Article

Quantum Attacks on MIBS Block Cipher Based on Bernstein–Vazirani Algorithm

Huiqin Xie [1,2], Zhangmei Zhao [1], Ke Wang [1], Yanjun Li [3] and Hongcai Xin [1,*]

1 Department of Cryptography Science and Technology, Beijing Electronic Science and Technology Institute, Beijing 100070, China
2 Key Laboratory of Cryptography of Zhejiang Province, Hangzhou Normal University, Hangzhou 311121, China
3 Information Industry Information Security Evaluation Center, The 15th Research Institute of China Electronics Technology Group Corporation, Beijing 100083, China
* Correspondence: hcxin_besti@163.com

Abstract: Because of the substantial progress in quantum computing technology, the safety of traditional cryptologic schemes is facing serious challenges. In this study, we explore the quantum safety of the lightweight cipher MIBS and propose quantum key-recovery attacks on the MIBS cipher by utilizing Grover's algorithm and Bernstein–Vazirani algorithm. We first construct linear-structure functions based on the 5-round MIBS cipher according to the characteristics of the linear transformations, and then we obtain a quantum distinguisher of the 5-round MIBS cipher by applying Bernstein–Vazirani algorithm to the constructed functions. Finally, utilizing this distinguisher and Grover's algorithm, we realize a 7-round key-recovery attack on the MIBS cipher, and then we expand the attack to more rounds of MIBS based on a similar idea. The quantum attack on the 7-round MIBS requires 156 qubits and has a time complexity of $2^{10.5}$. An 8-round attack requires 179 qubits and has a time complexity of 2^{22}. Compared with existing quantum attacks, our attacks have better time complexity when attacking the same number of rounds.

Keywords: quantum cryptanalysis; MIBS; key-recovery attack; Bernstein–Vazirani algorithm

MSC: 94A60

check for updates

Citation: Xie, H.; Zhao, Z.; Wang, K.; Li, Y.; Xin, H. Quantum Attacks on MIBS Block Cipher Based on Bernstein–Vazirani Algorithm. *Mathematics* **2024**, *12*, 2678. https://doi.org/10.3390/math12172678

Academic Editor: Cheng-Chi Lee

Received: 30 July 2024
Revised: 25 August 2024
Accepted: 26 August 2024
Published: 28 August 2024

1. Introduction

There has been substantial progress in quantum computer development in recent years. Experts in physics, quantum computing, and computer architecture are committed to realizing quantum computers. While quantum computing may bring benefits to research in many fields, it also brings challenges, especially for cryptography.

The key difference in quantum computing from classical information computing and processing is parallelism, which comes from the principle of superposition. This parallelism superiority makes it possible to execute a great quantity of computational paths simultaneously on quantum computers, so that some computational problems that cannot be solved by electronic computers may be solved by quantum computers. For example, factoring large integers will be solvable on quantum computers by utilizing Shor's algorithm [1]; however, the security of some widely used public key algorithms is built on it.

Apart from public key schemes, symmetric cryptography is under the threat of quantum attacks as well. Grover's algorithm [2] is the most representative example. It can be used for any unstructured search and brings a quadratic speedup. Searching a specific marked target in an M-element database using Grover's algorithm needs only $O(\sqrt{M})$ complexity, while classical algorithms need at least $O(M)$ complexity. Another famous example is Simon's algorithm [3], which was introduced to find periods. It is also frequently applied to cryptanalysis of symmetric ciphers. At first, Simon's algorithm was exploited to distinguish between a Feistel structure and a random function [4–6]. Afterwards, it was also utilized to find the key of the Even-Mansour (EM) scheme [6,7]. Lender et al. then utilized Grover's algorithm and Simon's algorithm simultaneously to extract the keys of FX ciphers [8]. Dong and Wang applied a similar method to attack a Feistel cipher [9] and the generalized Feistel cipher [10]. As for the Substitution–Permutation Network (SPN) structure, quantum attacks on the Advanced Encryption Standard (AES) algorithm were investigated by Jaques et al. [11]. Halak et al. evaluated the computation costs and performance of several quantum-attack-resilient cryptographic algorithms [12]. Recently, the research on the cryptanalysis of symmetric schemes has begun to pay attention to the Bernstein–Vazirani (BV) algorithm [13] and has obtained some good results [14,15].

In addition to the specific attacks on certain symmetric ciphers, analytic tools for symmetric ciphers must also be investigated for accurate security evaluation. In this direction, Grover's algorithm was used to accelerate the search involved in differential attacks [11,16], and it was also used in the search part of linear attacks and their variants [17]. Afterwards, the BV algorithm was exploited for finding differentials [14,18,19]. Zhou and Yuan combined the BV algorithm and Grover's algorithm for attacking Feistel ciphers [15]. Their attack strategy was inspired by the attack presented in [8,9], the main innovation being that it uses BV algorithm to distinguish the functions with nonzero linear structures from random functions instead of using Simon's algorithm to distinguish functions with nonzero periods from random functions. Quantum algorithms are also applied to the collision attack on Hash functions [20,21]. Quantum cryptanalysis under the related-key model has also been studied [22,23]. The attacks mentioned above all exhibit the acceleration superiority of quantum algorithms in symmetric cryptanalysis over classical algorithms.

In this study, we investigate quantum attacks on the MIBS cipher, which is a lightweight algorithm with a Feistel structure and designed specifically for constrained environments [24]. First, by analyzing the characteristics of the MIBS cipher, we construct linear-structure functions based on the 5-round encryption of the MIBS cipher. Then, we combine this function with the BV algorithm to construct a distinguisher of the 5-round MIBS cipher. Afterwards, we utilize Grover's algorithm and this distinguisher to implement a 7-round key-recovery attack on the MIBS cipher. Based on a similar idea, we further use the same distinguisher to implement 8-round and 9-round key-recovery attacks on the MIBS cipher. The 7-round, 8-round, and 9-round attacks require 156, 179, and 194 qubits, respectively, and their complexity are $2^{10.5}$, 2^{22}, and 2^{32}, respectively. Compared with existing quantum attacks, the quantum attacks presented in this article have the minimum complexity when attacking the same number of rounds. Our work further explores the "BV-meet-Grover" attack strategy and helps to evaluate the security of the MIBS cipher.

We first construct a periodic function based on the 5-round encryption, and then we combine Simon's algorithm with the found periodic function to obtain a 5-round quantum distinguisher. As the number of rounds increases, the encryption function becomes more complex, making it increasingly difficult to find periodic functions. The 5-round distinguisher is the longest distinguisher we can find. Using the constructed 5-round distinguisher to attack 7-round, 8-round, and 9-round MIBS requires guessing 21, 44, and 64 bits of subkeys, respectively. The complexity of the 10-round attack exceeds that of the key exhaustive attack. Therefore, we only provide 7-round, 8-round, and 9-round attacks on MIBS.

Organization. Section 2 recalls the the fundamental concepts of quantum computing and cryptography; Section 3 constructs a 5-round quantum distinguisher of the MIBS cipher; Section 4 presents key-recovery attacks on the MIBS cipher; Section 5 provides a summary.

2. Preliminaries

2.1. Mibs Block Cipher

Lightweight cipher MIBS applies a standard Feistel structure [24]. Each block has 64 bits, and the key length supports 80-bit and 64-bit. We only consider the 64-bit version in this paper. MIBS has 32 rounds. Figure 1 shows the encryption structure of one round. L_{i-1} and R_{i-1} are the left branch and right branch of the input of the i-th round, respectively. L_i and R_i are the left branch and right branch of the output of the i-th round, respectively. K_i is the i-th subkey ($i = 1, 2, \cdots, 32$). In the i-th round, L_{i-1} and K_i are input to the function F, and the XOR of R_{i-1} and the output of F is the left branch L_i of the output. The right branch R_i of the output directly takes the value of L_{i-1}. All operations appearing in MIBS are nibble-based. A nibble contains four bits.

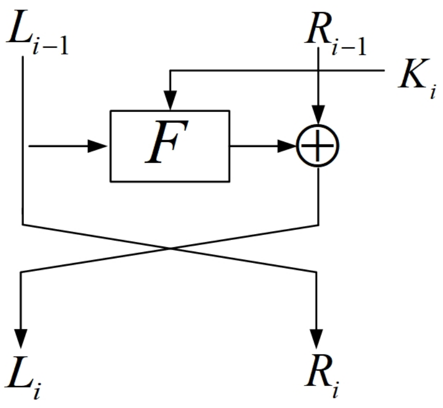

Figure 1. The i-th round function of MIBS.

Suppose the plaintext is $M \in \mathbb{F}_2^{64}$, then the encryption process of MIBS is as follows:

1. Divide M into two 32-bit parts $M = L_0||R_0$.
2. For $i = 1, 2, \cdots, 32$, compute

$$\begin{cases} L_i = R_{i-1} \oplus F(L_{i-1} \oplus K_i) \\ R_i = L_{i-1}, \end{cases}$$

   where K_i is the subkey of the i-th round generated by the key scheduling, and function F is defined below.
3. Output the ciphertext $C = R_{32}||L_{32}$. (The ciphertext is obtained by exchanging the left and right branches of the output $L_{32}||R_{32}$ of the last iteration.)

