-1, 1\}$ is called a spin. In practice, the problem can be characterized using: a vector of biases (3) and a matrix describing connections between variables (4):

$$h = \begin{bmatrix} h_0 & h_1 & \cdots & h_{n-1} \end{bmatrix}^T; \tag{3}$$

$$J = \begin{bmatrix} J_{0,0} & J_{0,1} & \cdots & J_{0,n-1} \\ J_{1,0} & J_{1,1} & \cdots & J_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ J_{n-1,0} & J_{n-1,1} & \cdots & J_{n-1,n-1} \end{bmatrix}. \tag{4}$$

The coefficients of each matrix can be determined using the matrix coefficients of the corresponding problem [1]. When moving from the Ising model to the QUBO problem, the matrix is determined as Equation (5).

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

In reverse, the conversion is analogous.

3.2 Privacy-preserving methods for quantum optimization

Currently available privacy-preserving methods [1, 37, 53] for quantum optimizations work by obfuscating an optimization problem so that it can be safely shared with a third party for solving. These methods work for both representations (QUBO problem and Ising model) of optimization problems. They can be categorized based on whether they increase the complexity of solving the obfuscated problem, the obfuscation method used (e.g., sign reversing, variable permutation, a combination of both, or other), and whether an attack against the method exists. Table 1 provides a comparison of privacy-preserving methods for quantum optimization. [37] focuses on applying the spin reversal transformation to the Ising problem. The simplest way to protect privacy is to obfuscate the problem using a homomorphic algorithm, employing a random sequence to reverse the signs in an Ising problem instance. Importantly, the minimal energy of the instance remains unchanged. As illustrated in [37], once decrypted, the solution to the obfuscated problem is identical to the solution of the original problem. The authors further demonstrate the efficiency of this method.

In [1], several distinct obfuscation methods are proposed for data processing. These methods include:

(1) Enigma-I, which obfuscates problem coefficients to introduce random bit flips, effectively concealing coefficients and outcomes through sign reversal, similar to the Spin Reversal Transformation [37].
(2) Enigma-II, which modifies the graph structure of QUBO problems by adding decoy qubits, followed by the random permutation of variables Π. The QUBO problem is then transformed into the Ising form, and sign reversal is applied.
(3) Enigma-III, which obscures connectivity information by transforming the graph to ensure all nodes have an identical number of connections. This method builds on Enigma-II by adding additional edges.

Out of the current privacy-preserving methods, only the first three methods listed in Table 1 do not increase the complexity of solving the obfuscated problem. The inherent behavior of optimization problems makes it difficult to determine the time complexity of a given problem a priori. However, these methods ensure that:

(1) the absolute values of the coefficients remain unchanged;
(2) the density of the matrix in the Ising representation does not increase;
(3) the total number of variables in the problem remains constant.

Therefore, the main parameters that influence the time complexity of the quantum optimization solver remain unchanged. As the time complexity of solving optimization problems using quantum annealing has not yet been fully resolved, one typically approximates the complexity of quantum annealing [30] as $O\left(e^{an^b}\right)$, for positive coefficients a, b, where $b \leq 1$ and n is the number of variables of the problem.

Therefore, the time complexity remains at the same level for those methods. However, we note that in some real-world applications, the three methods mentioned above may slightly increase or decrease the problem's time complexity.

Conversely, if the size of the optimization (QUBO) problem increases, for example, by employing privacy-enhancing methods that require the addition of decoy qubits (as in the full Enigma-II and full Enigma-III methods), then, the time complexity of solving such problems also increases. In fact, the full Enigma-3 algorithm [1] and the combination of digit-wise splitting and matrix permutation [53] cause an increase in both the total number of variables and the density of the problem. Further, the combination of digit-wise splitting and matrix permutation [53] requires solving multiple instances of optimization problems of the same size as the original problem. This approach further implies an increase in the time complexity of solving optimization problems obfuscated using this method.

Consequently, in this paper, we analyze the following methods for our attacks, which *do not* increase the complexity of solving the obfuscated problem and which *have not yet* been broken.

(1) **Variable permutation:** variable permutation, a component of Enigma-II [1]
(2) **Sign reversing + variable permutation:** simplification of Enigma-II [1]; we do not consider the addition of decoy variables.

Simplified Enigma-II (spin reversal transformation combined with variable permutation). Below, we provide details of the simplified Enigma-II method, which is the focus of our attack. This obfuscation scheme, presented in [1], employs the spin reversal transformation described in [37], variable permutations, and the addition of decoy variables. Based on the assumptions made, we simplify the scheme to the following form:

- in the first step, permutation π (given by the permutation matrix Π) of variables and coefficients of QUBO matrix is performed:

$$\begin{aligned} \tilde{Q} &= \Pi \cdot Q \cdot \Pi^T, \\ \tilde{x} &= \Pi \cdot x; \end{aligned} \tag{6}$$

- in the second step spin reversal transformation is performed on matrices $\tilde{h}$ and $\tilde{J}$.

Then, the obfuscation process works as follows:

(1) the client generates a secret key $y = (y_0, y_1, \ldots, y_{n-1})$ and a permutation matrix Π;
(2) using Equation (6), the client applies the following permutation: $\tilde{Q} = \Pi \cdot Q \cdot \Pi^T$,
(3) the client performs the spin reversal transformation and computes $\hat{h}$ and $\hat{J}$:

$$\begin{aligned} \hat{h}_i &= (-1)^{y_i} \tilde{h}_i, \\ \hat{J}_{i,j} &= (-1)^{y_i + y_j} \tilde{J}_{i,j}. \end{aligned} \tag{7}$$

(4) the client sends the obfuscated problem to the quantum cloud service and receives the solution $\hat{s}$;
(5) the client retrieves the solution using Equation (8) and permutation used:

$$s_i = (-1)^{y_{\pi(i)}} \hat{s}_{\pi}(i). \tag{8}$$

Table 1: Comparison of privacy-preserving methods for quantum optimization.

Obfuscation type	Obfuscation technique	Complexity	Remarks
Sign reversing	Spin reversal transformation [37] and Enigma-I [1]	Unchanged	Attack presented in [29]
Variable permutation	Variable permutation, part of Enigma-II [1]	Unchanged	Attack presented in this paper
Variable permutation + sign reversing	Simplified Enigma-II [1], without adding variables	Unchanged	Attack presented in this paper
Variable permutation + sign reversing	Full Enigma-3 algorithm [1]	Increased	No known attack
Other	Combination of digit-wise splitting and matrix permutation [53]	Increased	No known attack

3.3 Trivium cipher family

We demonstrate our attack in the context of an optimization problem, specifically an algebraic attack against the Trivium cipher family. We stress that we *do not* attempt to break the cipher itself; our attack targets the respective privacy-preserving methods. This paper uses the Trivium cipher [16] and its parameterized versions presented in [49]. The original Trivium cipher consists of three internal shift registers with nonlinear feedback, which are connected to form a circular structure. The output bit from one register is passed to the next: t_3 to register 1, t_1 to register 2, and t_2 to register 3. We can distinguish three types of parameterized cipher based on the number of internal registers. The internal state update is done analogously to the original version of the cipher. The following versions can be specified as:

- Univium, with one internal register and characterized by parameters(u_1, u_2, n_1):

$$\begin{aligned} z &= s_{3u_1} \oplus s_{3n_1}, \\ t_1 &= s_{3u_1} \oplus s_{3n_1} \oplus s_{3n_1-2} \cdot s_{3n_1-1} \oplus s_{3u_2}. \end{aligned} \tag{9}$$

- Bivium, with two internal registers and characterized by parameters(u_1, u_2, n_1) and (u_3, u_4, n_2):

$$\begin{aligned} z &= s_{3u_1} \oplus s_{3n_1} \oplus s_{3u_3} \oplus s_{3n_2}, \\ t_1 &= s_{3u_1} \oplus s_{3n_1} \oplus s_{3n_1-2} \cdot s_{3n_1-1} \oplus s_{3u_4}, \\ t_2 &= s_{3u_3} \oplus s_{3n_2} \oplus s_{3n_2-2} \cdot s_{3n_2-1} \oplus s_{3u_2}. \end{aligned} \tag{10}$$

- Trivium, with three internal registers and characterized by parameters(u_1, u_2, n_1), (u_3, u_4, n_2) and (u_5, u_6, n_3):

$$\begin{aligned} z &= s_{3u_1} \oplus s_{3n_1} \oplus s_{3u_3} \oplus s_{3n_2} \oplus s_{3u_5} \oplus s_{3n_3}, \\ t_1 &= s_{3u_1} \oplus s_{3n_1} \oplus s_{3n_1-2} \cdot s_{3n_1-1} \oplus s_{3u_4}, \\ t_2 &= s_{3u_3} \oplus s_{3n_2} \oplus s_{3n_2-2} \cdot s_{3n_2-1} \oplus s_{3u_6}, \\ t_3 &= s_{3u_5} \oplus s_{3n_3} \oplus s_{3n_3-2} \cdot s_{3n_3-1} \oplus s_{3u_2}. \end{aligned} \tag{11}$$

The original description of the cipher assumes numbering starting from 1. For ease of calculation, the register bits will be numbered from 0.

4 Attack Framework

In this section, we first describe a generic framework for solving an optimization task using a cloud-based quantum computing provider (see Figure 1). We then describe how our attack fits within this framework (see Figure 3).

