(In)security of stream ciphers against quantum annealing attacks on the example of the Grain 128 and Grain 128a ciphers

Michał Wroński[1][0000−0002−8679−9399], Elżbieta Burek[2][0000−0003−2937−0833], and Mateusz Leśniak[1][0009−0001−0092−2975]

[1] NASK National Research Institute, Kolska Str. 12, Warsaw, Poland
{michal.wronski, mateusz.lesniak}@nask.pl
[2] Military University of Technology, Kaliskiego Str. 2, Warsaw, Poland
elzbieta.burek@wat.edu.pl

Abstract. The security level of a cipher is a key parameter. While general-purpose quantum computers significantly threaten modern symmetric ciphers, other quantum approaches like quantum annealing have been less concerning. However, this paper argues that a quantum annealer specifically designed to attack Grain 128 and Grain 128a ciphers could soon be technologically feasible. Such an annealer would require 5,751 (6,751) qubits and 77,496 (94,708) couplers, with a qubit connectivity of 225 (249). Notably, the forthcoming D-Wave Advantage 2 with Zephyr topology will feature over 7,000 qubits and 60,000 couplers and a qubit connectivity 20. This work also shows that modern stream ciphers like Grain 128 and Grain 128a could be vulnerable to quantum annealing attacks. Although the exact complexity of quantum annealing is unknown, heuristic estimates suggest a $\sqrt{N}$ exponential advantage over simulated annealing for problems with N variables. We detail how to transform algebraic attacks on Grain ciphers into the QUBO problem, making our attack potentially more efficient than classical brute-force methods. We also show that applying our attack to rescaled versions of the Grain cipher, namely Grain l and Grain la versions, where l represents the key size, leads to our attack overtaking both brute-force and Grover's attacks for sufficiently large l. This is true, provided that quantum annealing has an exponential advantage over simulated annealing. Specifically, it is sufficient for the time complexity of quantum annealing for problems with N variables to be e^{N^β} for any $\beta < 1$. Finally, given the general nature of our attack method, all new ciphers should be scrutinized for vulnerability to quantum annealing attacks and at least match the AES cipher in terms of security level.

Keywords: stream cipher · Grain · quantum annealing · cryptanalysis

1 Introduction

Applying quantum annealing (QA) in cryptanalysis originated with attempts to perform factorization using this technique. Interestingly, current quantum annealers have significantly more resources—especially working qubits—than general-purpose quantum computers. For instance, the D-Wave Advantage boasts nearly 6,000 working qubits, while, in contrast, the largest general-purpose quantum computer, IBM Osprey, has only 433 working qubits. This disparity suggests that present-day quantum annealers could tackle more extensive cryptanalysis problems than general-purpose quantum computers.

This paper argues that a quantum annealer specifically designed to attack Grain 128 and Grain 128a ciphers could soon be technologically feasible. Such an annealer would require 5,751 (6,751) qubits and 77,496 (94,708) couplers, with a qubit connectivity of 225 (249). Notably, the forthcoming D-Wave Advantage 2 with Zephyr topology will feature over 7,000 qubits and 60,000 couplers and a qubit connectivity 20.

However, until recently, quantum annealing was perceived more as an intriguing novelty than a tool capable of solving genuine challenges. It is worth noting that the complexity of quantum annealing is not precisely determined; however, evidence and heuristic estimates suggest that its time complexity depends exponentially on $\sqrt{N}$ for many problems [20] when the barrier width remains independent of N, where N denotes the number of variables in the optimization problem executed on the quantum annealer.

The applicability of quantum annealing has been examined for several prevalent cryptanalytic challenges, such as:

1. Integer factorization, where transformation to the QUBO problem requires $\sim \frac{n^2}{4}$ variables for an integer with bit length n;
2. The discrete logarithm problem over prime fields, which, when transformed to the QUBO problem, needs $\sim 2n^2$ variables for a prime field with bit length n [24];
3. Various block ciphers, such as transforming the algebraic attack on AES-128 to the QUBO problem, which demands nearly 30,000 variables and up to 60,000 variables for AES-256 [6].

Given this analysis, one might easily conclude that quantum annealing has not been regarded as a formidable threat. However, the narrative shifts dramatically when discussing algebraic attacks on stream ciphers using quantum annealing.

This paper outlines how to execute an algebraic attack on the Grain 128 and Grain 128a ciphers using QA. Our approach aims to transform these attacks into a QUBO problem while minimizing the number of required logical variables.

Our strategy enables retrieving the complete internal state of the Grain 128 and Grain 128a ciphers using a consecutive sequence of 259 bits from the output keystream. If this stream resembles a random sequence, reversing the process can yield the key and IV vector straightforwardly.

Crucially, the derived QUBO problem for the algebraic attack on these ciphers requires only 5,751 logical variables for Grain 128 and 6,751 logical variables for Grain 128a, the latter being an improved version of the Grain 128 cipher. Direct calculations indicate that the attack's time complexity on Grain

128 depends exponentially on $\sqrt{5,751} \approx 75.84$, and for Grain 128a, depends exponentially on $\sqrt{6,751} \approx 82.16$, assuming that in analyzed problem, the energy barriers width is N independent and therefore quantum annealing has a $\sqrt{N}$ exponential advantage over simulated annealing (as well as brute-force attacks). Since these projections rely on heuristic estimates, it is challenging to predict the actual strength of our attack for these specific problems. Our estimates suggest that our present attacks might surpass brute-force methods for Grain ciphers.

We also show that applying our attack to rescaled versions of the Grain cipher (we use here the similar idea as in the case of rescaling Trivium cipher [22]), namely Grain l and Grain la versions, where l represents the key size, leads to our attack overtaking both brute-force (of time complexity equal to 2^l) and Grover's attacks (of time complexity equal to $2^{\frac{l}{2}}$) for sufficiently large l. This is true provided that quantum annealing has any exponential advantage over simulated annealing; specifically, it is sufficient for the time complexity of quantum annealing for a problem with N variables to be e^{N^β} for any $\beta < 1$. It may be a surprising result for readers who know there is no quantum general database-search algorithm faster than Grover's. The hint here is that the algebraic attack, which we use here, uses the strictly algebraic structure of the cipher. The weaker this structure is, the weaker it should be the attack.

Our paper concludes that every cryptographic algorithm should be assessed for its resilience against quantum annealing attacks. Although estimating the computational complexity of quantum annealing is inherently difficult, we believe adopting the same assumptions NIST used in the PQC competition [9] is reasonable. Specifically, for the first security level, a quantum attack on a given algorithm should be no more feasible than a quantum attack on AES-128 [13], which involves approximately 30,000 logical variables. For the third level, it should be comparable to AES-192 (around 45,000 logical variables), and for the fifth level, akin to AES-256 (approximately 60,000 logical variables).

2 Quantum Computing

Quantum computing is currently one of the most promising fields in computer science. The idea of applying quantum mechanics to computation was first introduced in 1980. In 1985, David Deutsch, a researcher from Oxford University, described a quantum Turing machine. A decade later, two groundbreaking algorithms were introduced:

- In 1994, Peter Shor presented an algorithm [21] that enables polynomial-time factorization of numbers, posing a threat to asymmetric cryptography.
- In 1996, Lov Grover introduced an algorithm [14] that allows for searching an unordered N-element set in $O(\sqrt{N})$ steps, effectively halving the effective key size and posing a threat to symmetric cryptography.

In quantum computing, the basic unit of information is a qubit, which can be represented as a unit vector in the $\mathbb{C}^2$ space using Dirac notation:

$$|q\rangle = \omega_0 |0\rangle + \omega_1 |1\rangle . \tag{1}$$

The coefficients ω_i are called amplitudes and satisfy the condition given by Equation (2):

$$|\omega_0|^2+|\omega_1|^2=1 \quad (\omega_0,\omega_1\in\mathbb{C}). \tag{2}$$

When measuring the value of a qubit, the outcome will be determined probabilistically. The measurement will yield the base vector $|i\rangle$ with a probability of $|\omega_i|^2$.

Computation in a quantum computer is performed using systems of quantum gates, which can be represented as unitary matrices with complex elements. Due to the properties of these matrices, operations performed with quantum gates are reversible.

Building quantum computers faces several challenges, the most significant of which is decoherence. Qubits are highly susceptible to external factors, even minor ones, which can introduce errors in computations. Scaling quantum registers exacerbates this issue, necessitating the application of error correction techniques. Current solutions require a large number of qubits for error correction, leaving fewer for computational tasks.

As of 2023, IBM's largest quantum processor, Osprey, has 433 qubits [10], and the construction of a 1,121-qubit [10] processor named Condor is planned.