The function $F : \mathbb{F}_2^{32} \to \mathbb{F}_2^{32}$ maps eight nibbles to eight nibbles:

$$\begin{aligned} F : (\mathbb{F}_2^4)^8 &\to (\mathbb{F}_2^4)^8 \\ x &\to P \circ M \circ S(x), \end{aligned}$$

where S is the substitution transformation, M and P are the mixing layer and the permutation layer, respectively. The S layer implements 8 identical Sboxes of 4 bits, all denoted s.

$$\begin{aligned} S : (\mathbb{F}_2^4)^8 &\to (\mathbb{F}_2^4)^8 \\ (x_8, x_7, \cdots, x_1) &\to (s(x_8), s(x_7), \cdots, s(x_1)). \end{aligned}$$

The definition of the Sbox s is presented in Table 1.

Table 1. SBox s.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$s(x)$	5	15	3	8	13	10	12	0	11	5	7	14	2	6	1	9

The M layer mixes 8 nibbles using the XOR operation:

$$\begin{gathered} M : (\mathbb{F}_2^4)^8 \rightarrow (\mathbb{F}_2^4)^8 \\ (y_8, y_7, \cdots, y_1) \rightarrow (u_8, u_7, \cdots, u_1), \end{gathered}$$

where

$$\begin{aligned} u_1 &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_7 \\ u_2 &= y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8 \\ u_3 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7 \oplus y_8 \\ u_4 &= y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_8 \\ u_5 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_6 \\ u_6 &= y_1 \oplus y_2 \oplus y_3 \oplus y_6 \oplus y_7 \\ u_7 &= y_2 \oplus y_3 \oplus y_4 \oplus y_7 \oplus y_8 \\ u_8 &= y_1 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_8. \end{aligned}$$

The P layer rearranges the input 8 nibbles in the order given in Table 2. That is, the P transformation is defined as

$$\begin{gathered} P : (\mathbb{F}_2^4)^8 \rightarrow (\mathbb{F}_2^4)^8 \\ (u_8, u_7, \cdots, u_1) \rightarrow (z_8, z_7, \cdots, z_1), \end{gathered}$$

where

$$\begin{gathered} z_1 = u_3, \quad z_2 = u_1, \quad z_3 = u_4, \quad z_4 = u_7, \\ z_5 = u_8, \quad z_6 = u_5, \quad z_7 = u_6, \quad z_8 = u_2. \end{gathered}$$

Table 2. P transformation.

-	1	2	3	4	5	6	7	8
P	2	8	1	3	6	7	4	5

For the convenience of cryptanalysis, we combine the transformations P and M. Let $PM = P \circ M$, which is a linear transformation and operates as follows:

$$\begin{gathered} PM : (\mathbb{F}_2^4)^8 \rightarrow (\mathbb{F}_2^4)^8 \\ (y_8, y_7, \cdots, y_1) \rightarrow (z_8, z_7, \cdots, z_1), \end{gathered} \tag{1}$$

where

$$\begin{aligned} z_1 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7 \oplus y_8 \\ z_2 &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_7 \\ z_3 &= y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_8 \\ z_4 &= y_2 \oplus y_3 \oplus y_4 \oplus y_7 \oplus y_8 \\ z_5 &= y_1 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_8 \\ z_6 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_6 \\ z_7 &= y_1 \oplus y_2 \oplus y_3 \oplus y_6 \oplus y_7 \\ z_8 &= y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8. \end{aligned}$$

The construction of F is depicted in Figure 2. $x_8, x_7, \cdots, x_1$ denotes the input of function F. First, the input is XORed with the 32-bit subkey K_i. Then, all nibbles are performed on 8 identical Sboxes s, respectively. The output of the Sboxes are denoted as $y_8, y_7, \cdots, y_1$. Subsequently, F performs PM transformation on these nibbles. PM is composed of XOR operations and position permutations as defined in Equation (1). We mark the PM transformation with a dashed box.

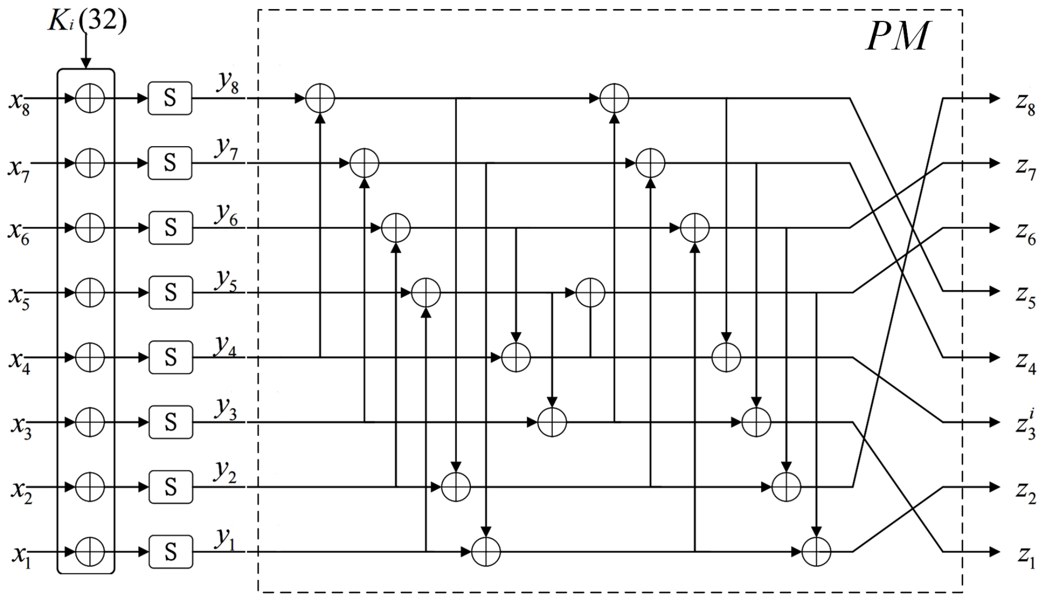

Figure 2. Construction of the function F.

The attack on the MIBS cipher needs to use the inverse of PM.

$$PM^{-1} : (\mathbb{F}_2^4)^8 \to (\mathbb{F}_2^4)^8$$
$$(z_8, z_7, \cdots, z_1) \to (y_8, y_7, \cdots, y_1),$$

where

$$\begin{aligned}
y_1 &= z_2 \oplus z_4 \oplus z_6 \oplus z_7 \oplus z_8 \\
y_2 &= z_1 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8 \\
y_3 &= z_1 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_6 \\
y_4 &= z_2 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_7 \\
y_5 &= z_2 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8 \\
y_6 &= z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_8 \\
y_7 &= z_1 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_7 \oplus z_8 \\
y_8 &= z_1 \oplus z_2 \oplus z_3 \oplus z_4 \oplus z_6 \oplus z_7.
\end{aligned}$$

The matrix forms of PM and PM^{-1} are

$$PM = \begin{bmatrix} 1&1&1&0&1&1&0&1 \\ 0&1&1&0&0&1&1&1 \\ 0&0&1&1&1&0&1&1 \\ 1&0&0&1&1&1&0&1 \\ 1&1&0&0&1&1&1&0 \\ 1&0&1&1&0&1&1&1 \\ 0&1&1&1&1&1&1&0 \\ 1&1&0&1&1&0&1&1 \end{bmatrix}, \quad PM^{-1} = \begin{bmatrix} 0&1&1&0&1&1&1&1 \\ 1&1&1&1&0&1&0&1 \\ 1&0&1&1&1&0&1&1 \\ 1&1&0&1&1&1&1&0 \\ 0&1&1&1&0&1&1&0 \\ 0&0&1&1&1&1&0&1 \\ 1&1&0&1&1&0&0&1 \\ 1&1&1&0&1&0&1&0 \end{bmatrix}.$$

The key scheduling of MIBS generates 32 32-bit round keys $K_1, K_2, \cdots, K_{32}$. Suppose the user key $K = (k_{63}, k_{62}, \cdots, k_0)$. K is stored in a 64-bit key register *state*. Initialize the state of the register as $state = [k_{63}, k_{62}, \cdots, k_0]$, i.e., $state_i = k_i$. The round key $K_i = k^i_{31}k^i_{30}\cdots k^i_0$ in the i-th round is equal to the leftmost 32 bits of the current register. Namely,

$$K_i = k^i_{31}k^i_{30}\cdots k^i_0 = state_{63}, state_{62}, \cdots, state_{32}.$$

After extracting the round key K_i, update the register as follows:

$$\begin{aligned}&state = state >>> 15,\\&state[63:60] = s(state[63:60]),\\&state[15:11] = state[15:11] \oplus Round\text{-}Counter,\end{aligned}$$

where $>>>$ denotes rotation to the right, and $state[j:i]$ denotes the j-th to the i-th bits of the register. $s(\cdot)$ is the Sbox defined in Table 1.

2.2. Bernstein–Vazirani Algorithm

The BV algorithm was introduced to find a secret vector $\rho \in \mathbb{F}_2^n$ when given the function $f(x) = \rho \cdot x = \sum_{i=1}^n \rho_i x_i$ [13]. The steps of the BV algorithm are illustrated in Figure 3.

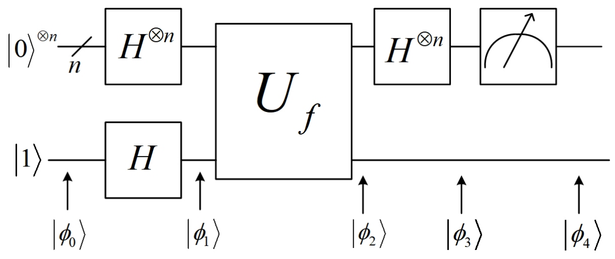

Figure 3. BV algorithm.