Figure 1 illustrates the steps taken by a client to solve a computational task using a cloud-based quantum computing provider. The client starts with a task suitable for a quantum annealer or a QAOA solver. Then, the client transforms the computational task into an optimization problem represented as either a QUBO problem or an Ising model. This transformation is performed using a dedicated method, such as those described in [17, 25, 28]. The transformation ensures that the task can be solved by the quantum computing provider. Next, the client selects an obfuscation method (described in Section 3.2) to produce an obfuscated optimization problem. Once obfuscated, the optimization problem is sent to the provider for processing. The provider uses its solver (either a quantum annealer or QAOA solver) to compute an obfuscated solution to the obfuscated problem. The provider then returns this solution to the client. Finally, the client deobfuscates the solution. By applying the same obfuscation technique used earlier, the client retrieves the solution, which corresponds to the optimal solution for the original task.

In our attacks, we assume that the attacker has access either to the obfuscated optimization problem alone or to both the obfuscated problem and its solution. If the attacker has access only to the obfuscated problem, it can use a quantum solver to compute the corresponding obfuscated solution.

5 Attacks Against Privacy-Preserving Quantum Optimization Methods

In this section, we present our attacks. First, we demonstrate how to identify the substitutions resulting from the Rosenberg linearization [24], which forms the basis of the attack on the variable permutation method. We then describe both attacks in detail. The first attack focuses on recovering the permutation by leveraging knowledge of the Rosenberg substitutions. The second attack combines the first attack with an attack on sign reserving presented in [29].

5.1 Vulnerability resulting from the Rosenberg linearization

The basis for performing an attack on the variable permutation method is knowledge of the application of the Rosenberg linearization [24] represented by Equation (12),

$$x_i x_j \leq x_k + 2(x_i x_j - 2x_k(x_i + x_j) + 3x_k). \tag{12}$$

The inequality allows the substitution for each quadratic monomial of a new variable with a penalty represented by Equation (13),

$$pen_{i,j\to k} = 2x_i x_j - 4x_k x_i - 4x_k x_j + 6x_k. \tag{13}$$

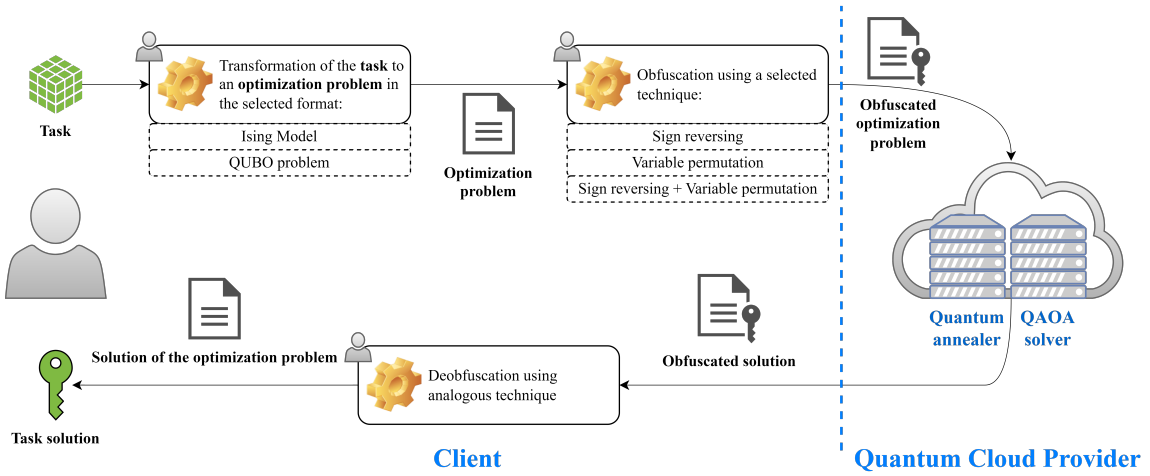

Figure 1: Generic framework for solving an optimization task using cloud-based quantum computing provider.

As shown in the algebraic attack transformation method for the QUBO problem presented in [6], the following component appears in the final polynomial which describes this QUBO problem.

$$Pen = c \sum_{i,j,k} pen_{i,j\to k} = \sum_{i,j,k} c \cdot pen_{i,j\to k}, \tag{14}$$

where c is the arbitrarily chosen constant. For the purpose of this analysis, let us assume that the value of $c = 10$. Substituting Equation (13) yields the following: $Pen = \sum_{i,j,k} 2c \cdot x_i x_j - 4c \cdot x_k x_i - 4c \cdot x_k x_j + 6c \cdot x_k$.

It follows that each substitution directly impacts the final form of the QUBO matrix. Each substitution introduces three specific coefficients:

- $-4c$ for two quadratic monomials $x_k x_i$ and $x_k x_j$, above the main diagonal;
- $6c + 1$ for a single linear monomial x_k, on the main diagonal.

The permutation of variables does not alter the coefficients in the matrix. As a result, even after permutation, the substitutions remain identifiable allowing one to find the partial permutations and the paths required to deobfuscate the QUBO problem. This knowledge of substitutions can be leveraged further in the attack. We show this below using the Trivium cipher as an example.

5.2 Finding path: Attack against the variable permutation method

In the matrices obtained for the QUBO problem corresponding to the algebraic attack on the Trivium cipher, we can easily identify the coefficients corresponding to the aforementioned monomials. There are 3 values in the selected columns: two -40 for quadratic monomials and 61 for linear monomials, representing an additional variable used in the linearization process. From the matrix, a set of tuples of the form shown: $(x_i, x_j : x_k)$ can be derived. The size of the set depends on the chosen cipher (see Section 3.3 for details):

- for Univium, $3n_1$ tuples are obtained, which can be arranged in 2 paths, with a length of $3n_1$ each, as shown in Equation (15);
- for Bivium, $2 \cdot 3n_2$ tuples are obtained, which can be arranged in 4 paths, with a length of $3n_2$ each, (2 for each internal register, analogous to Equation (15));
- for Trivium, $3 \cdot 3n_3$ tuples are obtained, which can be arranged in 6 paths, with a length of $3n_3$ each, (2 for each internal register, analogous to Equation (15)).

As illustrated by the construction of the Trivium cipher, shown in Section 3.3, quadratic monomials overlap in pairs, as shown in Figure 2. On this basis, the substitutions can be organized into the

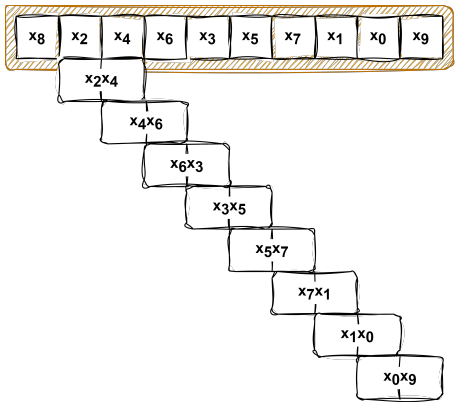

Figure 2: Example permutation of internal state with overlapping quadratic monomials.

following sequences:

$$\begin{aligned} R_t &= sub_0, sub_1, sub_2, \ldots, sub_{3n_t-1}, \\ R_t^{-1} &= sub_{3n_t-1}, sub_{3n_t-2}, sub_{3n_t-3} \ldots, sub_0, \end{aligned} \tag{15}$$

where:

- $t \in \{1, 2, 3\}$, depends on the type of cipher and identifies the number of internal registers;
- $sub_{idx} = (x_i, x_j : x_k)$, for the corresponding i, j, k from the set of substitutions;
- $sub_0 = (x_i, x_j : x_k)$, if x_i does not have a precursor;

- $sub_{idx} = (x_i, x_j : x_k)$, and $s_{idx+1} = (x'_i, x'_j : x'_k)$ only if $x_j = x'_i$;
- $sub_{3n_t-1} = (x_i, x_j : x_k)$, if x_j does not have a successor.

Finding the sequence described above is equivalent to finding a fragment of an obfuscating permutation. By selecting the columns corresponding to the first elements (and for the last word, both values in order) of each pair in the sequence, we recover the variables in their original order before the permutation. As the cipher design shows, the string does not cover the first bit and has length $3n_t + 1$, as illustrated in Figure 2. Another challenge involves the number of paths found and determining which one is correct.

To better illustrate the above technique, we will discuss the example shown in Figure 2. The internal state consists of a single 9-bit register, similar to the Univium cipher. We assume that the quadratic monomials depicted in Figure 2 were read from the QUBO matrix, following the approach described in Section 5.1. For simplicity, we omit the substituted variable for the given quadratic monomial in this analysis.

Next, we arrange the determined quadratic monomials into two sequences, as described in Equation 15. For clarity, we will focus only on R_1 in the following. First, we determine the first and last elements of the sequence:

- $sub_0 = (x_2, x_4)$, x_2 occurs only once, so it has no predecessor or successor. Without loss of generality, we assume it will be the initial word;
- $sub_7 = (x_0, x_9)$, x_9 occurs only once, and since the initial word is already determined, we assume it is the last word.

Then, we complete the middle of the sequence: $sub_1 = (x_4, x_6)$, $sub_2 = (x_6, x_3)$, $sub_3 = (x_3, x_5)$, $sub_4 = (x_5, x_7)$, $sub_5 = (x_7, x_1)$, $sub_6 = (x_1, x_0)$. At this point, we can designate the internal state as one of the two possibilities:

$$state \leftarrow (-, x_2, x_4, x_6, x_3, x_5, x_7, x_1, x_0, x_9)$$
$$state \leftarrow (-, x_9 x_0, x_1, x_7, x_5, x_3, x_6, x_4, x_2)$$

By comparing these options with the state shown in Figure 2, we observe that the first determined state is correct, albeit with one missing bit.

With some modifications, the presented technique can be applied to the combined method (sign reversing + variable permutation), which we show next.

5.3 Attack against the simplified Enigma-II algorithm

The simplified Enigma-II algorithms, presented in Section 3.2, combines two obfuscation techniques: sign reversing (in particular, spin reversal transformation) and variable permutation.