An alternative to standard quantum calculations is adiabatic quantum calculations, which are based on the slow evolution of a dynamic system that evolves in accordance with the external and internal forces acting on it. The forces acting on this system can be characterized by a time-varying Hamiltonian, which, in mathematical terms, is a Hermitian matrix. The Hamiltonian operator makes it possible to determine the observable energy of a quantum system at any moment, which corresponds to the eigenvalue of the eigenstate, i.e., the state of the system. In the set of all possible energy values of a given system, there is a ground state, i.e., a state with minimum energy. The remaining states are called excited states. The concept of energy gap is defined as the difference between the eigenvalues of the ground state and the first excited state at any moment of the system's evolution. According to the adiabatic theorem, if the system is in the ground state at the beginning of evolution, the minimum energy gap is greater than zero at any moment, and the process evolves slowly, the quantum system will remain in the ground state. The speed of the evolution process depends on the width of the energy gap; the smaller the gap, the slower the process must be executed.

A practical implementation of the theory of adiabatic quantum computing is quantum annealing, which is used to solve optimization problems. Suppose the quantum system is in the ground state at the end of the evolution process. In that case, the eigenstate of this system corresponds to the optimal solution of a given optimization problem.

Currently, quantum annealers offer a more practical alternative to general-purpose quantum computers, primarily because of their size. The company D-Wave provides access to computers with almost 6,000 qubits. Hybrid solutions have also been developed, where the quantum processor acts as an extension of a classical computer. These hybrid systems can handle even a million variables.

2.1 QUBO problem

Quantum annealing can be used to solve optimization problems in the form of QUBO (Quadratic Unconstrained Binary Optimization). This problem can be formulated as optimizing an expression given by Equation (3)

$$\min_{x \in \{0,1\}^n} x^T Q x, \tag{3}$$

where the vector x contains n binary variables, and the $n \times n$ matrix Q is an upper-triangular matrix containing real values [11]. The elements on the main diagonal correspond to linear monomials, while the elements above the main diagonal are the coefficients of the respective quadratic monomials. This structure of matrix Q allows for the minimization of the function given by

$$f(x) = \sum_i q_{i,i} x_i + \sum_{i<j} q_{i,j} x_i x_j. \tag{4}$$

To formulate a QUBO problem, the cipher under attack must first be represented by a system of polynomial equations over a finite field. The degree and form of the polynomials depend on the structure of the attacked cipher. Let us assume that a particular cipher is represented by the system of equations given in Equation (5) over $GF(2)$.

$$\begin{cases} f_0(x_0, \ldots, x_{n-1}) \equiv 0 \pmod{2}, \\ f_1(x_0, \ldots, x_{n-1}) \equiv 0 \pmod{2}, \\ \vdots \\ f_{t-1}(x_0, \ldots, x_{n-1}) \equiv 0 \pmod{2}. \end{cases} \tag{5}$$

This system can be transformed into a QUBO problem as outlined in [6]. To achieve this transformation, the following steps should be taken:

1. Each equation f_i should be transformed into an equation f'_i with binary variables and integer coefficients: $f'_i = f_i - 2k_i$, where k_i is an integer satisfying
$$k_i \leq \left\lfloor \frac{f_i^{max}}{2} \right\rfloor,$$
where f_i^{max} is the maximal value of the polynomial f_i when all its binary variables are set to one.
2. Each equation f'_i should be linearized to obtain f'_{lin_i} as well as the penalty Pen_i during the linearization. The variables k_i should be replaced with binary variables $x_0, x_1, \ldots, x_{bl(k_i^{max})-1}$:
$$k_i = \sum_{j=0}^{bl(k_i^{max})-2} 2^j x_j + (k_i^{max} - 2^{bl(k_i^{max})-1} + 1) \cdot x_{bl(k_i^{max})-1},$$
where $bl(y)$ is the bit-length of integer y and $k_i^{max} = \lfloor \frac{f_i^{max}}{2} \rfloor$.

3. The polynomial $F'_{\text{Pen}} = \sum_{i=0}^{t-1} (f'_{\text{lin}_i})^2 + c \cdot Pen$, where $Pen = \sum_{i=0}^{t-1} Pen_i$ and c is a constant to prevent achieving the minimum energy for incorrect solutions, should be determined.
4. Finally, the polynomial $F_{\text{Pen}} = F'_{\text{Pen}} - C$ is determined, where C is a constant term in F'_{Pen} corresponding to the minimum energy of the function.

Linearizing Boolean functions is an NP-hard task. However, the solution presented in [5] is sufficient for the above transformation. In each equation, the product of two variables is replaced with a new variable:

$$x_i x_j \to x_k,$$

and the penalty corresponding to this substitution is given by:

$$2(x_i x_j - 2x_k(x_i + x_j) + 3x_k).$$

The number of new variables depends on the order in which substitutions are made. A single substitution can be used in multiple monomials across different equations. Among various options, the most convenient approach is to prioritize substitutions based on their frequency of occurrence.

3 Grain ciphers family

The Grain cipher was first introduced in 2004 as a part of the eSTREAM project, aiming to identify optimal encryption algorithms based on specific criteria [17]. Created by Martin Hell, Thomas Johansson, and Willi Meier, the Grain cipher became a finalist in the Profile 2 category. It is designed with hardware efficiency in mind, focusing on limited resources like storage, gate count, and power consumption. Originally, it utilized an 80-bit key and a 64-bit initialization vector, relying on two 80-bit registers: the Linear Feedback Shift Register (LFSR) and the Non-Linear Feedback Shift Register (NFSR).

In 2006, an attack by Berbein, Bilbert, and Maximov [3] was presented, which required the following:

– 2^{43} operations;
– 2^{43} bits of memory;
– 2^{38} bits of keystream.

3.1 Grain 128

A new version called Grain 128 was introduced in response to the vulnerabilities exposed. It employs a 128-bit key and a 96-bit initialization vector while retaining the benefits of the original design [16]. Figure 1 illustrates the algorithm.

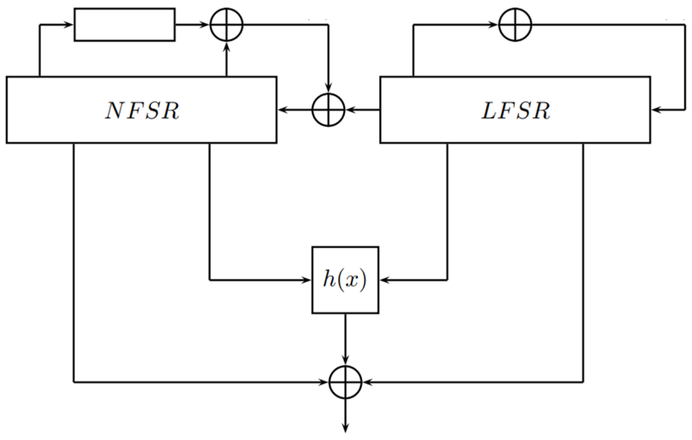

Fig. 1: Grain 128 cipher scheme [16]

The internal state consists of two 128-bit registers:

– LFSR with internal state $(s_i, s_{i+1}, \ldots, s_{i+127})$ and feedback function:

$$s_{i+128} = s_i + s_{i+7} + s_{i+38} + s_{i+70} + s_{i+81} + s_{i+96};$$

– NFSR with internal state $(b_i, b_{i+1}, \ldots, b_{i+127})$ and feedback function:

$$\begin{gathered} b_{i+128} = s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91}b_{i+96} + \\ + b_{i+3}b_{i+67} + b_{i+11}b_{i+13} + b_{i+17}b_{i+18} \\ + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + b_{i+68}b_{i+84}. \end{gathered}$$

From the 256-bit internal state, 9 bits are fed into the function $h(x)$ as defined by:

$$h(x) = x_0x_1 + x_2x_3 + x_4x_5 + x_6x_7 + x_0x_4x_8,$$

with x_k representing specific bits from the internal state:

$$b_{i+12}, s_{i+8}, s_{i+13}, s_{i+20}, b_{i+95}, s_{i+42}, s_{i+60}, s_{i+79}, s_{i+95}.$$

The algorithm's output bit is calculated as follows:

$$y_i = \sum_{j\in\mathcal{A}} b_{i+j} + h(x) + s_{i+93}, \tag{6}$$

where $\mathcal{A} = \{2, 15, 36, 45, 64, 73, 89\}$. The key and the IV vector need to be provided to initiate the algorithm's operation. The NFSR is completely filled with the key bits. Using the IV vector, the first 96 bits of the LFSR state are filled, while the remaining bits are set to 1. Afterward, the entire system is clocked 256 times, during which the generator output is xored to the inputs of each register.