The notation H in Figure 3 denotes the Hadamard gate, which maps the state $|0\rangle$ to the state $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and maps the state $|1\rangle$ to the state $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. The notation $H^{\otimes n}$ is a product of n Hadamard gates. Performing $H^{\otimes n}$ on $|0\rangle^{\otimes n}$ gives the state

$$\begin{aligned}H^{\otimes n}|0\rangle^{\otimes n} =&H|0\rangle \otimes H|0\rangle \otimes \cdots \otimes H|0\rangle\\=&\frac{1}{\sqrt{2^n}}(|0\rangle \oplus |1\rangle)^{\otimes n}\\=&\frac{1}{\sqrt{2^n}}\sum_{x\in\mathbb{F}_2^n}|x\rangle.\end{aligned}$$

The notation U_f denotes the unitary operator of f, which operates as follows:

$$U_f : |x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle.$$

The symbol at the end of the first quantum wire in Figure 3 denotes a measurement. Suppose a quantum state $\sum_{x\in\mathbb{F}_2^n} \alpha_x|x\rangle$ is measured, where α_x is a complex number and called the amplitude of $|x\rangle$, then for any vector $x \in \mathbb{F}_2^n$, the probability of the measurement result being x is equal to $|\alpha_x|^2$. The quantum states in Figure 3 are defined as follows:

$$\begin{aligned}
|\phi_0\rangle &= |0\rangle^{\otimes n}|1\rangle;\\
|\phi_1\rangle &= \sum_{x\in\mathbb{F}_2^n} \frac{|x\rangle}{\sqrt{2^n}} \cdot \frac{|0\rangle - |1\rangle}{\sqrt{2}};\\
|\phi_2\rangle &= \sum_{x\in\mathbb{F}_2^n} \frac{(-1)^{f(x)}|x\rangle}{\sqrt{2^n}} \frac{|0\rangle - |1\rangle}{\sqrt{2}};\\
|\phi_3\rangle &= \sum_{y\in\mathbb{F}_2^n} (\frac{1}{2^n} \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+y\cdot x})|y\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}};\\
|\phi_4\rangle &= |\rho\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}.
\end{aligned}$$

The last equation holds because

$$\frac{1}{2^n} \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+y\cdot x} = \frac{1}{2^n} \sum_{x\in\mathbb{F}_2^n} (-1)^{(\rho\oplus y)\cdot x} s = \begin{cases} 1 & y = \rho \\ 0 & y \neq \rho. \end{cases}$$

After measurement, the output is ρ with a probability of 1.

2.3. Linear Structure

Definition 1. *For a Boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2^n$, $\alpha \in \mathbb{F}_2^m$ is called a linear structure of f if*

$$f(x \oplus \alpha) \oplus f(x) = \beta, \ \ \forall x \in \mathbb{F}_2^m \tag{2}$$

holds for some $\beta \in \mathbb{F}_2^n$.

If β in Equation (2) is the n-dimensional zero vector 0^n, then α is also called a period of f. If a function has a nonzero period, we call it a periodic function. If a function has a nonzero linear structure, we call it a linear structure function. Particularly, for the case $n = 1$, Li et al. presented a quantum algorithm that can determine whether f has a nonzero linear structure in polynomial time [19].

Theorem 1 ([19])**.** *Any nonzero linear structure of $f : \mathbb{F}_2^m \to \mathbb{F}_2$ must be output by Algorithm 1. Conversely, taking $p(n) = n$, any vector output by Algorithm 1 is a linear structure of f except a negligible probability.*

Algorithm 1 Algorithm for finding linear structures of single-output functions

```
Input: quantum oracle of $f : \mathbb{F}_2^m \to \mathbb{F}_2$, a polynomial $p(n)$.
Output: linear structures of $f$.
 1: Define a set $T := \Phi$;
 2: for $p = 1, 2, \cdots, p(n)$ do
 3:     Execute BV algorithm on $f$, obtaining a vector $\omega$;
 4:     Let $T = T \cup \{\omega\}$;
 5: end for
 6: Solve the equation $\{x \cdot \omega = i | \omega \in T\}$ and obtain two solution sets $C^i$ for both
    $i = 0, 1$;
 7: if $C^0 \cup C^1 \subseteq \{0^m\}$ then
 8:     Output "Not linear structure function";
 9: else
10:     Output $C^0$ and $C^1$;
11: end if
```

2.4. Grover's Algorithm

Grover's algorithm [2] was introduced for unstructured search. Suppose the set to be searched is $\mathbb{F}_2^n$, and $u \in \mathbb{F}_2^n$ is the target vector. In a classical setting, it takes a time of 2^n to find u, while in a quantum setting, using Grover's algorithm only takes a time of $\sqrt{2^n}$. Grover's algorithm has three steps:

1. Prepare the quantum state
$$H^{(n)}|0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x \in \mathbb{F}_2^n} |x\rangle \triangleq |\phi\rangle$$
by applying Hadamard transform.
2. Construct the quantum oracle $\mathcal{O}_u$ of function
$$f_u(x) = \begin{cases} 1 & x = u \\ 0 & x \neq u. \end{cases}$$
$\mathcal{O}_u$ operates as
$$\mathcal{O}_u : |x\rangle \to (-1)^{f_u(x)}|x\rangle.$$
3. Let $\mathcal{O}_\phi = 2|\phi\rangle\langle\phi| - I$. Perform Grover's iteration $\mathcal{O}_\phi\mathcal{O}_u$ for $R \approx \frac{\pi}{4}\sqrt{2^n}$ times to obtain
$$(\mathcal{O}_\phi\mathcal{O}_u)^R|\phi\rangle \approx |u\rangle.$$
4. Return u.

When implementing Grover's algorithm, the quantum oracle $\mathcal{O}_{f_u}$ of the function f_u is given, which operates as follows:

$$\mathcal{O}_{f_u} : |x\rangle|y\rangle \to |x\rangle|y \oplus f_u(y)\rangle.$$

The oracle $\mathcal{O}_u$ can be constructed based on $\mathcal{O}_{f_u}$ as in Figure 4. Given the input state $|x\rangle$, $\mathcal{O}_u$ performs $\mathcal{O}_{f_u}$ on $|x\rangle$ and the auxiliary state $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Then, the whole quantum state is

$$\begin{aligned} &\mathcal{O}_{f_u}\left(|x\rangle\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \\ =&|x\rangle\frac{|0 \oplus f_u(x)\rangle - |1 \oplus f_u(x)\rangle}{\sqrt{2}} \\ =&(-1)^{f_u(x)}|x\rangle\frac{|0\rangle - |1\rangle}{\sqrt{2}}. \end{aligned}$$

The last equation holds because when $f_u(x) = 0$, the state of the second register remains unchanged, while when $f_u(x) = 1$, the state of the second register becomes $\frac{1}{\sqrt{2}}(|1\rangle - |0\rangle)$, which brings a negative sign. Then $\mathcal{O}_u$ discards the second register and outputs the state $(-1)^{f_u(x)}|x\rangle$ of the first register.

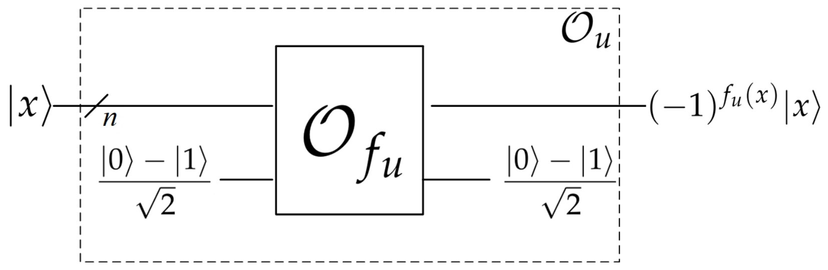
$\mathcal{O}_u$
$|x\rangle$
n
$\frac{|0\rangle - |1\rangle}{\sqrt{2}}$
$\mathcal{O}_{f_u}$
$\frac{|0\rangle - |1\rangle}{\sqrt{2}}$
$(-1)^{f_u(x)}|x\rangle$

Figure 4. The construction of $\mathcal{O}_u$.

3. Quantum Distinguisher of MIBS

One of the common methods for attacking block ciphers by quantum algorithms is to first construct a quantum distinguisher by Simon's algorithm or the BV algorithm, and then utilize Grover's algorithm to extract the correct key based on the constructed distinguisher. Specifically, Simon's algorithm can quickly determine whether a function is a periodic function. To obtain a distinguisher, the attacker first constructs a periodic function by using part of the encryption algorithm. Then, when the queried oracle is the block cipher, implementing Simon's algorithm on the constructed function should output a period. When the queried oracle is a random function, applying Simon's algorithm on the constructed function outputs a nonzero period with a negligible probability. Based on this significant difference, the attacker can distinguish between the block cipher and random function. In the phase of key recovery, the attacker guesses the round keys of several rounds after the distinguisher and uses the guessed keys to decrypt the ciphertexts obtained by querying. If the guessed round keys are correct, then the distinguisher performed on the partly decrypted ciphertexts should identify them as the outputs of a block cipher. If the round keys are incorrect, the partly decrypted ciphertexts are equivalent to the outputs of a random function. Thus, the distinguisher should identify them as outputs of a random function. By traversing all possible round keys, the attacker can recognize the correct key. In this process, Grover's algorithm can provide speedup. This attack strategy is called "Grover-meet-Simon" [8–10].