The method for decrypting spin reversal transformation (equivalent to the Enigma-I model) is detailed in [29] for the E_0 stream cipher. Extending this method to the Trivium cipher family is straightforward.

We use two techniques, with appropriate modifications, to perform the attack:

- the recovery of sequences from the QUBO matrix, as described earlier;
- an attack on spin reversal transformation, as detailed in [29].

Our attack consists of two phases: identifying potential permutations and recovering the solution using spin reversal transformation.

In the first phase, we identify potential permutations by refining the previously described method with the following modifications:

- due to sign changes in the Ising model, the coefficients on the QUBO matrix's main diagonal also change; we search for columns containing two values $Q_{i,j}$ such that $|Q_{i,j}| = 40$;
- to simplify the data structure, we store only the quadratic monomial information, we discard the new variable, modifying the collected tuple to a pair of (x_i, x_j);
- we select correct permutations during the second phase of the attack.

In the second phase, we arrange a system of equations based on the sequences R_i, with which each variable corresponding to an element of the parametrized bias vector h_P. For each element, we consider two equations : $\hat{h} - h^P = 0$ (this equation has a solution in the given set if the coefficient is not blinded) and $\hat{h} + h^P = 0$ (this equation has a solution in the set if the coefficient is blinded).

If neither equation has a solution within the set $\{-1, 1\}$, then the selected triple permutation should be rejected. Ultimately, this phase yields a set of ordered triples that represent permutation fragments for each register. By combining permutations π with knowledge of inverted signs, we can derive algebraic description of internal state, presented in Equation (17), to recover the solution from the obfuscated solution.

The number of potential solutions depends on the parameters of the cipher. For Trivium, if the condition in Equation (16) is not satisfied, the parametrized matrix equations corresponding to successive registers will repeat. We provide the analysis and derivation of the condition in Section 5.5.

$$3n_1 - 3u_1 \neq 3n_2 - 3u_3 \neq 3n_3 - 3u_5. \tag{16}$$

Repetitions may introduce false positives among the analyzed triplets. Equations that cannot be solved due to too many unknown variables may indicate triplets that should be discarded. After completing the attack, we get the following:

- possible keystream: some bits may be missing, but the gaps are minimal (a few bits at most);
- possible algebraic descriptions of internal states: these are in a format present on Equation (17); the internal state is determined by the bits of the obfuscated solution $\hat{s}_i \in \{-1, 1\}$ in the Ising model.

$$s_i = (-1)^{y_{\tilde{\pi}(i)}} \hat{s}_{\tilde{\pi}(i)}. \tag{17}$$

We note that the algebraic description of initial internal designated in the attack covers not only the initial internal state of the cipher but also the state after several steps. This approach allows the initial state to be determined even if some bits were not determined. Note that $\tilde{\Pi}$ is a partial permutation, which corresponds to selected paths R_m, R_n, R_o.

The correct internal state can be verified by generating an output string consistent with the recovered state.

Algorithm 1 provides an overview of the attack. We assume that the attacker has access to the following:

- the obfuscated problem $\hat{h}, \hat{J}$ or $\hat{Q}$, sent by customer to cloud-based quantum computing provider; the problem can be

Algorithm 1 Description of the attack on Enigma-II (sign reversing + variable permutation

$R_1, R_1^{-1}, R_2, R_2^{-1}, R_3, R_3^{-1} \leftarrow O(\hat{Q})$ ▷ R_i^{-1} is R_i in reverse order
$\mathcal{R}_1 = \{R_1, R_1^{-1}\}, \mathcal{R}_2 = \{R_2, R_2^{-1}\}, \mathcal{R}_3 = \{R_3, R_3^{-1}\}$
$h_P, J_P \leftarrow \phi(\hat{Q})$
for $\tilde{\Pi}_i \in \mathcal{R}_m \times \mathcal{R}_n \times \mathcal{R}_o, i = \overline{1, 2^3 \cdot 3!}$ **do**
  $\mathcal{A}_j \leftarrow \hat{h}_{\tilde{\pi}_i(j)} - h^P_{\tilde{\pi}_i(j)}, j = \overline{1, 3n_t}$ ▷ $t \in \{1, 2, 3\}$
  $z, y \leftarrow \sigma(\mathcal{A})$
  if $\perp$ is obtained **then**
    Discard the selected permutation Π_i
  else
    Determine the potential internal state s:
    $s_j = (-1)^{y_{\tilde{\pi}_i(j)}} \hat{s}_{\tilde{\pi}_i(j)}, j = \overline{1, 3n_t}$
  end if
  Keep the potential internal state s and keystream z
end for
return All stored potential internal states and keystream

easily converted from the Ising model to the QUBO problem, as these representations can be used interchangeably (see Section 3.1 for discussion);
- the obfuscated solution $\hat{s}$ to the above problem; the attacker can either obtain it from the provider or send the obfuscated problem to the provider itself for solving it if only has access to the obfuscated problem;
- oracle O, which returns a set of potential permutations as described earlier;
- oracle ϕ, as described in [29], which provides a parameterized model for a given cipher;
- algorithm σ which solves an obfuscated linear system of equations; this algorithm, as described earlier, returns either a potential keystream z and a potential of obfuscated key fragment y, or failure indicator $\perp$.

It is important to note that after identifying all three paths R_m, R_n, R_o, it is necessary to determine their order (which path is first, second, and third) as well as the direction of each path. It means that for three paths, one has to check $3! \cdot 2^3 = 48$ possibilities. We denote the partial permutation, which is a composite of selected paths, as $\tilde{\Pi}_i$. However, if condition (16) is met, then Algorithm 1 may be optimized, and then takes under 2 minutes.

Further, it is important to note that for ciphers satisfying the assumptions outlined in Section 5.5 (regarding the existence of a solution), only one specific permutation combination will resolve the system of equations without resulting in a contradiction.

5.4 Feasibility of Our Attacks with Sign Reversal and Variable Permutation Applied in Any Order: Proof and Example

We now show that our attack works regardless of the order in which sign reversing and variable permutation are applied.

Let us note that if a permutation (described by the permutation matrix Π) is applied first, followed by a spin-reversal transformation (described by the involutory matrix E, where only the diagonal coefficients are non-zero and belong to the set $\{-1, 1\}$), the following holds:

$$\begin{aligned} f &= s \cdot J \cdot s^T + s \cdot h^T \\ &= (s \cdot \Pi) \cdot (\Pi^T \cdot J \cdot \Pi) \cdot (s \cdot \Pi)^T + (s \cdot \Pi) \cdot (h \cdot \Pi)^T \\ &= (s \cdot \Pi \cdot E) \cdot (E \cdot \Pi^T \cdot J \cdot \Pi \cdot E) \cdot (s \cdot \Pi \cdot E)^T \\ &+ (s \cdot \Pi \cdot E) \cdot (h \cdot \Pi \cdot E)^T \\ &= \hat{s} \cdot \hat{J} \cdot \hat{s}^T + \hat{s} \cdot \hat{h}^T, \end{aligned}$$

where $\hat{s} = s \cdot \Pi \cdot E$, $\hat{J} = E \cdot \Pi^T \cdot J \cdot \Pi \cdot E$, and $\hat{h} = h \cdot \Pi \cdot E$. We also use the property that for every permutation matrix Π, $\Pi^{-1} = \Pi^T$.

However, if these operations are applied in reverse order–first performing the spin–reversal transformation (described by the involutory matrix E'), followed by the permutation transformation (described by the permutation matrix Π')-to obtain the same matrices $\hat{J} = E \cdot \Pi^T \cdot J \cdot \Pi \cdot E$ and $\hat{h} = h \cdot \Pi \cdot E$, the following must hold:

$$\begin{aligned} f &= s \cdot J \cdot s^T + s \cdot h^T \\ &= (s \cdot E') \cdot (E'^T \cdot J \cdot E') \cdot (s \cdot E')^T + (s \cdot E') \cdot (h \cdot E')^T \\ &= (s \cdot E' \cdot \Pi') \cdot ((\Pi')^T \cdot E' \cdot J \cdot E' \cdot (\Pi)') \cdot (s \cdot E' \cdot \Pi')^T \\ &+ (s \cdot E' \cdot \Pi') \cdot (h \cdot E' \cdot \Pi')^T \\ &= \hat{s} \cdot \hat{J} \cdot \hat{s}^T + \hat{s} \cdot \hat{h}^T. \end{aligned}$$

This implies that the following condition must hold:

$$\Pi \cdot E = E' \cdot \Pi'. \tag{18}$$

However, it is worth noting that given $\Pi \cdot E$ (which is what our attack ultimately identifies), it is straightforward to determine $E' \cdot \Pi'$, for which Equation (18) holds. Since E is a matrix with only diagonal coefficients that are non-zero (equal to -1 or 1), it is easy to factorize $\Pi \cdot E$ as the product of $E' \cdot \Pi'$. It can be shown that $\Pi = \Pi'$, and consequently, $E' = \Pi^T \cdot E \cdot \Pi$. Therefore, our attack remains effective regardless of the order in which these privacy-preserving operations are applied.