3.2 Grain 128a

An attack by Itai Dinur and Adi Shamir in 2011 [12] led to further modifications, resulting in the Grain 128a version [1]. This variant keeps the key and initialization vector sizes but introduces several changes:

- The NFSR feedback function has been modified as follows:

$$\begin{aligned} b_{i+128} = s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} \\ +b_{i+3}b_{i+67} + b_{i+11}b_{i+13} + b_{i+17}b_{i+18} \\ +b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} \\ +b_{i+68}b_{i+84} + b_{i+88}b_{i+92}b_{i+93}b_{i+95} \\ +b_{i+22}b_{i+24}b_{i+25} + b_{i+70}b_{i+78}b_{i+82}; \end{aligned}$$

- The output function is adjusted, employing the previously mentioned output function as a pre-output function, and the output bit is determined according to the following equation:
$$z_i = y_{64+2i},$$
  for mode with authentication or according to the equation:
$$z_i = y_i,$$
  for mode without authentication. In this paper, we use only the mode of the Grain 128a without authentication.
- The initialization method has also been modified. The internal state of the LFSR is filled with the IV vector, along with 31 bits set to 1 and one bit set to 0. The pre-output is xored to the inputs of each register during each initial step.

3.3 Cryptanalysis

Several methods have been developed to analyze and attack the Grain ciphers. One of the most significant contributions in this field was made in 2018 by Todo et al. [23]. This research demonstrated an attack with a complexity lower than required for an exhaustive search. Specifically, their Fast Correlation Attack could break the Grain 128 cipher with a time complexity of $O(2^{114.4})$ and a memory complexity of $O(2^{112.8})$. For the Grain 128a variant, the attack's time and memory complexities are $O(2^{115.4})$ and $O(2^{113.8})$, respectively.

Prior to this, Dinur and Shamir had presented another attack [12]. This method exploits the low algebraic degree of the non-linear functions used in the Grain 128 cipher, a factor considered in the design of later versions. Their attack enables key recovery with a complexity of $O(2^{103})$, assuming that the key belongs to a subset of 2^{118} potential keys.

These contributions have significantly influenced the understanding of the Grain cipher's security landscape, prompting the development of more secure versions and highlighting the need for ongoing research in this area.

Quantum attacks for gate-based quantum computers have also been considered in the case of the Grain family of ciphers [2], [18].

It is worth noting that quantum circuits implementing Grover's algorithm have been studied for quantum attacks on Grain v1 (with an 80-bit key) and Grain 128a in authentication mode [2]. However, both circuits require enormous resources, rendering the attacks impractical. From the current perspective, Grover's attack on Grain ciphers seems unlikely to be made more efficient.

Table 1 below presents the resource requirements for implementing Grover's circuit for Grain v1 and Grain 128a in authentication mode.

Cipher	#Clifford gates	#T gates	Gate Cost (G)	T-depth	Full depth (D)	#qubits
Grain 128a	$1.902 \cdot 2^{81}$	$1.546 \cdot 2^{81}$	$1.724 \cdot 2^{82}$	$1.737 \cdot 2^{80}$	$1.820 \cdot 2^{80}$	523
Grain v1	$1.623 \cdot 2^{57}$	$1.317 \cdot 2^{57}$	$1.470 \cdot 2^{58}$	$1.486 \cdot 2^{56}$	$1.795 \cdot 2^{55}$	347

Table 1: Resource estimations for Grover's algorithm with $\lfloor \frac{\pi}{4} 2^{\frac{k}{2}} \rfloor$ oracle iterations [2].

An adaptation of the HHL quantum algorithm [15] to algebraic attacks on Grain 128 and Grain 128a ciphers was described in [18]. In this method, similar to the method we described in this paper, equations based on the algebraic structure of the cipher are obtained and then transformed into an appropriate form. Moreover, the time complexity of the attack is $O\left(2^{21} N^{3.5} \kappa^2 e^{\epsilon} / \epsilon^{0.5}\right)$ for Grain 128 and $O\left(2^{21.5} N^{3.5} \kappa^2 e^{\epsilon} / \epsilon^{0.5}\right)$ for Grain 128a, where κ is the condition number of the matrix for the corresponding linear system and ϵ is a given error bound. It is worth noting that estimating how large κ is is challenging. Experiments on toy examples in [18] showed that κ depends on the keystream and can range from relatively small (the smallest obtained value was $2^{5.9932}$) to very large (about 2^{50} in that toy example). Additionally, the quantum resources required for the practical implementation of this attack have not been estimated. These factors indicate that, although the attack presented in [18] is of theoretical interest, its practical implementation and impact may be quite limited.

4 Transformation of the Grain 128 and Grain 128a ciphers to the QUBO problem

This section describes the transformation of the Grain 128 and Grain 128a ciphers to the QUBO problem, a crucial part of the algebraic attack on this cipher using quantum annealing.

4.1 Probability of identifying the correct inner state of the stream cipher

The assessment of an attack's efficacy involves understanding the odds of correctly identifying the cipher's internal state and, consequently, the secret key. Let k denote the number of consecutive output bits that we assume to be known or recoverable for this analysis.

Initially, the output sequence generated by the stream cipher is treated as a random sequence. For a cipher internal state comprising n bits and a known sequence of k consecutive bits, the mean number of inner states that could produce the same sequence is expected to be 1. We also posit that at least one correct inner state (the state responsible for generating the sequence) should exist for a given sequence. The challenge is to determine the possibility of other such states existing. The key points of this analysis are as follows:

1. At least one inner state generates the sequence α, composed of the first k output bits.
2. The cipher's output is assumed to behave like a random number generator.
3. The probability that a specific inner state S will generate the sequence α is 2^{-k}.
4. There are $2^n - 1$ trials (inner states) to consider, as we know at least one correct inner state exists.

We employ a binomial distribution model for this analysis, formulated as:

$$B(N,p) = \sum_{i=0}^{N} \binom{N}{i} p^i (1-p)^{N-i}. \tag{7}$$

For $N = 2^n - 1$ and $p = 2^{-k}$, it translates to:

$$B(2^n-1, 2^{-k}) = \sum_{i=0}^{2^n-1} \binom{2^n-1}{i} (2^{-k})^i (1-2^{-k})^{2^n-1-i}. \tag{8}$$

The average number of 'successes' (inner states generating sequence α) is $\mu = 1 + N \cdot p = 1 + \frac{2^n-1}{2^k}$.

To find the probability of having no additional inner states generating the same sequence α, we calculate the probability for 0 successes:

$$B(N,p,0) = (1-2^{-k})^{2^n-1} \approx e^{-2^{n-k}}, \tag{9}$$

where $e^{-2^{n-k}}$ is a good approximation given that 2^k and 2^n are generally large numbers.

We also want to estimate the probability that, found inner state, producing the correct k-bit output α is indeed the state used to generate this sequence. Let this probability be denoted as $C(N,p)$:

$$C(N,p) = \sum_{i=0}^{N} \binom{N}{i} \frac{1}{i+1} p^i (1-p)^{N-i}. \tag{10}$$

Through the application of generating function properties, this simplifies to:

$$\frac{1-(1-p)^{N+1}}{(N+1)p} \approx \frac{1-e^{-2^{n-k}}}{2^{n-k}}. \tag{11}$$

The data presented in Table 2 suggests that the absolute sizes of n and k are not as crucial as their difference $n-k$. A smaller $n-k$ difference increases the likelihood of identifying the correct inner state.

$n-k$	μ	$B(N,p,0)$	$C(N,p)$
3	9	0.00034	0.12496
2	5	0.01832	0.24542
1	3	0.13534	0.43233
0	2	0.36788	0.63212
-1	1.5	0.60653	0.78694
-2	1.25	0.77880	0.88480
-3	1.125	0.88250	0.94002
-4	1.0625	0.93941	0.96939
-5	1.03125	0.96923	0.98454
-6	1.015625	0.98450	0.99223
-7	1.007813	0.99222	0.99610
-8	1.003906	0.99610	0.99805
-9	1.001953	0.99805	0.99902
-10	1.000977	0.99902	0.99951

Table 2: Values of $\mu, B(N,p,0)$, and $C(N,p)$ for distinct values of $n-k$.

In the context of transforming algebraic attacks on Grain 128 and Grain 128a ciphers into the QUBO problem, it is worth noting that when $k=n$, the probability of the inner state found via quantum annealing being the correct one is over 63%. This level of likelihood is considered practically significant for the attack scenario. However, to obtain a larger probability (around 94%) of obtaining the proper internal state of the cipher, we assume that $k=n+3$ output keystream bits are known - in the case of both Grain 128 and Grain 128a. Therefore, it is equal to 259.

4.2 Generating a system of equations for the Grain cipher

According to the principles of algebraic attacks, the initial k bits of the output keystream are known. We aim to determine the initial state of each NFSR and LFSR internal register after initialization. We can backtrack through the initialization steps to recover the key based on the recovered initial state.