Similar to Grover-meet-Simon, the strategy "Grover-meet-BV" is also used [15]. In a Grover-meet-BV attack, the attacker constructs a linear-structure function instead of a periodic function and uses the BV algorithm to distinguish functions with nonzero linear structures from random functions, instead of using Simon's algorithm to distinguish functions with nonzero periods from random functions. Except for this point, other parts of these two attacks are the same. According to this attack strategy, we first construct a linear-structure function based on a 5-round encryption of MIBS; then, we present a 5-round quantum distinguisher of MIBS using this linear structure function and BV algorithm.

3.1. A Linear-Structure Function Based on 5-Round MIBS

In this subsection, we construct a linear-structure function based on 5-round MIBS. For the convenience of derivation, let F_i be the F transformation in the i-th round, and let the S layer in the i-th round be S_i, as presented in Figure 5. All F_is (S_i's) operate in the same way. The left-branch input in the i-th round is L_{i-1}, and the right branch is R_{i-1}.

According to Figure 5, it holds that

$$R_5 = R_3 \oplus F_4(L_3). \tag{3}$$

Select two arbitrary constant vectors $\delta_0, \delta_1 \in \mathbb{F}_2^4$ such that $\delta_0 \neq \delta_1$. For any variables $d \in \mathbb{F}_2$ and $x = (x_4, x_3, x_2, x_1) \in \mathbb{F}_2^4$, let the input of the 5-round MIBS be

$$\begin{aligned} L_0 &= (0^4, \delta_d, 0^4, 0^4, 0^4, 0^4, 0^4, 0^4), \\ R_0 &= PM(0^4, x, 0^4, 0^4, 0^4, 0^4, 0^4, 0^4), \end{aligned}$$

where $0^4 = (0,0,0,0)$, $\delta_d = \delta_0$ when $d = 0$, and $\delta_d = \delta_1$ when $d = 1$. PM is defined as in Equation (1). We use R_5 to construct a linear-structure function. Due to Equation (3), for computing R_5, we should first compute R_3 and L_3. Let $K_i[j]$ be the j-th nibble of the i-th round key K_i. That is,

$$K_i = (K_i[8], K_i[7], K_i[6], K_i[5], K_i[4], K_i[3], K_i[2], K_i[1]), \;\; i = 1, 2, \cdots, 32.$$

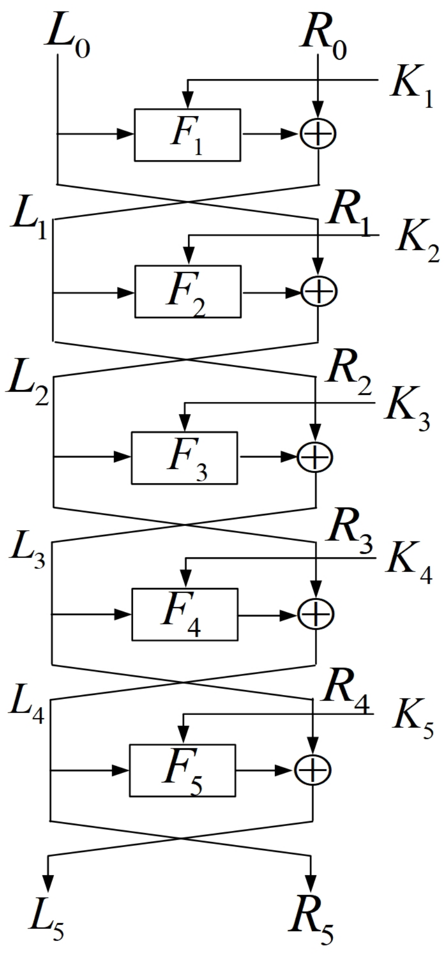

Figure 5. Illustration of 5-round distinguisher.

It holds that

$$\begin{aligned}&F_1(L_0 \oplus K_1)\\ =&F_1(K_1[8], \delta_d \oplus K_1[7], K_1[6], K_1[5], K_1[4], K_1[3], K_1[2], K_1[1])\\ =&PM \circ S_1(K_1[8], \delta_d \oplus K_1[7], K_1[6], K_1[5], K_1[4], K_1[3], K_1[2], K_1[1])\\ =&PM(s(K_1[8]), s(\delta_d \oplus K_1[7]), s(K_1[6]), s(K_1[5]), s(K_1[4]), s(K_1[3]), s(K_1[2]), s(K_1[1]))\\ =&PM(\mathcal{C}, \Delta_d, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}),\end{aligned}$$

where $\Delta_d = s(\delta_d \oplus K_1[7])$, $K_1[7]$ is a constant, and the notation $\mathcal{C}$ indicates that the corresponding nibble is a constant. The values of different nibbles marked with $\mathcal{C}$ may be different, but they are all restricted to constants that do not depend on variables x and d. Then,

$$\begin{aligned}L_1 &= F_1(L_0 \oplus K_1) \oplus R_0\\ &= PM(\mathcal{C}, \Delta_d, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}) \oplus PM(0^4, x, 0^4, 0^4, 0^4, 0^4, 0^4, 0^4)\\ &= PM(\mathcal{C}, \Delta_d \oplus x, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}).\end{aligned}$$

Therefore,

$$\begin{aligned} R_3 = L_2 &= R_1 \oplus F_2(L_1 \oplus K_2) \\ &= L_0 \oplus F_2(L_1 \oplus K_2) \\ &= L_0 \oplus F_2\big(PM(\mathcal{C},\Delta_d \oplus x,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2\big) \\ &= L_0 \oplus F_2\big(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2\big), \end{aligned} \tag{4}$$

where $* = \Delta_d \oplus x$. Since

$$\begin{aligned} L_2 &= F_2(L_1 \oplus K_2) \oplus R_1 \\ &= F_2\big(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2\big) \oplus R_1 \\ &= F_2\big(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2\big) \oplus L_0, \end{aligned}$$

we have

$$\begin{aligned} L_3 &= F_3(L_2 \oplus K_3) \oplus L_1 \\ &= F_3\Big(F_2\big(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2\big) \oplus L_0 \oplus K_3\Big) \oplus L_1. \end{aligned}$$

Before further deriving the linear-structure function, we first give Lemma 1.

Lemma 1. *Let* $g(*,\delta_d) = F_3\big(F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2) \oplus L_0 \oplus K_3\big)$*, where* $L_0 = (0^4,\delta_d,0^4,0^4,0^4,0^4,0^4,0^4)$ *and* $* = s(\delta_d \oplus K_1[7]) \oplus x$*; then, the value of the 5th nibble* $g(*,\delta_d)[5]$ *of* $g(*,\delta_d)$ *is only related to the value of* $*$*.*

Proof. According to the construction of the PM transformation,

$$\begin{aligned} &PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2 \\ =&(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*)) \oplus K_2 \\ =&(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*)), \end{aligned}$$

where $\mathcal{C}$ indicates that the corresponding nibble is a constant, and $\hbar(*)$ indicates that the value of the corresponding nibble is a function of $*$. Different nibbles marked with $\hbar(*)$ may be different functions of $*$, but their values are all restricted to only depend on the variable $*$. Notations $\mathcal{C}$ and $\hbar(*)$ are used to indicate the state of the corresponding nibbles, not a specific vector or function. The last equality holds since K_2 is a constant. Then,

$$\begin{aligned} &F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2) \\ =&PM \circ S_2(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*)) \\ =&PM(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*)) \\ =&(\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*)). \end{aligned}$$

Then,

$$\begin{aligned} &F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C}) \oplus K_2) \oplus L_0 \oplus K_3 \\ =&(\hbar(*),\hbar(*) \oplus \delta_d,\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*)) \end{aligned}$$

Therefore,

$$\begin{aligned}g(*,\delta_d) =&\big(g(*,\delta_d)[8],g(*,\delta_d)[7],\cdots,g(*,\delta_d)[2],g(*,\delta_d)[1]\big)\\ =&F_3(\hbar(*),\hbar(*)\oplus\delta_d,\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*))\\ =&PM\circ S_3(\hbar(*),\hbar(*)\oplus\delta_d,\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*))\\ =&PM(\hbar(*),?,\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*),\hbar(*))\\ =&(?,?,\hbar(*),\hbar(*),?,\hbar(*),?,?),\end{aligned}$$

where "?" means that the state of corresponding nibbles is uncertain. Due to the above equation, the 5th nibble $g(*,\delta_d)[5]$ of $g(*,\delta_d)$ is in the state $\hbar(*)$; thus, its values are only related to $*$. The notation $\hbar(*)$ means that every nibble marked with $\hbar(*)$ is a function of $*$. The $\hbar(*)$ does not refer to a specific function, but rather indicates that the values of the corresponding nibbles depend only on the value of $*$. It represents a kind of state of nibbles rather than a specific function. The $\hbar(*)$ symbols in the 2nd, 3rd, and 5th nibbles of $g(*,\delta_d)$ indicate that these three nibbles are all functions of $*$, and their values depend only on $*$, but they are not necessarily the same function. □

According to Lemma 1, we define the function

$$\begin{aligned}G:\mathbb{F}_2\times\mathbb{F}_2^4&\to\mathbb{F}_2^4\\ (d,x)&\to PM^{-1}(L_0\oplus R_5)[5]\oplus d^4,\end{aligned}$$

where

$$\begin{aligned}L_0&=(0^4,\delta_d,0^4,0^4,0^4,0^4,0^4,0^4),\\ R_0&=PM(0^4,x,0^4,0^4,0^4,0^4,0^4,0^4),\end{aligned}$$

and $(L_5,R_5)=MIBS^5(L_0,R_0)$, i.e., the ciphertext after a 5-round encryption of MIBS. PM^{-1} is the inverse function of PM. $PM^{-1}(L_0\oplus R_5)[5]$ is the 5th nibble of $PM^{-1}(L_0\oplus R_5)$. $d^4=(dddd)$.