Example 5.1. Let

$$\Pi = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

and

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Then

$$\Pi \cdot E = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$

On the other hand, if $\Pi \cdot E = E' \cdot \Pi' = E' \cdot \Pi$, then

$$E' = \Pi^T \cdot E \cdot \Pi = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

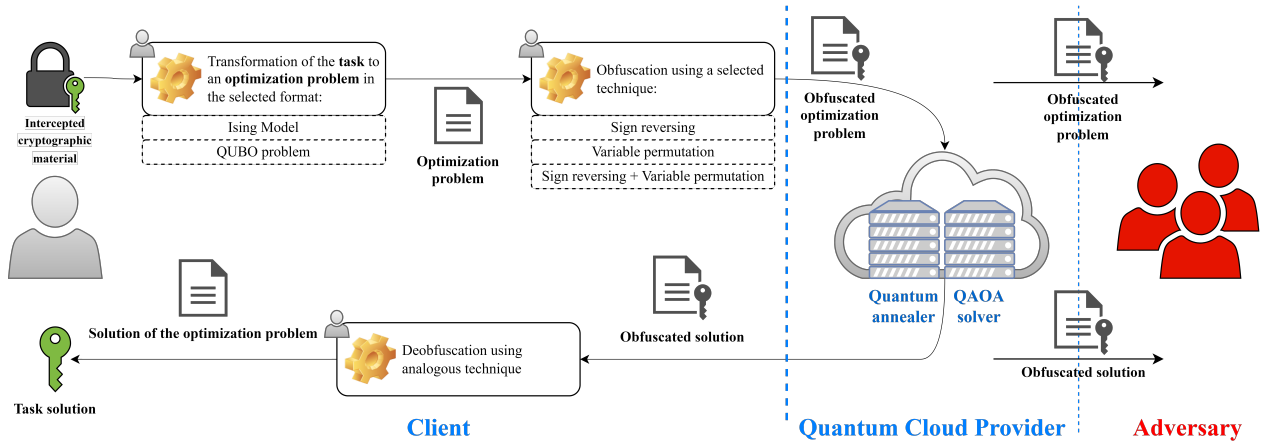

Figure 3: Attack framework using a framework for solving an optimization task using a cloud-based quantum computing provider.

It means that

$$E' = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}.$$

5.5 Analysis of our attacks

We now present the rationale supporting the correctness and effectiveness of the proposed attacks. For the attack to succeed, the arranged equations must be solvable. Let us analyze the modified equation describing the output bit. We assume that $U_i = 3u_i$ and $l_j = 3n_j$. Let $z_i + x_{U_1+i} + x_{l_1+i} + x_{U_3+i} + x_{l_2+i} + x_{U_5+i} + x_{l_3+i} - 2k_0 - 4k_1 = 0$.. After squaring both sides of the equation above, we obtain

$$\begin{aligned} &z_i + x_{U_1+i} + x_{l_1+i} + x_{U_3+i} + x_{l_2+i} + x_{U_5+i} + x_{l_3+i} + 4k_0 + 16k_1 \\ &+2z_i x_{U_1+i} + 2z_i x_{l_1+i} + 2z_i x_{U_3+i} + 2z_i x_{l_2+i} + 2z_i x_{U_5+i} + 2z_i x_{l_3+i} \\ &-4z_i k_0 - 8z_i k_1 + \ldots = 0. \end{aligned}$$

Since we are looking for keystream-dependent coefficients, we simplify the analysis by omitting expressions where z_i does not appear:

$$\begin{aligned} &\ldots + (1+2z_i)x_{3u_1+i} + (1+2z_i)x_{3n_1+i} + (1+2z_i)x_{3u_3+i} \\ &+(1+2z_i)x_{3n_2+i} + (1+2z_i)x_{3u_5+i} + (1+2z_i)x_{3n_3+i} \\ &+4(1-z_i)k_0 + 8(1-z_i)k_1 + \ldots = 0. \end{aligned}$$

From [49], we have: $u_1 < u_2 < n_1 < u_3 < u_4 < n_2 < u_5 < u_6 < n_3$. Now, we make the following substitutions:

- $\Delta_A = 3n_1 - 3u_1$;
- $\Delta_B = 3n_2 - 3u_3$;
- $\Delta_C = 3n_3 - 3u_5$.

Using the substitutions above, we divide the equation into register-specific parts based on the indices of the variables appearing in the equation: part for register 1: $(1+2z_i)x_{3u_1+i}+(1+2z_i)x_{3u_1+\Delta_A+i}$, part for register 2: $(1+2z_i)x_{3u_3+i}+(1+2z_i)x_{3u_3+\Delta_B+i}$, part for register 3: $(1+2z_i)x_{3u_5+i} + (1+2z_i)x_{3u_5+\Delta_C+i}$. As we can see from the above equations, for a certain i-th equation, the coefficient will depend on two variables, neither of which necessarily occurs independently in previous equations. If $\Delta_A = \Delta_B$ or $\Delta_B = \Delta_C$ or $\Delta_A = \Delta_C$, the same situation will occur for the corresponding registers. This leads to a situation in which some of the z_i variables may remain undetermined, because, for at least two registers. the patterns will be identical. However, if the condition (16) is met, then it holds that $\Delta_A \neq \Delta_B \neq \Delta_C$. Consequently, for each register a different pattern for path finding is used, allowing us to distinguish which path corresponds to register 1, register 2, and register 3. Based on this differentiation, we can determine the condition shown in Equation (16) when the equations corresponding to successive registers will be linearly independent. For example, for the coefficients corresponding to the first bits of the registers, three equations with only one variable repeated in pairs will be obtained. Such a configuration of equations allows us to unambiguously solve the obtained system of equations. Furthermore, this leads to a situation in which the whole keystream is recovered if the condition in Equation (16) is met. Notably, this situation holds for the full version of the Trivium cipher.

5.6 Time complexity analysis

We now analyze the time complexity of the attacks we have presented. In their paper, the authors of [1] presented a security analysis of their algorithm, and claimed the complexity of the brute-force attack to be:

$$O((n+m)\cdot(n+m)!\cdot(2^{n+m})), \tag{19}$$

where n is the number of variables in the basic QUBO problem (before the application of the obfuscation of the problem) or Ising model and m is the number of decoy variables. In our case, decoy variables are not being used, so $m = 0$. In this case, Equation (19) reduces to the following form:

$$O(n\cdot n!\cdot 2^n). \tag{20}$$

However, as shown below, our attack for the instance equivalent to the algebraic attack on the Trivium cipher is polynomial in n.

Theorem 5.2. *The algorithm of revealing both keys for spin reversal transformation E and permutation Π for the QUBO problem equivalent algebraic attack on Trivium ciphers family to takes at most*

$O(n^2)$ operations, where n is the number of variables in the QUBO problem (or Ising model).

Proof. Let us assume that we first want to identify all elements that may belong to any path $R_1, \ldots, R_k$, where k is the number of distinct paths. To find the proper permutation, we need to construct these paths. Note that identifying all pairs that may be used to construct all k paths requires $O(n^2)$ operations. The process of finding all pairs is as follows:

(1) search the matrix J row by row;
(2) in each row, check one coefficient if it appears like the result of using Rosenberg linearization. If yes, add pair (x_i, x_j) to the set S of substitutions.

The set S consists of $O(n)$ pairs, which each pair treated as a short path. Since the set S consists of $O(n)$ paths, we join these paths using the method presented in Section 5.2 to obtain k distinct paths R_i, for $i = \overline{1,k}$, as required for the given version of the cipher from Trivium ciphers family (1 path for Univium, 2 paths for Bivium, and 3 paths for Trivium).

After assigning all elements from S, we can construct the missing paths according to Equation (15). Finally, the process above, after $O(n^2)$ operations, returns the k distinct paths $R_1, \ldots, R_k, R_1^{-1}, \ldots, R_k^{-1}$. However, the order of the paths and the direction of each path remains unknown. As a result, $k! \cdot 2^k$ possibilities must be checked. Since $k \in \{1, 2, 3\}$, this does not affect the asymptotic behavior of the time complexity. Therefore, identifying the potential permutations π relevant from the attack's perspective requires $O(n^2)$ operations.

Now, let us address the complexity of the spin reversal transformation. As presented above, our attack on the spin reversal transformation always succeeds in the context of a full Trivium cipher. Therefore, we focus on identifying the spin reversal transformation for this case.

Let us note that, to determine the necessary part of the spin reversal transformation (used to identify whether coefficients associated with internal states are flipped or not), we need to solve at most $O(n)$ equations. These equations are derived solely using the vector h, and the matrix J is not required in this step. Each equation can then be solved sequentially. For every equation, we need to check two possibilities: if the spin reversal transformation key is in this place equal to -1 or 1.

The algorithm starts by analyzing the equations with only one unknown bit of keystream z_i. If it is possible to uniquely determine the value of z_i (possible only for one of the two possibilities for spin reversal, either $z_i = 0$ or $z_i = 1$), then this value of z_i may be substituted in the remaining equations. Solving each equation sequentially makes it possible to find all the necessary solutions in this case. A straightforward analysis shows that $O(n^2)$ operations are required: after solving each equation, its result must be substituted into the remaining equations, ultimately leading to $O(n^2)$ complexity. □

6 Practical Applicability of Our Attacks

In this section, we demonstrate the practicality of our attacks by applying them to specific problems. We begin with a pen-and-paper example of our attack, using the cryptanalysis of the Univium cipher as an optimization problem. Although such examples are unusual (and often impractical to show), we felt it was important to include one to underscore the level of vulnerability in the privacy-preserving methods we examine.

Additionally, we discuss an implementation of our attack in SageMath [44] and present the results of applying it to the cryptanalysis of the full Trivium cipher as an optimization problem.

6.1 Pen-and-paper attack using Univium

To demonstrate how our attack works, we will use a small example of an algebraic attack on the Univium cipher with a 9-bit internal state as the optimization task. We will use the simplified Enigma-II method for obfuscation. An attacker with access to both the obfuscated problem and the obfuscated solution would be able to retrieve the cipher's initial internal state and keystream. Since this is a pen-and-paper example, we will not have access to the obfuscated solution, which requires access to a quantum annealer. Instead, we will demonstrate our attack using only the obfuscated problem, showing that we can retrieve the keystream and the algebraic description of the internal state solely from the obfuscated problem. This description, when paired with the obfuscated solution, would reveal the actual initial internal state.