Let binary variables represent the initial states. The shift registers are regularly clocked in both the Grain 128 and Grain 128a ciphers and a single bit of the output keystream is generated per clock cycle. Hence, an equation over the binary field is generated for each known output bit y_i, where $i = \overline{0, k-1}$. During the shift register updates, each generated bit b_{i+128} and s_{i+128} is defined by the characteristic polynomials of these registers and is a function of the unknown binary variables representing the initial state. To avoid increasing the degree of the equations for the output bits, we generate an equation for each bit and represent these bits by successive unknown binary variables. In summary, a system of three equations is generated for each known output bit y_i in the following forms:

$$\begin{cases} s_i + s_{i+7} + s_{i+38} + s_{i+70} + s_{i+81} + s_{i+96} + s_{i+128} = 0, \\ s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} + b_{i+128} + b_{i+3}b_{i+67} + b_{i+11}b_{i+13} + \\ +b_{i+17}b_{i+18} + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + b_{i+68}b_{i+84} = 0, \\ y_i + b_{i+2} + b_{i+15} + b_{i+36} + b_{i+45} + b_{i+64} + b_{i+73} + b_{i+89} + s_{i+93} + \\ +b_{i+12}s_{i+8} + s_{i+13}s_{i+20} + b_{i+95}s_{i+42} + s_{i+60}s_{i+79} + b_{i+12}b_{i+95}s_{i+95} = 0, \end{cases} \tag{12}$$

for Grain 128, and

$$\begin{cases} s_i + s_{i+7} + s_{i+38} + s_{i+70} + s_{i+81} + s_{i+96} + s_{i+128} = 0, \\ s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} + b_{i+128} + b_{i+3}b_{i+67} + \\ +b_{i+11}b_{i+13} + b_{i+17}b_{i+18} + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + \\ +b_{i+68}b_{i+84} + b_{i+22}b_{i+24}b_{i+24} + b_{i+88}b_{i+92}b_{i+93}b_{i+95} = 0, \\ y_i + b_{i+2} + b_{i+15} + b_{i+36} + b_{i+45} + b_{i+64} + b_{i+73} + b_{i+89} + s_{i+93} + \\ +b_{i+12}s_{i+8} + s_{i+13}s_{i+20} + b_{i+95}s_{i+42} + s_{i+60}s_{i+79} + b_{i+12}b_{i+95}s_{i+94} = 0, \end{cases} \tag{13}$$

for Grain 128a.

Consider the following example, presenting the transformation of the equations system for one cycle of the Grain 128 cipher to the QUBO problem. Let us denote the initial state of the LFSR register using the variables $x_0, \ldots, x_{127}$, and let us denote the initial state of the NFSR register using the variables $x_{128}, \ldots, x_{255}$. In this example, let us consider the equations for cycle $i = 0$ and assume that the corresponding keystream bit is $y_0 = 1$. The registers are updated in each cycle of the Grain 128 cipher so that we will represent bits s_{128} and b_{128} from one update as x_{256} and x_{257}, respectively. According to these assumptions, we obtain a system of three equations (14) over the field $GF(2)$.

$$\begin{cases} f_0 = x_0 + x_7 + x_{38} + x_{70} + x_{81} + x_{96} + x_{256} \equiv 0 \pmod 2, \\ f_1 = x_0 + x_{128} + x_{154} + x_{184} + x_{219} + x_{224} + x_{257} + x_{131}x_{195} + x_{139}x_{141} + \\ +x_{145}x_{146} + x_{155}x_{187} + x_{168}x_{176} + x_{189}x_{193} + x_{196}x_{212} \equiv 0 \pmod 2, \\ f_2 = 1 + x_{130} + x_{143} + x_{164} + x_{173} + x_{192} + x_{201} + x_{217} + x_{93} + \\ +x_{140}x_8 + x_{13}x_{20} + x_{223}x_{42} + x_{60}x_{79} + x_{140}x_{223}x_{95} \equiv 0 \pmod 2, \end{cases} \tag{14}$$

Each equation of the obtained system is presented as the difference of the corresponding equation with binary variables and integer coefficients and multiple of the number 2. The obtained system has the form of equation (15), where k_0,

k_1 and k_2 are integers.

$$\begin{cases} f'_0 = x_0 + x_7 + x_{38} + x_{70} + x_{81} + x_{96} + x_{256} - 2k_0, \\ f'_1 = x_0 + x_{128} + x_{154} + x_{184} + x_{219} + x_{224} + x_{257} + x_{131}x_{195} + x_{139}x_{141} + \\ +x_{145}x_{146} + x_{155}x_{187} + x_{168}x_{176} + x_{189}x_{193} + x_{196}x_{212} - 2k_1, \\ f'_2 = 1 + x_{130} + x_{143} + x_{164} + x_{173} + x_{192} + x_{201} + x_{217} + x_{93} + \\ +x_{140}x_8 + x_{13}x_{20} + x_{223}x_{42} + x_{60}x_{79} + x_{140}x_{223}x_{95} - 2k_2, \end{cases} \tag{15}$$

Table 3 presents the values of variables enabling the representation of multiples k_i using subsequent binary variables and the polynomials obtained for the k_i variables.

i	f_i^{max}	k_i^{max}	$bl(k_i^{max})$	k_i
0	7	3	2	$x_{258} + 2x_{259}$
1	14	7	3	$x_{260} + 2x_{261} + 4x_{262}$
2	14	7	3	$x_{263} + 2x_{264} + 4x_{265}$

Table 3: Representing the value of the multiple k_i using binary variables.

In the system of equations (15), there is a linear equation f'_0 and two non-linear equations f'_1 and f'_2, of the second and third degree, respectively. Therefore, the process of linearization of the system (15) consists of substituting a new binary variable for each quadratic monomial in equations f'_1 and f'_2 and determining the appropriate penalty. The third-degree monomial in equation f'_2 is linearized by two substitutions. The substitutions made, and the appropriate penalties assigned are presented in Table 4. Finally, we obtain a system of linear equations of the form (16):

$$\begin{cases} f'_{\mathrm{lin}_0} = x_0 + x_7 + x_{38} + x_{70} + x_{81} + x_{96} + x_{256} - 2x_{258} - 4x_{259}, \\ f'_{\mathrm{lin}_1} = x_0 + x_{128} + x_{154} + x_{184} + x_{219} + x_{224} + x_{257} + x_{266} + x_{267} + \\ +x_{268} + x_{269} + x_{270} + x_{271} + x_{272} - 2x_{260} - 4x_{261} - 8x_{262}, \\ f'_{\mathrm{lin}_2} = 1 + x_{130} + x_{143} + x_{164} + x_{173} + x_{192} + x_{201} + x_{217} + x_{93} + \\ +x_{273} + x_{274} + x_{275} + x_{276} + x_{278} - 2x_{263} - 4x_{264} - 8x_{265}, \end{cases} \tag{16}$$

and the following penalties for individual equations:
$Pen_0 = 0$,
$Pen_1 = 2x_{131}x_{195} - 4x_{266}x_{131} - 4x_{266}x_{195} + 6x_{266} + 2x_{139}x_{141} - 4x_{267}x_{139} - 4x_{267}x_{141} + 6x_{267} + 2x_{145}x_{146} - 4x_{268}x_{145} - 4x_{268}x_{146} + 6x_{268} + 2x_{155}x_{187} - 4x_{269}x_{155} - 4x_{269}x_{187} + 6x_{269} + 2x_{168}x_{176} - 4x_{270}x_{168} - 4x_{270}x_{176} + 6x_{270} + 2x_{189}x_{193} - 4x_{271}x_{189} - 4x_{271}x_{193} + 6x_{271} + 2x_{196}x_{212} - 4x_{272}x_{196} - 4x_{272}x_{212} + 6x_{272}$,
$Pen_2 = 2x_{140}x_8 - 4x_{273}x_{140} - 4x_{273}x_8 + 6x_{273} + 2x_{13}x_{20} - 4x_{274}x_{13} - 4x_{274}x_{20} + 6x_{274} + 2x_{223}x_{42} - 4x_{275}x_{223} - 4x_{275}x_{42} + 6x_{275} + 2x_{60}x_{79} - 4x_{276}x_{60} - 4x_{276}x_{79} + 6x_{276} + 2x_{140}x_{223} - 4x_{277}x_{140} - 4x_{277}x_{223} + 6x_{277} + 2x_{277}x_{95} - 4x_{278}x_{227} -$