Theorem 2. *Function $G(d,x)$ is a linear-structure function, and $(1,\Delta_0\oplus\Delta_1)$ is its linear structure. Specifically,*

$$G(d,x)\oplus G(d\oplus 1,x\oplus\Delta_0\oplus\Delta_1)=(1111),$$

where $\Delta_0=s(\delta_0\oplus K_1[7])$ and $\Delta_1=s(\delta_1\oplus K_1[7])$ are constants.

Proof. According to Equation (3), it holds that

$$PM^{-1}(L_0\oplus R_5)=PM^{-1}(L_0)\oplus PM^{-1}(R_3)\oplus PM^{-1}(F_4(L_3)).$$

Due to Equation (4), we have

$$PM^{-1}(R_3)=PM^{-1}(L_0)\oplus PM^{-1}(F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C})\oplus K_2)).$$

Therefore,

$$PM^{-1}(L_0\oplus R_5)=PM^{-1}(F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C})\oplus K_2))\oplus PM^{-1}(F_4(L_3)). \quad (5)$$

We compute the first part:

$$\begin{aligned}&PM^{-1}(F_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C})\oplus K_2))\\ =&S_2(PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C})\oplus K_2))\\ =&S_2(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*))\\ =&(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*)).\end{aligned}$$

Then, we compute the second part:

$$\begin{aligned}
&PM^{-1}(F_4(L_3))\\
=&S_4(g(*,\delta_d)\oplus L_1)\\
=&S_4\big(g(*,\delta_d)\oplus PM(\mathcal{C},*,\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C},\mathcal{C})\big)\\
=&S_4\big((?,?,\hbar(*),\hbar(*),?,\hbar(*),?,?)\oplus(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*))\big)\\
=&(?,?,\hbar(*),\hbar(*),?,\hbar(*),?,?).
\end{aligned}$$

Combining these two parts gives

$$\begin{aligned}
&PM^{-1}(L_0\oplus R_5) && (6)\\
=&(\hbar(*),\hbar(*),\mathcal{C},\mathcal{C},\hbar(*),\mathcal{C},\hbar(*),\hbar(*))\oplus(?,?,\hbar(*),\hbar(*),?,\hbar(*),?,?) && (7)\\
=&(?,?,\hbar(*),\hbar(*),?,\hbar(*),?,?). && (8)
\end{aligned}$$

Thus, we have

$$G(d,x)=PM^{-1}(L_0\oplus R_5)[5]\oplus d^4=\hbar(*)\oplus d^4=\hbar(\Delta_d\oplus x)\oplus d^4$$

and

$$G(d\oplus 1,x\oplus\Delta_0\oplus\Delta_1)=\hbar(\Delta_d\oplus x)\oplus d^4\oplus(1111).$$

These two equations mean that

$$G(d,x)\oplus G(d\oplus 1,x\oplus\Delta_0\oplus\Delta_1)=(1111).$$

which indicates the conclusion. □

3.2. 5-Round Quantum Distinguisher

We have constructed a linear-structure function based on the 5-round encryption of MIBS. Combining this with the quantum algorithm, which can determine whether a function is a linear-structure function, we can obtain a quantum distinguisher of 5-round MIBS.

Based on Algorithm 1, Xie and Yang constructed a quantum algorithm that can determine whether a function has linear structures [14]. We present this quantum algorithm below.

Theorem 3 ([14])**.** *If $f:\mathbb{F}_2^m\to\mathbb{F}_2^n$ is a linear-structure function, then except with a negligible probability, Algorithm 2 on f with $p(n)=n$ will output a linear structure of f.*

Algorithm 2 [14] Algorithm for finding linear structures of multiple-output functions

Input: quantum oracle of $f:\mathbb{F}_2^m\to\mathbb{F}_2^n$, a polynomial $p(n)$. ($f=(f_1,f_2,\cdots,f_n)$.)
Output: a linear structure of f.

```
1: for $j=1,2,\cdots,n$ do
2:     Run Algorithm 1 on $f_j$ with $p(n)$;
3:     if Algorithm 1 returns "Not linear structure function" then
4:         Output "Not linear structure function";
5:     else
6:         Let $C_j=C_j^0\cup C_j^1$, where $C_j^0$ and $C_j^1$ are the outputs of Algorithm 1;
7:     end if
8: end for
9: if $C_1\cap C_2\cap\cdots\cap C_n\subseteq\{0^m\}$ then
10:     Return "Not linear structure function";
11: else
12:     Randomly choose a nonzero vector $\alpha\in C_1\cap\cdots\cap C_n$ and return $\alpha$;
13: end if
```

In a quantum distinguishing attack on 5-round MIBS, a quantum oracle $\mathcal{O}$ is available, which implements either the encryption of 5-round MIBS $MIBS^5$ or a random function RF. Since the linear-structure function G is defined only using the right part R_5 of the output of $MIBS^5$, it is assumed that the attacker can query the oracle, which merely returns the right branch R_5. Such assumption is commonly used in quantum distinguishing attacks [4–6]. Moreover, in the following key-recovery attack, we show how to construct such oracle via the oracle of complete encryption. Thus, the attacker can query $\mathcal{O}$ by implementing either the operator

$$\mathcal{O}:\sum_{\substack{u\in\mathbb{F}_2^{64}\\ v\in\mathbb{F}_2^{32}}}|u\rangle|v\rangle\rightarrow\sum_{\substack{u\in\mathbb{F}_2^{64}\\ v\in\mathbb{F}_2^{32}}}|u\rangle|v\oplus MIBS_R^5(u)\rangle$$

or the operator

$$\mathcal{O}:\sum_{\substack{u\in\mathbb{F}_2^{64}\\ v\in\mathbb{F}_2^{32}}}|u\rangle|v\rangle\rightarrow\sum_{\substack{u\in\mathbb{F}_2^{64}\\ v\in\mathbb{F}_2^{32}}}|u\rangle|v\oplus PF(u)\rangle,$$

where $MIBS_R^5:\mathbb{F}_2^{64}\rightarrow\mathbb{F}_2^{32}$ is the the encryption function of 5-round MIBS, which only returns the right 32 bits, and $PF:\mathbb{F}_2^{64}\rightarrow\mathbb{F}_2^{32}$ is a random function.

A quantum distinguisher of $\mathcal{O}$ is a quantum algorithm that can distinguish whether $\mathcal{O}$ implements $MIBS_R^5$ or a random function PF. In order to construct a quantum distinguisher, an intuitive idea is to construct the oracle $\mathcal{O}_G$ of function G based on $\mathcal{O}$

$$\mathcal{O}_G:\sum_{\substack{(d,x)\in\mathbb{F}_2\times\mathbb{F}_2^4,\\ y\in\mathbb{F}_2^4}}|d\rangle|x\rangle|y\rangle\rightarrow\sum_{\substack{(d,x)\in\mathbb{F}_2\times\mathbb{F}_2^4,\\ y\in\mathbb{F}_2^4}}|d\rangle|x\rangle|y\oplus G(d,x)\rangle,$$

and then run Algorithm 2 on $\mathcal{O}_G$ to determine whether it is the oracle of a linear-structure function, thereby determining whether $\mathcal{O}$ is the encryption of 5-round MIBS. If $\mathcal{O}$ implements $MIBS^5$, then Algorithm 2 will return a linear structure of G; otherwise, if $\mathcal{O}$ implements RF, then its probability of outputting a linear structure of G is negligible.