Our example consists of the following main steps:

(1) define the optimization task (an algebraic attack against Univium with 9 bits);
(2) transform the task to a QUBO problem for the generated keystream z;
(3) obfuscate the generated QUBO problem using the simplified Enigma-II method; to do so, we use the key y and permutation matrix Π;
(4) generate the parameterized QUBO matrix for the task;
(5) perform the attack to derive the algebraic description of the internal state and the keystream;
(6) verify the correctness of the solution using the keystream z, key y and permutation matrix Π.

Equation (21) shows the matrix structure corresponding to the cryptanalysis of the Trivium cipher. Each block, representing a fragment of the internal registers, may vary depending on the selected cipher. For instance, in the case of the Univium cipher, only the fragments corresponding to the a register and k variables will remain in the matrix.

$$\begin{array}{c} \\ a \\ b \\ c \\ k \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & k \end{array} \\ \begin{bmatrix} \mathbf{q_{a,a}} & \mathbf{q_{a,b}} & \mathbf{q_{a,c}} & \mathbf{q_{a,k}} \\ \mathbf{0} & \mathbf{q_{b,b}} & \mathbf{q_{b,c}} & \mathbf{q_{b,k}} \\ \mathbf{0} & \mathbf{0} & \mathbf{q_{c,c}} & \mathbf{q_{c,k}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{q_{k,k}} \end{bmatrix} \end{array} \tag{21}$$

The vector of variables x consists of two parts. The first part consists of the vectors of variables a, b and c, which represent:

- the internal state of successive registers;
- the internal state at successive clock cycles;
- substitutions made during linearization.

The second part is a vector that contains the k variables.

To illustrate how the attack works, we generated a problem for the Univium-$(3, 6, 9)$ cipher and the following stream: $z = (1, 1, 0, 0, 1, 0, 0, 0, 0)$. We then generated a QUBO matrix with 54 variables and obfuscated it using the given parameters, following the method described in [52]. Next, we applied the described attack

Table 2: Summary of the results for our attack in the context of the Univium cipher.

i	Solution	$\pi(i)$	Concealment bit	Correctness
0	This bit was not found.			
1	$-1 \cdot \hat{s}_{33}$	33	1	✓
2	$1 \cdot \hat{s}_{47}$	47	0	✓
3	$1 \cdot \hat{s}_{31}$	31	0	✓
4	$1 \cdot \hat{s}_{22}$	22	0	✓
5	$-1 \cdot \hat{s}_{23}$	23	1	✓
6	This bit was not found.			
7	$1 \cdot \hat{s}_{2}$	2	0	
8	$1 \cdot \hat{s}_{37}$	37	0	✓
9	$1 \cdot \hat{s}_{0}$	0	0	✓

to the generated data. In the initial step, we searched for a sequence of substitutions and found the following partial permutations:

$$\tilde{\Pi}_1 = (-, x_{33}, x_{47}, x_{31}, x_{22}, x_{23}, x_{15}, x_2, x_{37}, x_0, x_{41}),$$
$$\tilde{\Pi}_1^{-1} = (-, x_{41}, x_0, x_{37}, x_2, x_{15}, x_{23}, x_{22}, x_{31}, x_{47}, x_{33}).$$

For the sequence we found, we arranged systems of equations. The correct sequence that allowed us to arrange the system of equations described in the attack discussion was the system of equations for the sequence Π_1. The resulting system of equations is:

	$y_{\tilde{\pi}(i)} = 0$	$y_{\tilde{\pi}(i)} = 1$
$\mathcal{A}_1$:	$z_1 - 9/2 = 7/2$	$z_1 - 9/2 = -7/2$
$\mathcal{A}_2$:	$z_2 - 19/2 = -19/2$	$z_2 - 19/2 = 19/2$
$\mathcal{A}_3$:	$z_3 - 9 = -9$	$z_3 - 9 = 9$
$\mathcal{A}_4$:	$z_4 - 9 = -8$	$z_4 - 9 = 8$
$\mathcal{A}_5$:	$z_5 - 9 = 9$	$z_5 - 9 = -9$
$\mathcal{A}_6$:	$z_0 + z_6 - 17/2 = 15/2$	$z_0 + z_6 - 17/2 = -15/2$
$\mathcal{A}_7$:	$z_7 - 15/2 = -15/2$	$z_7 - 15/2 = 15/2$
$\mathcal{A}_8$:	$z_8 - 17/2 = -17/2$	$z_8 - 17/2 = 17/2$
$\mathcal{A}_9$:	$-17/2 = -17/2$	$-17/2 = 17/2$
$\mathcal{A}_{10}$:	$-5/2 = 5/2$	$-5/2 = -5/2$

As a result of solving the system of equations, the following were obtained: partial keystream: $z' = (-, 1, 0, 0, 1, 0, -, 0, 0)$ and algebraic description of part of the internal state:

$$(-, -\hat{s}_{33}, \hat{s}_{47}, \hat{s}_{31}, \hat{s}_{22}, -\hat{s}_{23}, -, \hat{s}_2, \hat{s}_{37}, \hat{s}_0).$$

As expected, we could not find the entire stream for the Univium cipher. When we analyzed the system of equations, we observed that equation $\mathcal{A}_6$ does not allow us to determine the solutions uniquely.

In the final step, we verify the correctness of the results. Table 2 summarizes the determined algebraic description of the initial internal state. Analyzing the table confirms the correctness of the attack. For each row, except 0 and 6 for which the equations could not be determined, compare the resulting algebraic equation describing the selected state bit with the corresponding bits of key y bit and permutation π elements.

6.2 Practical attack using Trivium

We prepared an implementation of our attack against Enigma-II in SageMath [44] and conducted experiments to verify its correctness and practicality. The SageMath implementation contains 600 lines of code and is not optimized. Completing the entire simulation took approximately 21 hours. We next provide a detailed breakdown of the time spent on different stages of the attack.

In our experiment, we used the original version of the Trivium cipher [16] and generated a keystream of 288 bits. Below we list the steps we performed and the time required for each:

- problem generation: transformation of the algebraic attack on Trivium to the QUBO problem and obfuscation using the simplified Enigma-II method to produce the obfuscated QUBO matrix.
  Time required: 10.5 hours;
- QUBO matrix parameterization: generation of the parameterized matrix for cryptanalysis of the Trivium cipher using the method presented in [29].
  Time required: 9.5 hours;
- attack execution: determination of potential variable permutations, analysis of the determined systems of equations, and determination of the algebraic description of the initial internal state.
  Time required: 1 hour.

As observed in [29] and confirmed by our results, the bottleneck of the attack lies in the generation of the parameterized matrix. The time spent on problem generation and obfuscation does not pose a significant issue for two reasons: first, these steps are typically implemented on the client; second, we used existing libraries and generic methods that were not optimized for this specific task.

We note that the parameterized matrix is generated only once. Once generated, the matrix can be reused in subsequent attacks on obfuscated QUBO problems involving cryptanalysis of a cipher with the same parameters. In this scenario, the entire attack is reduced to the final phase, which can be completed in approximately one hour.

In our experiment, we recovered the entire keystream. As we discussed in Section 5.3, this result occurred because condition (16) was fulfilled. The attack produced an algebraic description of the internal state and the keystream. We verified the correctness of the results by comparing the recovered keystream and the algebraic description of the state analogously to Table 2.

7 Countermeasures to Our Attacks

This section presents three specific methods to mitigate our attacks: changing the coefficient in the Rosenberg substitution, forming linear combinations of Boolean equations, and applying affine transformations to the Boolean variables. These methods are compatible with both representations of optimization problems (QUBO and Ising model). Furthermore, these methods can strengthen the spin reversal and variable permutation techniques.

Our attack consists of two steps: the first involves determining potential permutations, and the second focuses on identifying reversed spins. As described in Section 5.1, the determination of permutations is possible due to the knowledge of the specific coefficients used in constructing the polynomial that defines the QUBO problem. Our countermeasures aim to make the identification of variable permutations impossible–or at least significantly more

challenging–by modifying the constant c that appears in Equation 14.

While even the simplest algebraic attack on the Univium cipher requires 54 variables, illustrating countermeasures with such complexity can be unwieldy. Therefore, to demonstrate our countermeasures, we use a smaller example: a system of three Boolean equations with three variables which represents an optimization task. Despite the simplicity of this example, the concepts presented can be applied to protect against practical attacks. We borrow this specific example from [6] and present further details in Subsection 7.1.

7.1 Transforming a System of Boolean Equations into a QUBO Problem

The following system of equations is given:

$$\begin{cases} f_0 : x_0x_1 + x_2 + 1 \equiv 0 \ (mod\ 2), \\ f_1 : x_1x_2 + x_0 \equiv 0 \ (mod\ 2), \\ f_2 : x_0 + x_1 + x_2 + 1 \equiv 0 \ (mod\ 2). \end{cases} \tag{22}$$

Let us transform system 22 into the QUBO problem. First, note that

$$\begin{cases} f_0 : x_0x_1 + x_2 + 1 = 2k_0, \\ f_1 : x_1x_2 + x_0 = 2k_1, \\ f_2 : x_0 + x_1 + x_2 + 1 = 2k_2, \end{cases} \tag{23}$$

which is equivalent to the system of equations

$$\begin{cases} f_0' : x_0x_1 + x_2 + 1 - 2k_0 = 0, \\ f_1' : x_1x_2 + x_0 - 2k_1 = 0, \\ f_2' : x_0 + x_1 + x_2 + 1 - 2k_2 = 0, \end{cases} \tag{24}$$

Next, linearization is performed:

- $x_3 = x_0x_1$,
- $x_4 = x_1x_2$.