f_i	quadratic monomials of the f_i	new variable for substitution	penalty for substitution
f_1	$x_{131}x_{195}$	x_{266}	$2x_{131}x_{195} - 4x_{266}x_{131} - 4x_{266}x_{195} + 6x_{266}$
	$x_{139}x_{141}$	x_{267}	$2x_{139}x_{141} - 4x_{267}x_{139} - 4x_{267}x_{141} + 6x_{267}$
	$x_{145}x_{146}$	x_{268}	$2x_{145}x_{146} - 4x_{268}x_{145} - 4x_{268}x_{146} + 6x_{268}$
	$x_{155}x_{187}$	x_{269}	$2x_{155}x_{187} - 4x_{269}x_{155} - 4x_{269}x_{187} + 6x_{269}$
	$x_{168}x_{176}$	x_{270}	$2x_{168}x_{176} - 4x_{270}x_{168} - 4x_{270}x_{176} + 6x_{270}$
	$x_{189}x_{193}$	x_{271}	$2x_{189}x_{193} - 4x_{271}x_{189} - 4x_{271}x_{193} + 6x_{271}$
	$x_{196}x_{212}$	x_{272}	$2x_{196}x_{212} - 4x_{272}x_{196} - 4x_{272}x_{212} + 6x_{272}$
f_2	$x_{140}x_8$	x_{273}	$2x_{140}x_8 - 4x_{273}x_{140} - 4x_{273}x_8 + 6x_{273}$
	$x_{13}x_{20}$	x_{274}	$2x_{13}x_{20} - 4x_{274}x_{13} - 4x_{274}x_{20} + 6x_{274}$
	$x_{223}x_{42}$	x_{275}	$2x_{223}x_{42} - 4x_{275}x_{223} - 4x_{275}x_{42} + 6x_{275}$
	$x_{60}x_{79}$	x_{276}	$2x_{60}x_{79} - 4x_{276}x_{60} - 4x_{276}x_{79} + 6x_{276}$
	$x_{140}x_{223}$	x_{277}	$2x_{140}x_{223} - 4x_{277}x_{140} - 4x_{277}x_{223} + 6x_{277}$
	$x_{277}x_{95}$	x_{278}	$2x_{277}x_{95} - 4x_{278}x_{227} - 4x_{278}x_{95} + 6x_{278}$

Table 4: Substitutions made in the process of linearization of the example system.

$4x_{278}x_{95} + 6x_{278}$.
Since the f'_0 equation was linear, its penalty after the linearization process is 0.

Assuming $k = 259$, we obtain a system of polynomial equations comprising 777 equations and 774 binary variables for each of the Grain 128 and Grain 128a ciphers.

The equations are then linearized, and the multiples of k_i are represented using additional binary variables, as shown in the earlier example.

Finally, the QUBO problem for the Grain 128 cipher consists of 6,213 binary variables, while the Grain 128a cipher consists of 8,285 binary variables.

4.3 The first improvement

The last section described a basic attack method on the Grain-128 and Grain-128a.

This section describes the first improvement of our attack. A straightforward method to construct an equation for the algebraic attack involves computing, for each bit of the output keystream, the update functions for y_i, b_{i+128}, and s_{i+128}. However, we will demonstrate that if we have k consecutive bits of the output keystream, then we need to create k equations for y_i, one for each $i = \overline{1,k}$. For register b, we need to create only $k - 33$ updating equations, and for register s, we need to create only $k - 33$ updating equations for Grain 128 and only $k - 34$ updating equations for Grain 128a. It is worth noting that one does require k equations for the output bit y_i. For register b, we use up to b_{i+95} since it's the highest index appearing in y_i, and similarly, for register s, we use up to s_{i+95} for Grain 128 cipher and up to s_{i+94} for Grain 128a cipher.

This simple observation allows us to reduce the number of necessary logic variables estimated in the previous section from $6,213$ to $5,751$ for Grain 128 cipher and from $8,285$ to $7,623$ for Grain 128a cipher, thereby decreasing the attack's computational complexity.

4.4 The second improvement

The second improvement involves a more meticulous examination of the equations describing the Grain ciphers. While the first improvement applies to both Grain 128 and Grain 128a, this second improvement is specific to Grain 128a.

We will examine the update function of register b. The insight here is straightforward: if in the update function for b_{i+128} there is a monomial composed of $b_{i+l}b_{i+m}$ for $l < m$, verify that there is not another distinct monomial consisting of $b_{i+r}b_{i+s}$ where $l < r < s$. It's important to note that if $t = m - l = s - r$, then during the linearization of $b_{i+r}b_{i+s}$ and replacing $b_{i+r}b_{i+s}$ with a new variable v, after $u = r - l$ steps, during the linearization of $b_{i+l}b_{i+m}$, there's no need to use a new variable w as the linearization was done u steps earlier using variable v. Consider the following example. In the update equation for b_{i+128}, the monomials that appear are:

- degree 2 monomials: $b_{i+11}b_{i+13}$ and $b_{i+61}b_{i+65}$;
- degree 4 monomial $b_{i+88}b_{i+92}b_{i+93}b_{i+95}$.

Note that for $b_{i+88}b_{i+92}$, the difference between the indices is $t = 4$, and for $b_{i+93}b_{i+95}$, the difference is $t = 2$. During linearization, the term $b_{i+88}b_{i+92}$ is replaced by a new variable v_1, and $b_{i+93}b_{i+95}$ by v_2.

Similarly, for $b_{i+11}b_{i+13}$, the difference between indices is $t = 2$, and for $b_{i+61}b_{i+65}$, the difference is $t = 4$.

This implies that while variables v_1 and v_2 will not be reused during the linearization in the update equation for b_{i+128}, v_1 will be reused $u = 27$ steps later in the update equation for $b_{(i+27)+128} = b_{i+155}$, because $b_{(i+27)+61}b_{(i+27)+65} = b_{i+88}b_{i+92}$. Similarly, v_2 will be reused $u = 82$ steps later, because $b_{(i+82)+11}b_{(i+82)+13} = b_{i+93}b_{i+95}$.

This observation enables us to reduce the total number of necessary variables by almost 900 and obtain a QUBO problem with 6,751 variables.

5 Computational Complexity of the Attack

In general, precisely estimating the time complexity of finding an optimal solution using quantum annealing is challenging. As discussed in [19], the problem complexity primarily depends on the number of logical variables.

The significance of the number of logical variables is evident. Typically, the smaller this number, the lower the time complexity of the problem. Additionally, there is a higher probability that such a problem will be successfully embedded into a real quantum annealer.

Moreover, estimating the time complexity of quantum annealing often depends solely on the number of logical variables. In [20], the time complexity of quantum annealing is heuristically analyzed. Specifically, it is assumed that the probability of quantum tunneling is represented by $e^{-\frac{w\sqrt{\Delta}}{\Gamma}}$, where Δ denotes the barrier height (of order N), w represents the barrier width, and Γ signifies the kinetic energy or the strength of quantum fluctuation. For many problems, the barrier width may be independent of N, resulting in a quantum tunneling probability with an exponent of order $\sqrt{N}$. This can be written as $c^{-\sqrt{N}}$ for some $c > 1$. The annealing time τ_Q must then satisfy [20]:

$$\tau_Q c^{-\sqrt{N}} \int_0^1 c^{-\sqrt{N}x}\, dx = 1, \tag{17}$$

leading to $\tau_Q = \frac{\ln(c)\sqrt{N}c^{\sqrt{N}}}{1-c^{-\sqrt{N}}}$.

It is worth noting that this leads to the property that for any nonzero constant ϵ, the following holds:

$$\begin{aligned} \tau_Q &= \omega\left(e^{N^{\frac{1}{2}-\epsilon}}\right), \\ \tau_Q &= o\left(e^{N^{\frac{1}{2}+\epsilon}}\right), \end{aligned} \tag{18}$$

and therefore, it is often assumed that when the barrier widths are N-independent, the asymptotic complexity of QA may be expressed as $e^{\sqrt{N}}$. The above asymptotic is interpreted in the exponential sense.

While this provides only a heuristic estimation of the time complexity for solving a problem using quantum annealing, it offers insight into the attack's potency.

5.1 Problem rescaling

An intriguing question emerges: Can one estimate the complexity of QA for a specific problem, even if not universally? Although this task is compelling, it appears formidable. A pragmatic approach might involve rescaling the problem to analyze the energy landscapes of these rescaled versions, thereby assessing the behavior of barrier heights and widths. However, such analyses are typically feasible only for small-scale problems. Additionally, rescaled versions of certain problems might exhibit different behaviors compared to the original problems under scrutiny.

Next, consider the scaling of the Grain 128 and Grain 128a ciphers. For an algebraic attack on Grain ciphers, scaling could be performed as follows: One should scale the secret key of length l and the inner state of length $2l$, where l is the size parameter. For Grain 128 and Grain 128a, this parameter is equal to 128. The equations used to generate the keystream and update registers b and s should be similar; they should consist of the same number of monomials, and the monomials should have the same degrees as in the original Grain 128 or Grain 128a cipher. Only the indices should be changed to use the entire necessary inner state. Similar rescaling has been suggested in the case of Trivium cipher [22]. Lastly, one must also assume that at least $2l$ consecutive output keystream bits are known and that equations are generated for all these bits.