Figure 6 shows how to construct the oracle $\mathcal{O}_G$ of function G based on $\mathcal{O}$. The unitary operator $CNOT^{(32)}$ is composed of 32 $CNOT$ gates and works as follows:

$$CNOT^{(32)}:\sum_{u,v\in\mathbb{F}_2^{32}}|u\rangle|v\rangle\rightarrow\sum_{u,v\in\mathbb{F}_2^{32}}|u\rangle|v\oplus u\rangle.$$

Similarly, $CNOT^{(4)}$ is composed of 4 $CNOT$ gates and works as follows:

$$CNOT^{(4)}:\sum_{\substack{u_8,u_7,\cdots,u_1\in\mathbb{F}_2^4\\ v\in\mathbb{F}_2^4}}|u_8,u_7,\cdots,u_1\rangle|v\rangle\rightarrow\sum_{\substack{u_8,u_7,\cdots,u_1\in\mathbb{F}_2^4\\ v\in\mathbb{F}_2^4}}|u_8,u_7,\cdots,u_1\rangle|v\oplus u_5\rangle.$$

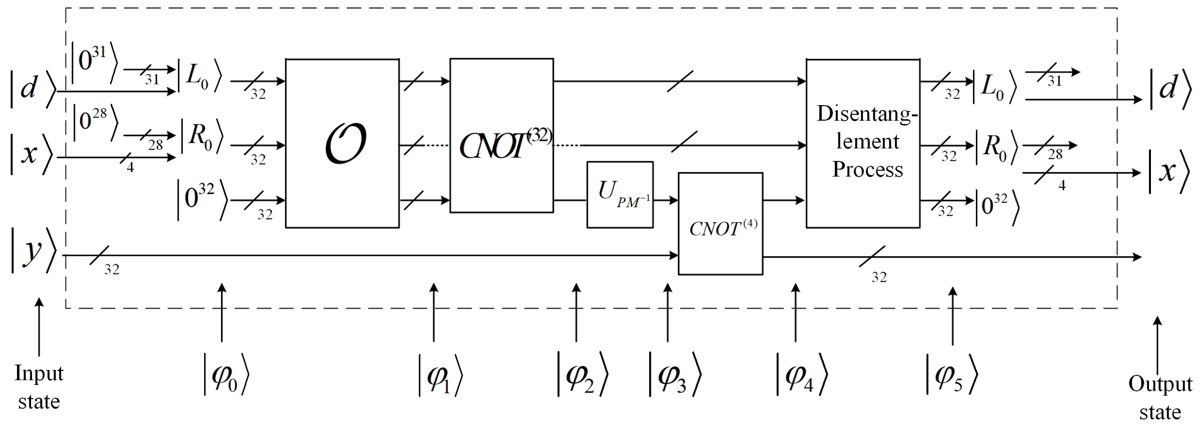

Figure 6. Construction of $\mathcal{O}_G$.

The unitary operator $U_{PM^{-1}}$ is defined as

$$U_{PM^{-1}}: \sum_{u\in\mathbb{F}_2^{32}} |u\rangle \to \sum_{u\in\mathbb{F}_2^{32}} |PM^{-1}(u)\rangle$$

and can be realized as shown in Figure 7. The input $|d\rangle$ of $\mathcal{O}_G$ is combined with 31 auxiliary states $|0\rangle$ to form the state $|L_0\rangle$, and the input $|x\rangle$ is combined with 28 auxiliary states $|0\rangle$ to form the state $|R_0\rangle$. The states in Figure 6 are defined as below.

$$\begin{aligned}
\text{Input state} &= \sum_{\substack{(d,x)\in\mathbb{F}_2\times\mathbb{F}_2^4,\\ y\in\mathbb{F}_2^4}} |d\rangle|x\rangle|y\rangle,\\
|\varphi_0\rangle &= \sum_{\substack{(d,x)\in\mathbb{F}_2\times\mathbb{F}_2^4,\\ y\in\mathbb{F}_2^4}} |0^4\rangle|\delta_d\rangle|0^4\rangle\cdots|0^4\rangle|0^4\rangle|x\rangle|0^4\rangle\cdots|0^4\rangle|0^{32}\rangle|y\rangle\\
&= \sum_{\substack{(d,x)\in\mathbb{F}_2\times\mathbb{F}_2^4,\\ y\in\mathbb{F}_2^4}} |L_0\rangle|R_0\rangle|0^{32}\rangle|y\rangle,\\
|\varphi_1\rangle &= \begin{cases} \sum_{d,x,y}|L_0\rangle|R_0\rangle|MIBS_R^5(L_0,R_0)\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } MIBS_R^5\\ \sum_{d,x,y}|L_0\rangle|R_0\rangle|RF(L_0,R_0)\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } RF\end{cases}\\
|\varphi_2\rangle &= \begin{cases} \sum_{d,x,y}|L_0\rangle|R_0\rangle|R_5\oplus L_0\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } MIBS_R^5\\ \sum_{d,x,y}|L_0\rangle|R_0\rangle|RF(L_0,R_0)\oplus L_0\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } RF\end{cases}\\
|\varphi_3\rangle &= \begin{cases} \sum_{d,x,y}|L_0\rangle|R_0\rangle|PM^{-1}(R_5\oplus L_0)\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } MIBS_R^5\\ \sum_{d,x,y}|L_0\rangle|R_0\rangle|PM^{-1}(RF(L_0,R_0)\oplus L_0)\rangle|y\rangle, & \text{if } \mathcal{O} \text{ is oracle of } RF\end{cases}
\end{aligned}$$

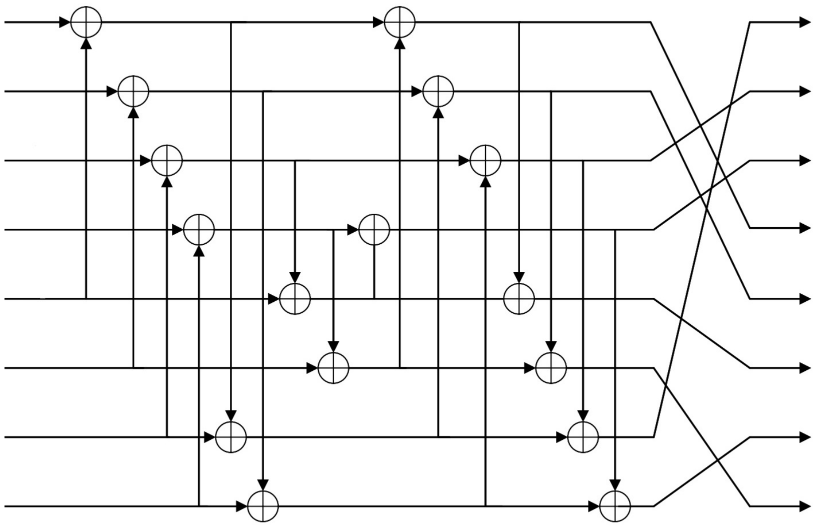

Figure 7. Quantum circuit of $U_{PM^{-1}}$.

Therefore, when $\mathcal{O}$ is the oracle of $MIBS_R^5$, it holds that

$$\begin{aligned}
|\varphi_4\rangle &= \sum_{d,x,y}|L_0\rangle|R_0\rangle|PM^{-1}(R_5\oplus L_0)\rangle|y\oplus PM^{-1}(R_5\oplus L_0)[5]\rangle\\
&= \sum_{d,x,y}|L_0\rangle|R_0\rangle|PM^{-1}(R_5\oplus L_0)\rangle|y\oplus G(d,x)\rangle.
\end{aligned}$$

When $\mathcal{O}$ is the oracle of random function RF, it holds that

$$|\varphi_4\rangle = \sum_{d,x,y}|L_0\rangle|R_0\rangle|PM^{-1}(RF(L_0,R_0)\oplus L_0)\rangle|y\oplus PM^{-1}(RF(L_0,R_0)\oplus L_0)[5]\rangle.$$

The disentanglement process is to disentangle the registers denoted $|d\rangle$, $|x\rangle$, and $|y\rangle$ from the registers of the auxiliary states. Thus, after this process, the state will be

$$|\varphi_5\rangle = \sum_{d,x,y} |L_0\rangle|R_0\rangle|0^{32}\rangle|y \oplus G(d,x)\rangle$$

if $\mathcal{O}$ is the oracle of $MIBS_R^5$, or

$$|\varphi_5\rangle = \sum_{d,x,y} |L_0\rangle|R_0\rangle|0^{32}\rangle|y \oplus PM^{-1}(RF(L_0,R_0) \oplus L_0)[5]\rangle \tag{9}$$

if $\mathcal{O}$ is the oracle of random function RF. Since RF is a random function from 64 bits to 32 bits, $PM^{-1}(RF(L_0,R_0) \oplus L_0)[5]$ can also been seen as a random function mapping 5 bits to 4 bits given input (d,x). Let $RF_{5,4}$ denote the random function from 5 bits to 4 bits; then, we have

$$|\varphi_5\rangle = \sum_{d,x,y} |L_0\rangle|R_0\rangle|0^{32}\rangle|y \oplus RF_{5,4}(d,x)\rangle \tag{10}$$

when $\mathcal{O}$ is the oracle of RF. The output state of $\mathcal{O}_G$ shown in Figure 6 is

$$\text{output state} = \begin{cases} \sum_{d,x,y} |d\rangle|x\rangle|y \oplus G(d,x)\rangle, & \text{if } \mathcal{O} \text{ is oracle of } MIBS_R^5 \\ \sum_{d,x,y} |d\rangle|x\rangle|y \oplus RF_{5,4}(d,x)\rangle, & \text{if } \mathcal{O} \text{ is oracle of } RF. \end{cases}$$

The quantum oracle $\mathcal{O}_G$ has been constructed; then, we present the quantum distinguisher of 5-round MIBS. Given the access to the oracle $\mathcal{O}$, the distinguisher $\mathcal{D}^{\mathcal{O}}$ works as follows:

(1) Construct the oracle $\mathcal{O}_G$ based on $\mathcal{O}$ as in Figure 6;
(2) Implement Algorithm 2 using oracle $\mathcal{O}_G$;
(3) If Algorithm 2 returns a linear structure, output $|1\rangle$; otherwise, output $|0\rangle$.

The output $|1\rangle$ indicates that $\mathcal{O}$ is the oracle of $MIBS_R^5$, and output $|0\rangle$ indicates that $\mathcal{O}$ is the oracle of random function RF. According to Theorem 2, $\mathcal{D}^{\mathcal{O}}$ can correctly distinguish the 5-round MIBS from a random function.