For these two substitutions, the following penalties are obtained:$Pen_1 = 2(x_0x_1 - 2x_3(x_0 + x_1) + 3x_3)$ and $Pen_2 = 2(x_1x_2 - 2x_4(x_1 + x_2) + 3x_4)$. These penalties will be added to the QUBO problem at the end of the transformation. After linearization, the resulting system of equations is:

$$\begin{cases} f'_{lin_0} : x_3 + x_2 + 1 - 2k_0 = 0, \\ f'_{lin_1} : x_4 + x_0 - 2k_1 = 0, \\ f'_{lin_2} : x_0 + x_1 + x_2 + 1 - 2k_2 = 0. \end{cases} \tag{25}$$

The values of k_0, k_1, and k_2 are limited by the maximal value of each of the equations. Because the maximal value of $x_3 + x_2 + 1$ equals 3, it must hold that $2k_0 \leq 3$, and therefore the maximal value of k_0 is 1. Similarly, because the maximal value of $x_4 + x_0$ is 2, it must hold that $2k_1 \leq 2$, and the maximal value of k_1 is 1. The maximal value of $x_0 + x_1 + x_2 + 1$ equals 4. Therefore, $2k_2 \leq 4$, and the maximal value of k_2 is 2. Since the sum of Boolean variables must represent variables k_0, k_1, k_2, one can write $k_0 = x_5$, $k_1 = x_6$, $k_2 = x_7 + x_8$.

Finally, the system of equations is equivalent to:

$$\begin{cases} x_3 + x_2 + 1 - 2x_5 = 0, \\ x_4 + x_0 - 2x_6 = 0, \\ x_0 + x_1 + x_2 + 1 - 2(x_7 + x_8) = 0. \end{cases} \tag{26}$$

However, let us note that the system of equations in Equation 26 is not a Boolean function but a pseudo-Boolean function.

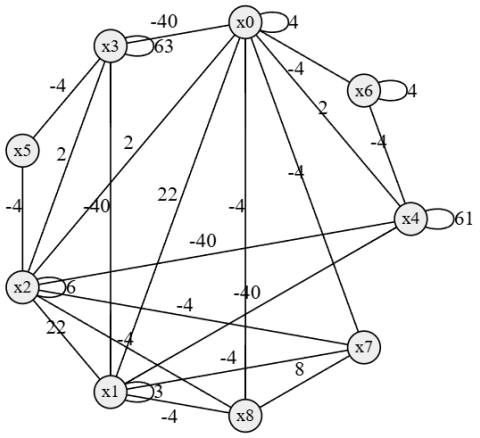

Figure 4: Graph G_1 which represents the QUBO problem obtained from the system of equations in Equation 22.

Finally, the final QUBO problem may be obtained as

$$F_{Pen} = (f'_{lin_0})^2 + (f'_{lin_1})^2 + (f'_{lin_2})^2 + c \cdot Pen - C,$$

where $Pen = Pen_1 + Pen_2$ are penalties obtained during linearization, and $c = 10$ is an arbitrarily chosen constant. Here, $c = 10$ was chosen to ensure that incorrect substitutions are avoided during the function minimization process with a higher probability than in the case of $c = \frac{1}{2}$. Of course, the QUBO problem cannot include the degree 0 term, denoted here by C. Thus, the final QUBO problem is in the form

$$(f'_{lin_0})^2 + (f'_{lin_1})^2 + (f'_{lin_2})^2 + c \cdot Pen,$$

but $-C$ gives the minimal energy of such a QUBO problem.

The final QUBO problem can be expressed by the following quadratic polynomial: $F_{Pen} = 22x_0x_1 + 2x_0x_2 + 22x_1x_2 - 40x_0x_3 - 40x_1x_3 + 2x_2x_3 + 2x_0x_4 - 40x_1x_4 - 40x_2x_4 - 4x_2x_5 - 4x_3x_5 - 4x_0x_6 - 4x_4x_6 - 4x_0x_7 - 4x_1x_7 - 4x_2x_7 - 4x_0x_8 - 4x_1x_8 - 4x_2x_8 + 8x_7x_8 + 4x_0 + 3x_1 + 6x_2 + 63x_3 + 61x_4 + 4x_6$.

The graph structure of this problem is as follows: Figure 4.

The large absolute values of the weights in the given graph make it easy to identify which variable is substituted for a given quadratic monomial. For example, it is evident that there is a loop for x_3 with a weight of 63, and the edges from x_0 to x_3 and from x_1 to x_3 have weights of -40. From this, it is straightforward to conclude that x_3 is the variable used to substitute x_0x_1 during linearization. Similarly, since the loop for x_4 has a weight of 61 and the edges from x_1 to x_4 and from x_2 to x_4 have weights of -40, it is clear that x_4 is the auxiliary variable used to substitute x_1x_2 during linearization.

This same technique was used in our attack on the Trivium cipher family. It allows us to find the path and identify the correct permutation of variables.

7.2 Changing coefficient in Rosenberg substitution

The first possible method to mitigate the attacks we present (particularly to strengthen the variable permutation method) involves reducing the constant c (Equation 14), which is multiplied by every

Rosenberg penalty during the generation of the QUBO problem. Throughout this paper, we assumed that the constant c is large enough in order to enforce correct substitutions during the quantum optimization process. However, a large coefficient c is not formally required.

Using $c = \frac{1}{2}$ is also valid as the minimal solution still yields the correct solution for the original system of equations, and the coefficients remain integers. The only practical drawback of this setting could be the potentially longer time required to obtain the correct solution to the original system of equations, as the penalty for incorrect substitutions would be smaller. Despite this, reducing c appears to be the simplest method to prevent finding a permutation of the original system of equations.

7.3 Linear combination of equations

Since the optimization problems in our case are highly structured, their graph representations exhibit a similarly structured nature. A straightforward approach to modifying the graph structure is to perform simple linear combinations of the equations prior to transforming the problem into QUBO form, resulting in a new system of equations equivalent to the original one. This process produces a modified–and less structured–graph representing the QUBO problem to be solved.

A linear transformation of the given system of equations can be performed as follows. We combine the first and last equations (f_0 and f_2) in Equation 22, as new first equation (f_0) and obtain the following system of equations:

$$\begin{cases} f_0 : x_0x_1 + x_0 + x_1 \equiv 0 \ (mod\ 2), \\ f_1 : x_1x_2 + x_0 \equiv 0 \ (mod\ 2), \\ f_2 : x_0 + x_1 + x_2 + 1 \equiv 0 \ (mod\ 2). \end{cases} \tag{27}$$

We also use the penalty constant $c = \frac{1}{2}$. As a result:

(1) identifying the task becomes more difficult in this case;
(2) due to the altered graph structure, deobfuscating the problem should be significantly harder (or even impossible) under these conditions;
(3) constructed, it is possible to maintain the same number of variables or, in some cases, even reduce the number of variables.

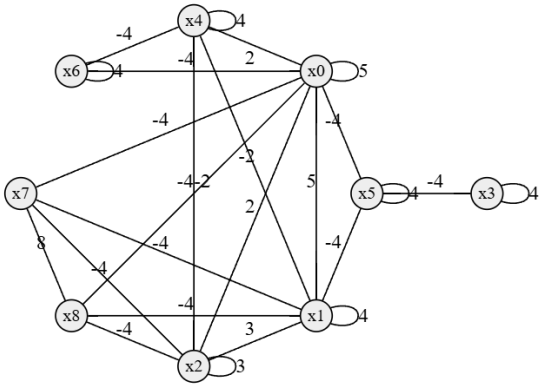

Figure 5: Graph G_2 which represents the QUBO problem obtained from the system of equations in Equation 27.

The graph G_2, shown in Figure 5, differs significantly from the graph G_1, shown in Figure 4 in its structure. Specifically, in G_1, the degree of the node x_2 is 8, whereas in G_2, the degree of the same node is reduced to 6. Additionally, the weights of G_2 do not easily reveal the substitutions performed during linearization. Furthermore, if the linear combination of equations is constructed carefully, it is possible to obtain a QUBO problem with the same or even a smaller number of binary variables.

We will demonstrate this approach using the example presented in Equation (27).

We first add equation f_2 to equation f_0. As $f_{2\ max} = 4$, and $f_{0\ max} = 3$, k_2 will require 2 binary variables, while k_0 requires 1 binary variable. At first glance, it might seem that modifying equation f_0 by adding equation f_2 would result in a new equation f_0 with a maximal value of 7. However, this is not the case because some monomials cancel out. Consequently, the maximal value of the new equation f_0 remains 3, and the number of binary variables required for the new k_0 remains 1.

More generally, we can formulate the following statement: If, for a general system of Boolean equations, one adds equation j to equation i, and

$$\lfloor log_2 f_{i\ max} \rfloor \geq \lfloor log_2 f_{k\ max} \rfloor, \tag{28}$$

where $f_k = f_i + f_j$, such linear combinations do not increase the number of binary variables. However, in some cases, this approach may also reduce the number of additional variables if condition (28) is strict. This scenario occurs when certain monomials cancel out during the addition.

Therefore, if one adds the equation f_0 to the equation f_2, then the new system of equations will be of the form:

$$\begin{cases} f_0 : x_0x_1 + x_2 + 1 \equiv 0 \ (mod\ 2), \\ f_1 : x_1x_2 + x_0 \equiv 0 \ (mod\ 2), \\ f_2 : x_0x_1 + x_0 + x_1 \equiv 0 \ (mod\ 2). \end{cases} \tag{29}$$

In this case, the maximal values are as follows: $f_0 = 3$, $f_1 = 2$, and $f_2 = 3$.