We will present an interesting result as follows:

Theorem 1. *Using the rescaled versions of Grain 128 and Grain 128a ciphers, the method presented above, we can show that an algebraic attack on these ciphers overtakes both brute-force and Grover's attacks for sufficiently large l, even if the barrier width grows with the number of variables. The only assumption here is that the QA problem with N variables has any advantage over simulated annealing (SA) in the exponential sense, and its complexity is equal to e^{N^β} for any $\beta < 1$.*

Proof. Let us only note that the number of variables n required for transformation of algebraic attack on Grain-like cipher to the QUBO problem depends linearly on l, namely is equal to $\sim dl$ for some positive d. Therefore, the asymptotic time complexity of solving such a QUBO problem may be expressed as $e^{(dl)^\beta}$. According to our assumption, $\beta < 1$. It is straightforward to note that there will be sufficiently large l, for which $e^{(dl)^\beta} < 2^l$, which means that for such l, our attack overtake brute force attack, as well as there will be sufficiently large l for which $e^{(dl)^\beta} < 2^{\frac{l}{2}}$, which means that for such l our attack overtakes Grover's attack.

It may be a surprising result for readers who know there is no quantum general database-search algorithm faster than Grover's. The hint here is that the algebraic attack, which we use here, uses the strictly algebraic structure of the cipher. The weaker this structure is, the weaker it should be the attack.

For a foundational perspective, throughout this paper, we will employ the heuristic estimation, suggesting that the time complexity of QA depends exponentially on $\sqrt{N}$.

5.2 The sparsity of the QUBO matrix and the range of the coefficients' values

This factor is vital for the practical execution of the attack. Presently, the topology employed by D-Wave quantum annealers does not always support direct connections between all required qubits. In such scenarios, logical qubits are represented by multiple physical qubits, leading to the creation of "chains." Thus, a sparser matrix often results in shorter chains, implying fewer physical qubits to embed the problem in the quantum annealer.

Additionally, the magnitude of the coefficients is crucial. In D-Wave computers, each coefficient is scaled to fit within the range $[-1, 1]$ for the coefficients of quadratic monomials and within $[-2, 2]$ [11] for the coefficients of linear monomials. This implies that if the differences between the original coefficient values are large, scaling errors could significantly impact the quality of computations.

Grain 128 and Gran 128a ciphers' coefficients in their QUBO matrix Q are within the range $[-40, 64]$ for quadratic monomials and within the range $[1, 64]$ for linear monomials. The difference between the extreme values of these ranges is so small that the smallest value after scaling will have a significant number in the second decimal place, which is high precision and sufficient for a physical quantum annealer.

6 Embedding the QUBO problem equivalent to the algebraic attack on the Grain 128 and Grain 128a ciphers in the D-Wave QPU Chip

Quantum computers based on quantum annealing solve optimization problems in three steps.

First, a given problem is formulated as an optimization problem compatible with a quantum annealer. This problem is represented by a graph known as the problem graph, where vertices represent variables, and edges connect those variables for which the coefficient of the quadratic monomial is nonzero.

Following this step, if the problem graph is not already in the native model, the Ising model, it is transformed into this model. Subsequently, the problem is embedded into the Quantum Processing Unit (QPU), which has a topology represented by a hardware graph. For clarity, the Ising model is described by the equation:

$$f_{\text{Ising}}(S) = \sum_{i} h_i s_i + \sum_{i<j} J_{ij} s_i s_j, \tag{19}$$

where s_i and s_j are spin variables, while the h_i coefficients represent external forces applied to the particles, and the J_{ij} coefficients represent the interaction forces between neighboring particles. The transformation from the QUBO problem to the Ising model involves a simple linear transformation: $s_i = 1 - 2x_i$. During the embedding process, known as minor-embedding, a subgraph of the hardware graph is identified to represent the problem graph. The hardware graph comprises nodes as qubits and edges as couplers, constituting the D-Wave machine's physical architecture. Biases, electronic signals applied to the qubits and couplers, correspond to the weights h_i and J_{ij}.

Finally, the quantum annealer iteratively performs the quantum annealing process to find an optimal solution to the given optimization problem.

Finding an efficient embedding, one that minimizes the use of physical qubits, is challenging, especially since the hardware graph is not complete. A quadratic Boolean optimization problem can only be embedded in D-Wave hardware if its graph is a subgraph of the D-Wave hardware graph. To address this, D-Wave Systems Inc. developed a heuristic method, *minorminer* [7], which iteratively attempts to find a feasible embedding.

Currently, three D-Wave QPU architectures are available: Chimera, Pegasus, and Zephyr. Qubits are oriented vertically or horizontally in all these topologies, as shown in Figure 2, forming a grid of identical fragments called unit cells. This paper will focus on the Zephyr topology for embedding the problem. In the Zephyr topology, each qubit, marked in green in Figure 2, connects to 16 orthogonal qubits via internal couplers (black dots), two qubits of the same orientation via odd couplers (red), and two similarly aligned qubits via external couplers (blue).

D-Wave hardware has certain limitations. For example, while arbitrary floating-point values can be sent via the D-Wave Leap interface, the analog circuitry has a fixed range and limited precision, affecting the fidelity of the embedded problem.

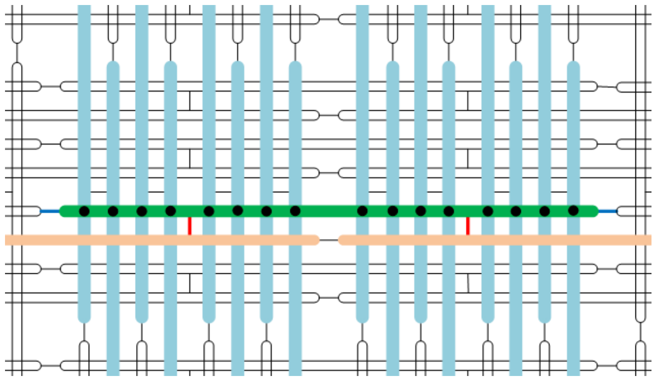

Fig. 2: Overall diagram of the Zephyr topology. *Source: based on [4]*

Moreover, the incomplete nature of the hardware graph leads to chaining—where one logical variable is mapped to multiple physical qubits—especially for densely connected problems. For instance, consider the optimization problem defined by the following objective function: $x_0x_1 + 3x_0x_6 + 2x_0x_7 + 2x_2x_3 + 4x_0x_4 + 3x_2x_5 + 6x_2x_7 + x_3x_5 + 2x_3x_6 + 4x_5x_6 + 2x_5x_7 + 6x_4x_6 + x_4x_7 + x_6x_7 + 6x_0 + 2x_3 + x_5 + 3x_6 + 4x_7$. The graph of this problem is displayed in Figure 3. This problem

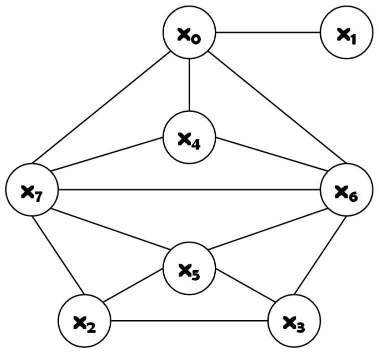

Fig. 3: Graph of the example problem.

involves eight logical variables, with the maximum vertex degree in its problem graph being 5 for variables x_6 and x_7. Despite on Zephyr topology allowing a single qubit to connect to 20 other qubits, the problem requires chaining during embedding, necessitating nine physical qubits. Figure 4 on the left shows the problem embedding in the Zephyr topology grid, while on the right shows the result of embedding the problem in the physical QPU chip with the Zephyr topology, where a chain of two physical qubits represents the boolean x_7.

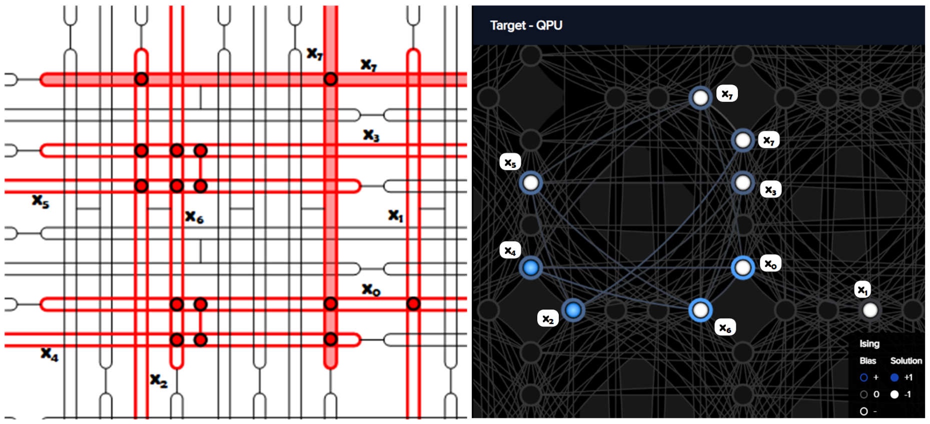

Fig. 4: Result of embedding the example problem in the Zephyr architecture.