4. Key-Recovery Attack

We first give a 7-round key-recovery attack on MIBS utilizing the distinguisher proposed in Section 3.2. We consider a chosen plaintext attack, where the oracle of the 7-round MIBS is available. Namely, the oracle

$$\mathcal{O}_{MIBS^7} : \sum_{u,v \in \mathbb{F}_2^{64}} |u\rangle|v\rangle \rightarrow \sum_{u,v \in \mathbb{F}_2^{64}} |u\rangle|v \oplus MIBS^7(u)\rangle$$

can be queried by an attacker. Through querying $\mathcal{O}_{MIBS^7}$, the attacker can obtain the superposition state of the ciphertexts after 7-round encryption; then, the attacker guesses the relevant bits of the 6th- and 7th-round keys and decrypts the ciphertexts for two rounds to obtain the ciphertexts of $MIBS_R^5$ (R_5). Therefore, for each guessed candidate round key of the 6th and 7th rounds, the attacker can use it to decrypt 2 rounds to obtain oracle $\mathcal{O}$, which is the oracle of $MIBS_R^5$ when the guessed key is right, and is the oracle of the random function RF when the guessed key is wrong. Using the distinguisher $\mathcal{D}^{\mathcal{O}}$ defined in Section 3.2 with queries to $\mathcal{O}$ can determine whether the guessed round key bits are right. If the round key bits are right, $\mathcal{D}^{\mathcal{O}}$ will output $|1\rangle$; otherwise, it will output $|0\rangle$.

The key is how to compute R_5 using the least bits of K_6 and K_7 given the ciphertext (L_7, R_7). Since

$$\begin{aligned} & g(d,x) \\ =& PM^{-1}(MIBS_R^5(L_0,R_0) \oplus L_0)[5] \\ =& PM^{-1}(R_5)[5] \oplus PM^{-1}(L_0)[5], \end{aligned}$$

to construct oracle $\mathcal{O}_G$, we actually only need to compute the 5th nibble of $PM^{-1}(R_5)$ instead of the entire R_5. Therefore, we can slightly change the way to generate $\mathcal{O}_G$ so that we do not need the entire R_5. $\mathcal{O}_G$ can still be constructed from $\mathcal{O}$ using the method in Section 3.2, except that $\mathcal{O}$ is no longer the oracle of the entire R_5, but only the part of R_5 required for computing $PM^{-1}(R_5)[5]$. This does not bring any essential differences but can void guessing the unnecessary key bits during key-recovery attack. As shown in Figure 8, it holds that

$$\begin{aligned} PM^{-1}(R_5)[5] &= PM^{-1}(F_6(L_5 \oplus K_6))[5] \oplus PM^{-1}(L_6)[5] \\ &= S_6(L_5 \oplus K_6)[5] \oplus PM^{-1}(R_7)[5] \\ &= s(L_5[5] \oplus K_6[5]) \oplus PM^{-1}(R_7)[5] \\ &= s(R_6[5] \oplus K_6[5]) \oplus PM^{-1}(R_7)[5] \\ &= s\big(F_7(R_7 \oplus K_7)[5] \oplus L_7[5] \oplus K_6[5]\big) \oplus PM^{-1}(R_7)[5] \\ &= s\Big(PM\big(S_7(R_7 \oplus K_7)\big)[5] \oplus L_7[5] \oplus K_6[5]\Big) \oplus PM^{-1}(R_7)[5]. \end{aligned}$$

Since L_7 and R_7 are known, according to the definition of PM, to compute $PM(S_7(R_7 \oplus K_7))[5]$, we only need to guess the 1st, 3rd, 4th, 5th, and 8th nibbles of K_7. Therefore, $K_6[5]$, $K_7[8,5,4,3,1]$ are enough for computing the value of $PM^{-1}(R_5)[5]$ or $g(d,x)$. There are 24 bits needed to be guessed. Considering the key scheduling, there may exist repetitive bits.

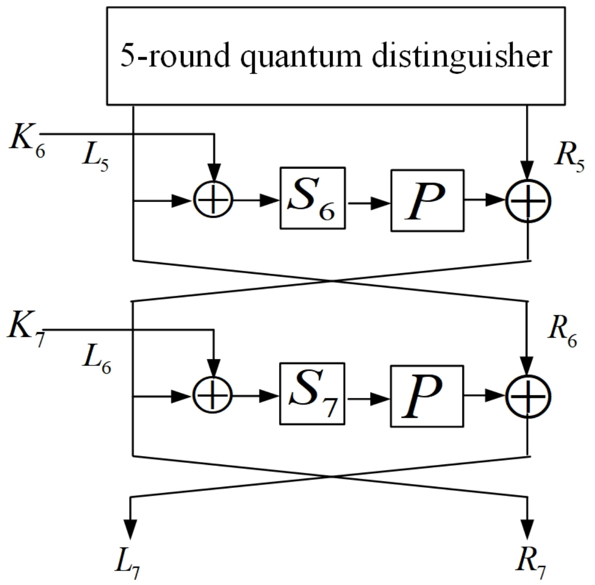

Figure 8. Illustration of key-recovery attack.

Table 3 shows the repetition bits of the subkeys in 7–10 rounds generated as the key scheduling. Suppose the state of the key register in the 6th round of key scheduling is

$$state = a_{63}a_{62}\cdots a_1a_0,$$

then, $K_6 = a_{63}a_{62}\cdots a_{32}$ and $K_7 = a_{14}a_{13}\cdots a_0 a_{63}\cdots a_{47}$. Here, we omit the Sbox transformation since the determined transformation does not affect the amount of bits that is required to be guessed. According to Table 3, the 2nd, 3rd, and 4th bits of $K_7[1]$ are the same as the 1st, 2nd, and 3rd bits of $K_6[5]$. Thus, in fact, we only need to guess 21 key bits:

$$K_6[5], K_7[8,5,4,3,1(1)],$$

where $K_7[8,5,4,3,1(1)]$ denotes the 8th, 5th, 4th, and 3rd nibbles and the 1st bit of the 1st nibble of K_7.

Table 3. Repetition of round key bits.

K_6	63	62	61	60	59	58	57	56	55	54	53	52	51	**50**	**49**	**48**
	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
K_7	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	63
	62	61	60	59	58	57	56	55	54	53	52	51	**50**	**49**	**48**	47
K_8	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	**14**
	13	**12**	**11**	10	9	8	7	6	5	4	3	**2**	**1**	**0**	**63**	**62**
K_9	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	**29**
	28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**
K_{10}	**59**	**58**	**57**	**56**	**55**	54	53	52	**51**	**50**	**49**	**48**	**47**	46	45	**44**
	43	**42**	**41**	**40**	**39**	**38**	**37**	**36**	**35**	**34**	**33**	**32**	**31**	**30**	**29**	**28**

Define

$$\begin{aligned}\overline{G}: \mathbb{F}_2^{21} \times \mathbb{F}_2 \times \mathbb{F}_2^4 &\to \mathbb{F}_2^4 \\ (K^{6,7}, d, x) &\to G(d,x) \text{ under the decryption of } K^{6,7}.\end{aligned}$$

Given the oracle of $MIBS^7$, by decrypting under the relevant keys in the 6th and 7th rounds, it is easy to obtain the oracle of $PM^{-1}(R_5)[5]$. Then, we can construct the oracle of $\overline{G}$ using a similar method in Section 3.2. The oracle of $\overline{G}$ can play the role of $\mathcal{O}_G$ in the distinguisher $\mathcal{A}$. Thus, $\mathcal{A}$ will output $|1\rangle$ when $K^{6,7}$ is the correct 21-bit key and output $|0\rangle$ when $K^{6,7}$ is the wrong 21-bit subkey. Taking $\mathcal{A}$ as the oracle $\mathcal{O}_u$ in Grover's algorithm, it will search for the right key bits: $K_6[5], K_7[8,5,4,3,1(1)]$.

According to [8], this key-recovery attack needs a total of

$$n_k + n_{in} \times l + n_{out} \times l \tag{11}$$

qubits, where n_k is the number of bits of the subkeys to be recovered, n_{in} is the input length of the linear structure function G, n_{out} is the output length of G, and $l = 2(n_{in} + \sqrt{n_{in}})$. $n_k = 21$, $n_{in} = 5$, $n_{out} = 4$, and $l \approx 15$. Thus, this attack requires 156 qubits. The time complexity is $\sqrt{2^{21}} = 2^{10.5}$.

Consider attacking 8-round MIBS using the same distinguisher. By similar derivation, to compute $PM^{-1}(R_5)[5]$ based on the ciphertext (L_8, R_8), we need to guess the subkeys

$$K_6[5], K_7[8,5,4,3,1(1)], K_8.$$

According to Table 3, K_8 has 9 repetitive bits, so there are only 44 bits to be recovered. According to Equation (11), an 8-round key-recovery attack requires 179 qubits. The corresponding time complexity is $\sqrt{2^{44}} = 2^{22}$.

By similar derivation, a 9-round attack requires 194 qubits, and the time complexity is $\sqrt{2^{59}} = 2^{29.5}$. A 10-round attack requires 199 qubits, and the time complexity is $\sqrt{2^{64}} = 2^{32}$. The authors in [25] also presented quantum attacks on MIBS. The time complexity of their 7-round, 8-round, and 9-round attacks is 2^{12}, 2^{28}, and 2^{44}, respectively. The complexity of our attacks proposed in this article is lower.