Consequently, $k_0 = x_5$, $k_1 = x_6$, and $k_2 = x_7$, where x_5, x_6, x_7 are binary variables. Since two additional binary variables are required for linearization, the total number of binary variables needed to transform system of equations in Equation 22 into the QUBO form is 9. In contrast, transforming the equivalent system of equations in Equation 31 using the proposed method will require only 8 binary variables. Furthermore, the structure of the graph will also differ in this case. We apply $c = \frac{1}{2}$ to the penalty constant.

The graph representing the resulting QUBO problem is shown in Figure 6.

It is worth noting that after applying a small Rosenberg constant to the penalties and performing linear combinations of equations, additional techniques such as spin reversal transformation and variable transformation can be applied to further protect the original task.

7.4 Applying affine transformation of basic variables in the original Boolean system of equations

We now present the final method which is performed before transforming the system into the QUBO problem. This method can only

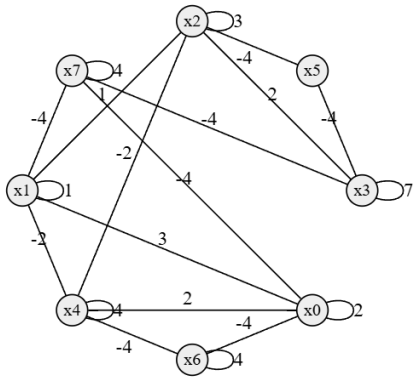

Figure 6: Example permutation of internal state with overlapping quadratic monomials.

be applied when the resulting QUBO problem is derived from a system of Boolean equations. In this approach, every valid affine transformation of variables generates a new system of Boolean equations. This transformed system is highly likely to have a significantly different internal structure. Consequently, analyzing the QUBO problem derived from the transformed system of equations is unlikely to reveal information about the original system of equations.

Assume that the system of equations is given by Equation (22). Let us perform a simple affine transformation on the original variables as follows:

$$\begin{cases} x_0 = y_0, \\ x_1 = y_0 + y_1, \\ x_2 = y_0 + y_1 + y_2. \end{cases} \tag{30}$$

In this case, the original system of Boolean equations undergoes significant changes:

$$\begin{cases} y_0(y_0 + y_1) + y_0 + y_0 + y_1 + 1 \equiv 0 \ (mod\ 2), \\ (y_0 + y_1)(y_0 + y_1 + y_2) + y_0 \equiv 0 \ (mod\ 2), \\ y_0 + y_0 + y_1 + y_0 + y_1 + y_2 + 1 \equiv 0 \ (mod\ 2). \end{cases} \tag{31}$$

Simplifying further, the system of equations can be reduced to the following:

$$\begin{cases} y_0 + y_1 + y_0y_1 + 1 \equiv 0 \ (mod\ 2), \\ y_1 + y_0y_2 + y_1y_2 \equiv 0 \ (mod\ 2), \\ y_0 + y_2 + 1 \equiv 0 \ (mod\ 2). \end{cases} \tag{32}$$

In the case presented above, the system of equations (32) differs significantly from the system of equations in Equation 22. Notably, in Equation 32, three distinct quadratic monomials occur, whereas in Equation 22, only two quadratic monomials exist. Furthermore, as illustrated by the graph in Figure 7, the graph structure of the QUBO problem derived from the system of equations in Equation 31 differs substantially from the graph structure of the QUBO problem obtained by transforming the system of equations in Equation 22.

Additionally, in the above case, one extra Boolean variable is required to construct the QUBO problem. As noted, this is because three quadratic monomials occur instead of two, necessitating one additional variable for linearization. Consequently, the final QUBO problem and its graph structure differ significantly from those obtained from the original system of equations. This makes the problem of identifying and properly interpreting the solution much more challenging than in the original case, where spin reversal transformations and variable substitutions are applied.

It is also worth noting that simple variable permutation (of the variables of the original system if equations) is a special case of the affine transformation presented. However, the affine transformation described here is far more general. If the transformation from the vector $X = (x_0, \ldots, x_{n-1})$ to $Y = (y_0, \ldots, y_{n-1})$ can be represented as a quadratic matrix M of size n, where $det(M) \neq 0$, the transformation is bijective. Therefore, if the solution is found as vector Y, the corresponding solution given by X is unambiguous. Finally, in affine transformations, it is also possible to add the constant value 1 to any linear combination of the variables x .

In conclusion, the countermeasures we have presented are effective in the cryptanalysis of ciphers from the Trivium family. However, when using affine transformations, these methods may result in a new problem with more variables than the original one. Once this step is completed, spin reversal transformations should be applied to further obscure the problem.

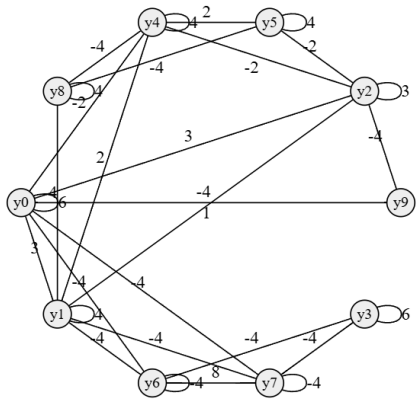

Figure 7: Graph G_4 which represents the QUBO problem obtained from the system (32)**.**

8 Conclusions

This paper presents the first attack on two privacy-preserving methods for quantum optimization: variable permutation and its combination with sign reversal. We demonstrate that even when these methods are applied together, an adversary with access to the obfuscated problem and its obfuscated solution can efficiently recover the original optimization problem using a polynomial-time attack on commodity hardware.

Our attack uses a specific optimization problem (an algebraic attack on the Trivium cipher family) and leverages its structure. While not all optimization problems share this vulnerability, there is no definitive method to classify problems as resistant to our attack.

The existence of our attack poses a significant threat, as clients cannot risk using a compromised method with known weaknesses. This risk is particularly acute for legal compliance, where relying on such methods could be deemed negligent. Although we propose countermeasures to mitigate our attack and two other unbroken

privacy-preserving methods for quantum optimization (which increase the complexity of solving the obfuscated problem) exist, our results underscore the significant challenges in achieving robust privacy for quantum optimization problems.

Ultimately, our paper shows that universally effective privacy-preserving techniques for quantum optimization are still significantly lacking. This limitation poses a fundamental obstacle to the secure deployment of quantum optimization services.

References

[1] Ramin Ayanzadeh, Ahmad Mousavi, Narges Alavisamani, and Moinuddin Qureshi. 2023. Enigma: Privacy-Preserving Execution of QAOA on Untrusted Quantum Computers. arXiv:2311.13546 [quant-ph] https://arxiv.org/abs/2311.13546

[2] Joao Basso, Edward Farhi, Kunal Marwaha, Benjamin Villalonga, and Leo Zhou. 2022. The Quantum Approximate Optimization Algorithm at High Depth for MaxCut on Large-Girth Regular Graphs and the Sherrington-Kirkpatrick Model. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPICS.TQC.2022.7

[3] David Biron, Ofer Biham, Eli Biham, Markus Grassl, and Daniel A Lidar. 1999. Generalized Grover search algorithm for arbitrary initial amplitude distribution. *Lecture notes in computer science* (1999), 140–147.

[4] Michał Borowski, Paweł Gora, Katarzyna Karnas, Mateusz Błajda, Krystian Król, Artur Matyjasek, Damian Burczyk, Miron Szewczyk, and Michał Kutwin. 2020. New Hybrid Quantum Annealing Algorithms for Solving Vehicle Routing Problem. In *Computational Science – ICCS 2020*, Valeria V. Krzhizhanovskaya, Gábor Závodszky, Michael H. Lees, Jack J. Dongarra, Peter M. A. Sloot, Sérgio Brissos, and João Teixeira (Eds.). Springer International Publishing, Cham, 546–561. https://doi.org/10.1007/978-3-030-50433-5_42

[5] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2014. (Leveled) Fully Homomorphic Encryption without Bootstrapping. *ACM Trans. Comput. Theory* 6, 3, Article 13 (July 2014), 36 pages. https://doi.org/10.1145/2633600

[6] Elżbieta Burek, Michał Wroński, Krzysztof Mańk, and Michał Misztal. 2022. Algebraic Attacks on Block Ciphers Using Quantum Annealing. *IEEE Transactions on Emerging Topics in Computing* 10, 2 (2022), 678–689. https://doi.org/10.1109/TETC.2022.3143152

[7] James Clark, Tristan West, Joseph Zammit, Xiaohu Guo, Luke Mason, and Duncan Russell. 2019. Towards Real Time Multi-robot Routing using Quantum Computing Technologies. In *Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region* (Guangzhou, China) *(HPCAsia '19)*. Association for Computing Machinery, New York, NY, USA, 111–119. https://doi.org/10.1145/3293320.3293333

[8] Brazilian National Congress. 2018. General Personal Data Protection Act (LGPD).

[9] National People's Congress. 2016. Cybersecurity Law of the People's Republic of China.

[10] United States Congress. 1974. Family Educational Rights and Privacy Act of 1974.

[11] United States Congress. 1996. Health Insurance Portability and Accountability Act of 1996.

[12] United States Congress. 2001. Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism Act of 2001.

[13] United States Congress. 2002. Federal Information Security Management Act of 2002.

[14] United States Congress. 2010. Dodd–Frank Wall Street Reform and Consumer Protection Act.

[15] United States Congress. 2018. Clarifying Lawful Overseas Use of Data Act.

[16] Christophe De Cannière. 2006. Trivium: A Stream Cipher Construction Inspired by Block Cipher Design Principles. In *Information Security*, Sokratis K. Katsikas, Javier López, Michael Backes, Stefanos Gritzalis, and Bart Preneel (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 171–186.