Embedding the QUBO problem for a full attack on the Grain 128 cipher would require $5,751$ logical qubits and $77,496$ couplers. The soon-to-be-available D-Wave Advantage 2, which employs the Zephyr architecture, will contain more than $7,000$ physical qubits and more than $60,000$ couplers. Unfortunately, despite the similarities between the required and available resources, it will not be possible to embed problems for a full attack on the Grain 128 and Grain 128a ciphers due to their size and density.

Currently, only an experimental prototype of the system with the next-generation Zephyr architecture is available, featuring merely 576 physical qubits. Therefore, for practical research on embedding the QUBO problem for ciphers like Grain 128, an embedding simulation was performed using the *minorminer.find_embedding()* function. A target graph was defined in the Zephyr topology with $7,440$ qubits. In this hardware graph, we successfully embedded a smaller instance of the Grain 128 cipher, Grain 20, which uses 20-bit registers and proportionally shifted indexes of variables in polynomials. Due to the shift in variable indexes, some inefficient monomials were omitted. Of course, the reduced version of the Grain cipher does not provide cryptographic security. It is only intended to provide an illustration of the density of the QUBO problem. The obtained reduced instance of the Grain cipher is as follows:

The internal state is stored in two 20-bit registers:

– LFSR with internal state:

$$(s_i, s_{i+1}, \ldots, s_{i+19})$$

accompanied by the feedback function:

$$s_{i+20} = s_i + s_{i+1} + s_{i+6} + s_{i+11} + s_{i+15};$$

– NFSR with internal state:

$$(b_i, b_{i+1}, \ldots, b_{i+19})$$

accompanied by the feedback function:

$$b_{i+20} = s_i + b_i + b_{i+4} + b_{i+9} + b_{i+15} + \\ b_i b_{i+10} + b_{i+4} b_{i+9} + b_{i+6} b_{i+8} + b_{i+11} b_{i+12}.$$

Of the 40 bits of the internal state, 6 are input to the function $h(x)$ as depicted by the following equation:

$$h(x) = x_0 x_1 + x_2 x_3 + x_4 x_2 x_5,$$

where the variables x_k correspond to the following bits of the state:

$$s_{i+2}, s_{i+3}, b_{i+15}, s_{i+7}, b_{i+2}, s_{i+15}.$$

The function used to determine the output bit is as follows:

$$y_i = \sum_{j \in \mathcal{A}} b_{i+j} + h(x) + s_{i+14}, \tag{20}$$

where $\mathcal{A} = \{0, 2, 7, 14\}$.

The graph of the QUBO problem for the reduced Grain cipher, dubbed Grain 20, consists of 675 vertices and 6,191 edges. The maximum degree of a vertex in the problem graph is 100 for 15 vertices. Embedding this problem into the Zephyr hardware graph requires 4,852 qubits and 10,368 couplers. For 639 logical variables, chains ranging from 2 to 45 physical qubits need to be created.

Since the D-Wave computer is designed to solve various optimization problems within a certain class, it may be worth considering a hardware graph dedicated solely to attacks against the Grain 128 and Grain 128a ciphers. The ideal hardware graph for this purpose would have each physical qubit corresponding to exactly one logical qubit. Additionally, the degree of each physical qubit and its number of connections should match the degrees of the vertices and edges in the problem graph. This way, no chains would be created during embedding. However, these requirements can be relaxed by defining a connectivity parameter, which specifies the maximum number of couplers for a single physical qubit (i.e., the degree of the qubit in the hardware graph).

The problem graph for the Grain 128 cipher consists of 5,751 vertices and 77,496 edges, with the maximum vertex degree being 225 for 98 vertices. The degrees of all vertices of the problem graph for the Grain 128 cipher are shown in Table 5.

degree of vertex	number of vertices	degree of vertex	number of vertices	degree of vertex	number of vertices	degree of vertex	number of vertices
4	259	44	10	89	32	163	5
8	452	46	23	92	4	166	5
10	2	48	10	94	5	168	5
15	777	49	9	95	105	177	2
16	680	53	17	96	6	183	8
17	1295	55	16	108	3	184	1
18	1582	56	2	110	17	192	4
20	6	57	7	122	1	196	1
24	7	58	1	123	1	199	8
25	14	59	3	125	5	201	1
27	16	60	8	127	6	203	4
31	1	63	1	129	5	205	2
32	1	65	43	130	3	207	8
33	8	72	7	143	1	209	3
34	8	75	1	145	10	211	32
35	24	77	9	147	1	225	98
36	30	79	5	149	5		
39	1	80	2	161	11		
41	2	82	1	162	3		

Table 5: The degrees of all vertices of the problem graph for the Grain 128 cipher.

Because a maximum vertex degree of 225 is too large, let us assume that the connectivity parameter is set to 50. Each vertex in the problem graph with a degree greater than 50 is represented by a chain of qubits with a length of $\left\lceil \frac{\deg(x_i)}{50} \right\rceil$, where $\deg(x_i)$ is the degree of vertex x_i. As a result, we obtain 534 chains: 275 chains have a length of 2 qubits, 52 chains have a length of 3 qubits, 50 chains have a length of 4 qubits, and 157 chains have a length of 5 physical qubits. Moreover, the degree of each physical qubit or chain of qubits in the hardware graph exactly matches the degree of the corresponding vertex in the problem graph. In summary, a dedicated hardware graph consisting of $6,908$ physical qubits and $78,653$ couplers can be constructed to attack the Grain 128 cipher.

A similar analysis can be performed for the Grain 128a cipher. Its problem graph consists of $6,751$ vertices and $94,708$ edges, with the maximum vertex degree being 249 for a single vertex. The degrees of all vertices of the problem graph for the Grain 128a cipher are shown in Table 6.

degree of vertex	number of vertices	degree of vertex	number of vertices	degree of vertex	number of vertices	degree of vertex	number of vertices
2	81	46	6	91	1	163	1
4	288	48	4	93	32	169	11
8	450	50	18	98	4	171	5
10	1	52	10	99	96	174	1
15	777	53	25	100	5	176	2
17	1295	55	7	103	1	180	6
20	934	56	2	107	6	182	4
22	1873	57	5	108	5	185	2
24	343	58	1	111	1	197	9
25	14	59	13	114	3	198	1
26	8	60	7	116	8	204	4
27	17	61	2	122	9	208	1
28	10	62	1	135	3	219	7
34	2	65	31	136	2	221	2
35	1	67	2	137	6	223	14
36	14	69	12	139	8	225	3
37	9	72	7	140	3	227	32
38	5	81	1	151	3	245	96
39	23	83	9	153	1	249	1
40	18	84	1	155	2		
41	3	86	2	157	3		
44	5	87	4	159	6		

Table 6: The degrees of all vertices of the problem graph for the Grain 128a cipher.

Continuing with the previous hardware graph design assumptions, we obtain 552 chains: 179 chains have a length of 2 qubits, 156 chains have a length of 3 qubits, 47 chains have a length of 4 qubits, 73 chains have a length of 5 qubits, and 97 chains have a length of 6 physical qubits. The chain length for a dedicated architecture with a given connectivity degree *cnv* (the maximal number of connections to a qubit) for a vertex *v* of degree *deg*(*v*) can be calculated as follows:

$$\begin{cases} 1, & \text{if } deg(v) < cnv, \\ \frac{deg(v)-2}{cnv-2}, & \text{if } deg(v) > 2 \text{ and } cnv > 2. \end{cases} \tag{21}$$

Here, we assume that each qubit has the same maximal connectivity *cnv* and that $cnv \geq 3$. Note that if $cnv < 3$, it will be impossible to embed graphs in which any vertex has a degree of at least 3 into such an architecture, even making chains. We also note that assuming we have linear chains, each inner element of the chain uses two connections for its links to other elements in the chain. Elements at both the beginning and end of the chain "lose" only one connection to another element in the chain. Therefore, at least one edge is always going from the beginning and one edge from the end of the chain to some other physical (also logical) qubits. This leads to the result that if $deg(v) > cnv > 2$, the length of a linear chain can be calculated as $\frac{deg(v)-2}{cnv-2}$.

A dedicated hardware graph consisting of $8,079$ physical qubits and $96,036$ couplers can be constructed for an attack on the Grain 128a cipher.

Based on similar calculations, we estimated the resources required for dedicated architectures to embed the problem graph for attacks on both the Grain 128 and Grain 128a ciphers, assuming a connectivity parameter of 20. Table 7 compares the resources for publicly available and dedicated architectures, depending on the maximum vertex degree in the hardware graph.