5. Results and Discussion

In this article, we proposed quantum attacks on the MIBS cipher based on the BV algorithm. Specifically, we first fully utilize the characteristics of the linear transformations of the MIBS cipher to construct a linear-structure function. Then, we use the fact that the BV algorithm can quickly determine whether a function has nonzero linear structures to design a 5-round quantum distinguisher for the MIBS cipher, which can effectively distinguish the encryption of the 5-round MIBS cipher from a random function. Subsequently, by analyzing the key scheduling of the MIBS cipher, we find the repeated bits between round keys. Combined with Grover's algorithm, we realize a 7-round key-recovery attack on MIBS and generalize the attack to more rounds. The quantum attack on 7-round MIBS requires 156 qubits and has a time complexity of $2^{10.5}$. The 8-round attack requires 179 qubits, and the time complexity is 2^{22}. Compared with the existing quantum attacks, our attack has the smallest time complexity. We believe this study contributes to evaluating the safety of the MIBS cipher in the quantum world and helps to further explore the "BV-meet-Grover" attack strategy.

For further research, how to reduce the resource consumption and time complexity of the attacks on the MIBS cipher is worth studying. We can also study the applications of the BV algorithm and other quantum algorithms to key-recovery attacks on various block ciphers. Quantum attacks on other symmetric primitives, such as hash functions and stream ciphers, are also a meaningful direction. For example, we can apply the quantum attack strategies introduced in [20,21] to attack other hash functions [26,27]. We can also apply quantum algorithms to enhance the classical attacks on stream ciphers that have been proposed [28,29] or to attack other cryptographic schemes [30,31].

Author Contributions: Conceptualization, H.X. (Huiqin Xie), Z.Z. and H.X. (Huiqin Xie); Formal analysis, H.X. (Huiqin Xie), Z.Z., K.W. and Y.L.; Funding acquisition, H.X. (Huiqin Xie); Investigation, H.X. (Huiqin Xie), K.W. and H.X. (Huiqin Xie); Methodology, H.X. (Huiqin Xie) and Z.Z.; Validation, H.X. (Huiqin Xie) and Y.L.; Visualization, H.X. (Huiqin Xie) and H.X. (Hongcai Xin); Writing—original draft, H.X. (Huiqin Xie), Z.Z. and H.X. (Huiqin Xie); Writing—review and editing, H.X. (Huiqin Xie). All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Beijing Natural Science Foundation (no. 4234084), the Open Research Fund of Key Laboratory of Cryptography of Zhejiang Province (no. ZCL21012) and Fundamental Research Funds for the Central Universities (no. 328202202).

Data Availability Statement: The original contributions presented in this study are included in the article, and further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Shor, P.W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
2. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.
3. Simon, D.R. On the power of quantum computation. *SIAM J. Comput.* **1997**, *10*, 1474–1483. [CrossRef]
4. Kuwakado, H.; Morii, M. Quantum Distinguisher between the 3-Round Feistel Cipher and the Random Permutation. In Proceedings of the IEEE International Symposium on Information Theory, Austin, TX, USA, 13–18 June 2010; pp. 2682–2685.
5. Santoli, T.; Schaffner, C. Using Simon's algorithm to attack symmetric-key cryptographic primitives. *Quantum Inf. Comput.* **2017**, *17*, 65–78. [CrossRef]
6. Kaplan, M.; Leurent, G.; Leverrier, A.; Naya-Plasencia, M. Breaking Symmetric Cryptosystems Using Quantum Period Finding. In Proceedings of the CRYPTO'16, Barbara, CA, USA, 14–18 August 2016; pp. 207–237.
7. Kuwakado, H.; Morii, M. Security on the Quantum-type Even-Mansour Cipher. In Proceedings of the Information Theory and its Applications, Honolulu, HI, USA, 28–31 October 2012; pp. 312–316.
8. Leander, G.; May, A. Grover Meets Simon–Quantumly Attacking the FX-construction. In Proceedings of the ASIACRYPT'17, Hong Kong, China, 3–7 December 2017; pp. 161–178.
9. Dong, X.; Wang, X. Quantum key-recovery attack on Feistel structures. *Sci. China Inf. Sci.* **2018**, *10*, 240–246. [CrossRef]

10. Dong, X.; Wang, X. Quantum cryptanalysis on some generalized Feistel schemes. *Sci. China Inf. Sci.* **2019**, *62*, 22501:1–22501:12. [CrossRef]
11. Jaques, S.; Naehrig, M.; Roetteler, M.; Virdia, F. Implementing Grover Oracles for Quantum Key Search on AES and LowMC. In Proceedings of the EUROCRYPT'20, Zagreb, Croatia, 10–14 May 2020; pp. 280–310.
12. Halak, B.; Gibson, T.; Henley, M.; Botea, C.B.; Heath, B.; Khan, S. Evaluation of performance, energy, and computation costs of quantum-attack resilient encryption algorithms for embedded devices. *IEEE Access* **2024**, *12*, 8791–8805. [CrossRef]
13. Bernstein, E.; Vazirani, U. Quantum complexity theory. *SIAM J. Comput.* **1997**, *26*, 1411–1473. [CrossRef]
14. Xie, H.; Yang, L. Using Bernstein-Vazirani algorithm to attack block ciphers. *Des. Codes Cryptogr.* **2019**, *87*, 1161–1182. [CrossRef]
15. Zhou, B.; Yuan, Z. Quantum key-recovery attack on Feistel constructions: Bernstein-Vazirani meet Grover's algorithm. *Quantum Inf. Process.* **2021**, *20*, 330. [CrossRef]
16. Zhou, Q.; Lu, S.; Zhang, Z.; Sun, J. Quantum differential cryptanalysis. *Quantum Inf. Process.* **2015**, *14*, 2101–2109. [CrossRef]
17. Kaplan, M.; Leurent, G.; Leverrier, A.; Naya-Plasencia M. Quantum Differential and Linear Cryptanalysis. In Proceedings of the Fast Software Encryption, Bochum, Germany, 20–23 March 2016; pp. 71–94.
18. Li, H.; Yang, L. Quantum Differential Cryptanalysis to the Block Ciphers. In Proceedings of the 6th International Conference on Applications and Techniques in Information Security, Beijing, China, 4–6 November 2015; pp. 44–51.
19. Li, H.; Yang, L. A quantum algorithm to approximate the linear structures of Boolean functions. *Math. Struct. Comput. Sci* **2018**, *28*, 1–13. [CrossRef]
20. Hosoyamada A., Sasaki Y. Finding Hash Collisions with Quantum Computers by Using Differential Trails with Smaller Probability than Birthday Bound. In Proceedings of the EUROCRYPT'20, Zagreb, Croatia, 10–14 May 2020; pp. 249–279.
21. Dong, X.; Sun, S.; Shi, D.; Gao, F.; Wang, X.; Hu, L. Quantum Collision Attacks on AES-Like Hashing with Low Quantum Random Access Memories. In Proceedings of the ASIACRYPT'20, Daejeon, Republic of Korea, 7–11 November 2020; pp. 727–757.
22. Roetteler, M.; Steinwandt, R. A note on quantum related-key attacks. *Inf. Process. Lett.* **2015**, *115*, 40–44. [CrossRef]
23. Hosoyamada, A.; Aoki K. On quantum related-key attacks on iterated Even-Mansour ciphers. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2019**, *102*, 27–34. [CrossRef]
24. Izadi, M.; Sadeghiyan, B.; Sadeghian, S.S.; Khanooki, H.A. MIBS: A New Lightweight Block Cipher. In Proceedings of the International Conference on Cryptology and Network Security—CANS, Kanazawa, Japan, 12–14 December 2009; pp. 334–348.
25. Li, Y.; Lin, H.; Yi, Z.; Xie, H. Quantum Cryptanalysis of MIBS. *J. Cryptologic Res.* **2021**, *8*, 989–998.
26. Hannusch, C.; Horváth, G. Properties of Hash Functions based on Gluškov Product of Automata. *J. Autom. Lang. Comb.* **2021**, *26*, 55–65.
27. Grassi, L.; Khovratovich, D.; Rechberger, C.; Roy, A.; Schofnegger, M. Poseidon: A New Hash Function for Zero-Knowledge Proof Systems. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), Virtual Event, 11–13 August 2021; pp. 519–535.
28. Domosi, P.; Horvath, G.; Molnar, F. T.; Kovacs, S.; Diene, A. A side-channel attack against an automata theory based stream cipher. *Logic Lang. Algebr. Syst. Relat. Areas Comput. Sci.* **2021**, *2193*, 64–72.
29. Mascia, C.; Piccione, E.; Sala, M. An algebraic attack on stream ciphers with application to nonlinear filter generators and WG-PRNG. *Adv. Math. Commun.* **2024**, *18*, 1710–1722. [CrossRef]
30. Dömösi, P.; Hannusch, C.; Horváth, G. A cryptographic system based on a new class of binary error-correcting codes. *Tatra Mt. Math. Publ.* **2019**, *73*, 83–96.
31. Drăgoi, V. F.; Szocs, A. Structural Properties of Self-dual Monomial Codes with Application to Code-Based Cryptography. In Proceedings of the 18th IMA International Conference on Cryptography and Coding, Virtual Event, 14–15 December 2021; pp. 16–41.