[17] Łukasz Dzierzkowski. 2024. The generalized method of solving ECDLP using quantum annealing. *arXiv e-prints*, Article arXiv:2410.08725 (Oct. 2024), arXiv:2410.08725 pages. https://doi.org/10.48550/arXiv.2410.08725 arXiv:2410.08725 [cs.CR]

[18] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A Quantum Approximate Optimization Algorithm. arXiv:1411.4028 [quant-ph] https://arxiv.org/abs/1411.4028

[19] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Leo Zhou. 2022. The Quantum Approximate Optimization Algorithm and the Sherrington-Kirkpatrick Model at Infinite Size. *Quantum* 6 (July 2022), 759. https://doi.org/10.22331/q-2022-07-07-759

[20] Sebastian Feld, Christoph Roch, Thomas Gabor, Christian Seidel, Florian Neukart, Isabella Galter, Wolfgang Mauerer, and Claudia Linnhoff-Popien. 2019. A Hybrid Solution Method for the Capacitated Vehicle Routing Problem Using a Quantum Annealer. *Frontiers in ICT* 6 (2019). https://doi.org/10.3389/fict.2019.00013

[21] Craig Gentry, Amit Sahai, and Brent Waters. 2013. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In *Advances in Cryptology – CRYPTO 2013*, Ran Canetti and Juan A. Garay (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 75–92.

[22] Erica Grant, Travis S. Humble, and Benjamin Stump. 2021. Benchmarking Quantum Annealing Controls with Portfolio Optimization. *Phys. Rev. Appl.* 15 (Jan 2021), 014012. Issue 1. https://doi.org/10.1103/PhysRevApplied.15.014012

[23] IBM. 2023. *The hardware and software for the era of quantum utility is here.* https://www.ibm.com/quantum/blog/quantum-roadmap-2033

[24] Rosenberg IG. 1975. REDUCTION OF BIVALENT MAXIMIZATION TO THE QUADRATIC CASE. *CAH. CENTRE ET. RECH. OPERAT.; BELG.; DA. 1975; VOL. 17; NO 1; PP. 71-74; BIBL. 10 REF.* (1975).

[25] Shuxian Jiang, Keith A. Britt, Alexander J. McCaskey, Travis S. Humble, and Sabre Kais. 2018. Quantum Annealing for Prime Factorization. *Scientific Reports* 8, 1 (05 Dec 2018), 17667. https://doi.org/10.1038/s41598-018-36058-z

[26] Tadashi Kadowaki and Hidetoshi Nishimori. 1998. Quantum annealing in the transverse Ising model. *Phys. Rev. E* 58 (Nov 1998), 5355–5363. Issue 5. https://doi.org/10.1103/PhysRevE.58.5355

[27] California legislature. 2018. California Consumer Privacy Act of 2018.

[28] Mateusz Leśniak, Elżbieta Burek, and Michał Wroński. 2024. Unsafe Mechanisms of Bluetooth, $$E_0$$Stream Cipher Cryptanalysis with Quantum Annealing. In *Computational Science – ICCS 2024*, Leonardo Franco, Clélia de Mulatier, Maciej Paszynski, Valeria V. Krzhizhanovskaya, Jack J. Dongarra, and Peter M. A. Sloot (Eds.). Springer Nature Switzerland, Cham, 389–404.

[29] Mateusz Leśniak and Michał Wroński. 2024. Privacy for Quantum Annealing. Attack on Spin Reversal Transformations in the case of cryptanalysis. arXiv:2409.17744 [cs.CR] https://arxiv.org/abs/2409.17744

[30] Andrew Lucas. 2014. Ising formulations of many NP problems. *Frontiers in Physics* 2 (2014). https://doi.org/10.3389/fphy.2014.00005

[31] Anuradha Mahasinghe and Youvin Jayasinghe. 2022. An initial step toward a quantum annealing approach to the discrete logarithm problem. *SECURITY AND PRIVACY* 5, 4 (2022), e234. https://doi.org/10.1002/spy2.234 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/spy2.234

[32] Qurban A. Memon, Mahmoud Al Ahmad, and Michael Pecht. 2024. Quantum Computing: Navigating the Future of Computation, Challenges, and Technological Breakthroughs. *Quantum Reports* 6, 4 (2024), 627–663. https://doi.org/10.3390/quantum6040039

[33] Satoshi Morita and Hidetoshi Nishimori. 2008. Mathematical foundation of quantum annealing. *J. Math. Phys.* 49, 12 (12 2008), 125210. https://doi.org/10.1063/1.2995837 arXiv:https://pubs.aip.org/aip/jmp/article-pdf/doi/10.1063/1.2995837/13869474/125210_1_online.pdf

[34] Samuel Mugel, Carlos Kuchkovsky, Escolástico Sánchez, Samuel Fernández-Lorenzo, Jorge Luis-Hita, Enrique Lizaso, and Román Orús. 2022. Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. *Physical Review Research* 4, 1 (Jan. 2022). https://doi.org/10.1103/physrevresearch.4.013006

[35] Florian Neukart, Gabriele Compostella, Christian Seidel, David von Dollen, Sheir Yarkoni, and Bob Parney. 2017. Traffic Flow Optimization Using a Quantum Annealer. *Frontiers in ICT* 4 (2017). https://doi.org/10.3389/fict.2017.00029

[36] Parliament of Poland. 1997. Kodeks karny (Criminal Code).

[37] Daniel O'Malley and John K. Golden. 2020. Homomorphic Encryption for Quantum Annealing with Spin Reversal Transformations. In *2020 IEEE High Performance Extreme Computing Conference (HPEC)*. 1–6. https://doi.org/10.1109/HPEC43674.2020.9286176

[38] European Parliament. 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance). , 88 pages.

[39] European Parliament. 2019. Cybersecurity Act.

[40] German Parliament. 1871. Strafgesetzbuch (StGB).

[41] Siani Pearson and Azzedine Benameur. 2010. Privacy, Security and Trust Issues Arising from Cloud Computing. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science*. 693–702. https://doi.org/10.1109/CloudCom.2010.66

[42] Ronald L. Rivest and Michael L. Dertouzos. 1978. ON DATA BANKS AND PRIVACY HOMOMORPHISMS. https://api.semanticscholar.org/CorpusID:6905087

[43] Gili Rosenberg, Poya Haghnegahdar, Phil Goddard, Peter Carr, Kesheng Wu, and Marcos López de Prado. 2016. Solving the Optimal Trading Trajectory Problem Using a Quantum Annealer. *IEEE Journal of Selected Topics in Signal Processing* 10, 6 (2016), 1053–1060. https://doi.org/10.1109/JSTSP.2016.2574703

[44] SageMath. [n. d.]. *Why SageMath?* https://www.sagemath.org/library-why.html

[45] D-Wave Systems. 2019. *TechRepublic: D-Wave Announces 5,000-Qubit Fifth Generation Quantum Annealer.* https://www.dwavesys.com/company/newsroom/media-coverage/techrepublic-d-wave-announces-5-000-qubit-fifth-generation-quantum-annealer/

[46] D-Wave Systems. 2022. *Ahead of the Game: D-Wave Delivers Prototype of Next-Generation Advantage2 Annealing Quantum Computer.* https://www.dwavesys.com/company/newsroom/press-release/ahead-of-the-game-d-wave-delivers-prototype-of-next-generation-advantage2-annealing-quantum-computer/

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

[48] Neil C. Thompson, Shuning Ge, and Gabriel F. Manso. 2022. The Importance of (Exponentially More) Computing Power. arXiv:2206.14007 [cs.AR] https://arxiv.org/abs/2206.14007

[49] Yun Tian, Gongliang Chen, and Jianhua Li. 2009. On the Design of Trivium. Cryptology ePrint Archive, Paper 2009/431. https://eprint.iacr.org/2009/431

[50] Davide Venturelli and Alexei Kondratyev. 2019. Reverse quantum annealing approach to portfolio optimization problems. *Quantum Machine Intelligence* 1, 1 (01 May 2019), 17–30. https://doi.org/10.1007/s42484-019-00001-w

[51] Michał Wroński and Łukasz Dzierzkowski. 2024. 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* 24, 7&8 (2024), 0541–0564.

[52] Michał Wroński, Elżbieta Burek, and Mateusz Leśniak. 2024. (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), 1–14. https://doi.org/10.1109/TETC.2024.3474856

[53] Moyang Xie, Yuan Zhang, Sheng Zhong, and Qun Li. 2024. Privacy-Preserving Quantum Annealing for Quadratic Unconstrained Binary Optimization (QUBO) Problems. arXiv:2409.18601 [cs.CR] https://arxiv.org/abs/2409.18601

[54] Sheir Yarkoni, Alex Alekseyenko, Michael Streif, David Von Dollen, Florian Neukart, and Thomas Bäck. 2021. Multi-car paint shop optimization with quantum annealing. In *2021 IEEE International Conference on Quantum Computing and Engineering (QCE).* 35–41. https://doi.org/10.1109/QCE52317.2021.00019

[55] Sheir Yarkoni, Florian Neukart, Eliane Moreno Gomez Tagle, Nicole Magiera, Bharat Mehta, Kunal Hire, Swapnil Narkhede, and Martin Hofmann. 2020. Quantum Shuttle: traffic navigation with Quantum computing. In *Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software* (Virtual, USA) *(APEQS 2020).* Association for Computing Machinery, New York, NY, USA, 22–30. https://doi.org/10.1145/3412451.3428500

[56] Sheir Yarkoni, Elena Raponi, Thomas Bäck, and Sebastian Schmitt. 2022. Quantum annealing for industry applications: introduction and review. *Reports on Progress in Physics* 85, 10 (sep 2022), 104001. https://doi.org/10.1088/1361-6633/ac8c54