Quantum Annealer	#Qubits	#Couplers	Connectivity	Chain lenght
DWave 2000Q	$2,048$	$6,016$	6	
DWave Advantage	$> 5,000$	$> 35,000$	15	
DWave Advantage 2	$> 7,000$	$> 60,000$	20	
Dedicated (Grain 128)	$\approx 5,800$	$\approx 77,500$	max 225	1
Dedicated (Grain 128)	$\approx 6,900$	$\approx 78,700$	max 50	5
Dedicated (Grain 128)	$\approx 9,600$	$\approx 81,500$	max 20	13
Dedicated (Grain 128a)	$\approx 6,800$	$\approx 95,000$	max 249	1
Dedicated (Grain 128a)	$\approx 8,100$	$\approx 96,100$	max 50	6
Dedicated (Grain 128a)	$\approx 13,100$	$\approx 101,050$	max 20	14

Table 7: Comparison of required resources for different quantum annealer architectures.

7 Discussion

Quantum annealing is a subject of debate for many researchers and professionals. Common arguments cited against considering quantum annealing as a serious threat include:

1. NIST contends that "due to their specialized nature, these analog quantum devices are not believed to be of relevance to cryptanalysis." [8];
2. the behavior of physical quantum annealers has not been definitively proven to be genuinely quantum;
3. the computational complexity of QA is not well-understood;
4. a prevalent assumption exists that it is not a serious threat (this argument is often heard).

While we acknowledge these concerns, we must not limit our research to gate-based quantum computers and the primary applications of Grover's and Shor's algorithms (and their variants). Otherwise, other potentially useful approaches could be overlooked.

To summarize our contributions:

1. We have demonstrated how to transform the algebraic attack on Grain 128 and Grain 128a ciphers and precisely defined the problem—identifying the number of logical variables and their interconnections;

2. We propose that it may be possible that a dedicated quantum annealer for the problem we presented will be constructed in the near future, perhaps within a few years;

3. We suggest that the time complexity of our attack may exceed that of a classical brute-force attack, although this claim is challenging to validate. Analysis suggests that quantum annealing could offer a time complexity advantage, exponentially on the order of $\sqrt{N}$, over simulated annealing (and brute force attack) in our attack scenario;

4. We have also demonstrated that applying our attack to rescaled versions of the Grain 128 and Grain 128a ciphers, specifically Grain l and Grain la, where l represents the key size, yields an advantage over both brute-force and Grover's attack for sufficiently large l, provided that quantum annealing has any exponential advantage over simulated annealing. Specifically, it would suffice if the time complexity of quantum annealing for a problem with N variables is e^{N^β} for any $\beta < 1$.

Given the nascent state of this area of research, much remains to be explored. However, if there is even a slight possibility that the attacks we have presented could pose a threat, they warrant serious consideration.

Comparing our attack with other quantum attacks described in [2] and [18], our attack appears to have greater potential for practical realization. Upon analyzing Tables 7 and 1, it becomes evident that our method is technically easier to implement and is much more realistic than the quantum circuits for Grover's algorithm presented in [2]. On the other hand, the method presented in [18], similar to our approach, has a time complexity that strongly depends on the exact form of the algebraic equations derived from the cipher's structure. However, no estimates for the quantum resources required to implement this attack have been presented. As a result, it is difficult to determine whether the attack utilizing the adaptation of the HHL algorithm can be practically executed.

In the end, it is also worth noting that an algebraic attack using quantum annealing on Grain 128 and Grain 128a requires much less logical qubits than the attack on other algorithms with similar security levels. The results are presented in the Table 8

Problem	N =#Qubits	$\sqrt{N}$	Sec. Lev. ($\log_2 e \cdot \sqrt{N}$)	Sec. Lev. ($\frac{\pi}{4} \cdot \sqrt{N}$)
IFP	$\approx 2,360,000$	1 536	2 216	1,206
DLP	$\approx 18,900,000$	4 347	6 272	3,414
AES-128	$\approx 30,000$	173	250	136
Grain 128	5,751	76	109	60
Grain 128a	6,751	82	119	65

Table 8: Comparison of reduction of different algorithms (ensuring about 128-bit classical security level) to the QUBO problem, assuming the time complexity $a^{\sqrt{N}}$ for two values of a: e and $2^{\frac{\pi}{4}}$, and therefore security level: $\log_2 e \cdot \sqrt{N}$ and $\frac{\pi}{4} \cdot \sqrt{N}$

Every cryptographic algorithm should be assessed for its resilience against quantum annealing attacks. Although estimating the computational complexity of quantum annealing is inherently difficult, we believe adopting the same assumptions NIST used in the PQC competition is reasonable. Specifically, for the first security level, a quantum attack on a given algorithm should be no more feasible than a quantum attack on AES-128, which involves approximately 30,000 logical variables. For the third level, it should be comparable to AES-192 (around 45,000 logical variables), and for the fifth level, akin to AES-256 (approximately 60,000 logical variables).

Acknowledgments

We thank Mateusz Buczek and Przemysław Kubiak for their suggestions on improving our paper. We also thank Krzysztof Mańk for useful discussions.

References

1. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: a new version of grain-128 with optional authentication. International Journal of Wireless and Mobile Computing **5**(1), 48–59 (2011)
2. Anand, R., Maitra, S., Maitra, A., Mukherjee, C.S., Mukhopadhyay, S.: Resource estimation of grovers-kind quantum cryptanalysis against fsr based symmetric ciphers. Cryptology ePrint Archive (2020)
3. Berbain, C., Gilbert, H., Maximov, A.: Cryptanalysis of grain. In: International Workshop on Fast Software Encryption. pp. 15–29. Springer (2006)
4. Boothby, K., King, A.D., Raymond, J.: Zephyr topology of d-wave quantum processors. D-Wave Technical Report Series (2021)
5. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete applied mathematics **123**(1-3), 155–225 (2002)
6. Burek, E., Wroński, M., Mańk, K., Misztal, M.: Algebraic attacks on block ciphers using quantum annealing. IEEE Transactions on Emerging Topics in Computing **10**(2), 678–689 (2022)

7. Cai, J., Macready, W.G., Roy, A.: A practical heuristic for finding graph minors. arXiv preprint arXiv:1406.2741 (2014)
8. Chen, L., Chen, L., Jordan, S., Liu, Y.K., Moody, D., Peralta, R., Perlner, R.A., Smith-Tone, D.: Report on post-quantum cryptography, vol. 12. US Department of Commerce, National Institute of Standards and Technology … (2016)
9. Chen, L., Moody, D., Liu, Y.: Nist post-quantum cryptography standardization. Transition **800**(131A), 164 (2017)
10. Collins, H., Nay, C.: Ibm unveils 400 qubit-plus quantum processor and next-generation ibm quantum system two (2022)
11. D-WAVE, T.Q.C.C.: Getting started with the d-wave system. User manual (2020)
12. Dinur, I., Shamir, A.: Breaking grain-128 with dynamic cube attacks. In: Fast Software Encryption: 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers 18. pp. 167–187. Springer (2011)
13. Dworkin, M.J., Barker, E.B., Nechvatal, J.R., Foti, J., Bassham, L.E., Roback, E., Dray Jr, J.F.: Advanced encryption standard (aes) (2001)
14. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
15. Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Phys. Rev. Lett. **103**, 150502 (Oct 2009). https://doi.org/10.1103/PhysRevLett.103.150502, `https://link.aps.org/doi/10.1103/PhysRevLett.103.150502`
16. Hell, M., Johansson, T., Maximov, A., Meier, W.: A stream cipher proposal: Grain-128. In: 2006 IEEE International Symposium on Information Theory. pp. 1614–1618. IEEE (2006)
17. Hell, M., Johansson, T., Meier, W.: Grain: a stream cipher for constrained environments. International journal of wireless and mobile computing **2**(1), 86–93 (2007)
18. Liu, W., Gao, J.: Quantum security of grain-128/grain-128a stream cipher against hhl algorithm. Quantum Information Processing **20**, 1–22 (2021)
19. Lucas, A.: Ising formulations of many np problems. Frontiers in physics **2**, 5 (2014)
20. Mukherjee, S., Chakrabarti, B.K.: Multivariable optimization: Quantum annealing and computation. The European Physical Journal Special Topics **224**(1), 17–24 (2015)
21. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. Ieee (1994)
22. Tian, Y., Chen, G., Li, J.: On the design of trivium. Cryptology ePrint Archive, Paper 2009/431 (2009), `https://eprint.iacr.org/2009/431`, `https://eprint.iacr.org/2009/431`
23. Todo, Y., Isobe, T., Meier, W., Aoki, K., Zhang, B.: Fast correlation attack revisited: cryptanalysis on full grain-128a, grain-128, and grain-v1. In: Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II 38. pp. 129–159. Springer (2018)
24. Wroński, M.: Practical solving of discrete logarithm problem over prime fields using quantum annealing. In: Groen, D., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds.) Computational Science – ICCS 2022. pp. 93–106. Springer International Publishing, Cham (2022)