mathematics

MDPI

Article

Quantum Chosen-Cipher Attack on Camellia

Yanjun Li [1,*], Qi Wang [2], Dingyun Huang [2], Jian Liu [1] and Huiqin Xie [2]

[1] Infomation Industry lnformation Security Evaluation Center, The 15th Research Institute of China Electronics Technology Group Corporation, Beijing 100083, China
[2] Department of Cryptography Science and Technology, Beijing Electronic Science and Technology Institute, Beijing 100070, China; wq58416562@163.com (Q.W.); huangdingyun1213@163.com (D.H.)
* Correspondence: liyjwuyh@163.com

Abstract: The Feistel structure represents a fundamental architectural component within the domain of symmetric cryptographic algorithms, with a substantial body of research conducted within the context of classical computing environments. Nevertheless, research into specific symmetric cryptographic algorithms utilizing the Feistel structure is relatively scarce in quantum computing environments. This paper, for the first time, proposes a five-round distinguisher for Camellia under the quantum chosen-ciphertext attack (qCCA) setting, with its effectiveness empirically validated. Additionally, by combining Grover's algorithm and Simon's algorithm, we construct a nine-round key-recovery attack model against Camellia. Through an in-depth analysis of Camellia's key expansion algorithm, we significantly reduce the complexity of the key-recovery attack. The proposed attack achieves a time complexity of $2^{61.5}$ for recovering the correct key bits and requires 531 quantum bits.

Keywords: Feistel cipher; quantum chosen-ciphertext attacks; Grover's algorithm; Simon's algorithm; Camellia

MSC: 94A60; 81P94; 68Q12; 68W40; 68P25

check for updates

Academic Editor: Antanas Cenys

Received: 14 March 2025
Revised: 14 April 2025
Accepted: 22 April 2025
Published: 24 April 2025

Citation: Li, Y.; Wang, Q.; Huang, D.; Liu, J.; Xie, H. Quantum Chosen-Cipher Attack on Camellia. *Mathematics* **2025**, *13*, 1383. https://doi.org/10.3390/math13091383

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Quantum computing has the potential to address problems that are intractable for classical computing, particularly in the field of cryptanalysis. For certain complex problems, such as large-scale search or the identification of specific structures, the computational complexity on classical computers often grows exponentially. In contrast, quantum computing leverages quantum superposition and quantum parallelism to significantly improve computational efficiency. For example, Shor's quantum algorithm [1] can break the classical RSA public-key encryption system in polynomial time, while Grover's algorithm [2] reduces the time complexity of unstructured database search from the classical $O(2^n)$ to the quantum $O(2^{n/2})$, posing a direct threat to the key search process of symmetric encryption algorithms. Similarly, Simon's algorithm [3] can effectively break cryptographic schemes with specific structural properties by identifying the periodicity of certain functions. In addition, recent studies in the field of quantum computing have demonstrated that its applications have expanded to advanced domains such as secure communication and machine learning. For example, Zhou et al. [4] proposed a multi-party semi-quantum private comparison protocol based on d-dimensional GHZ states, which highlights significant advancements in leveraging quantum technology for privacy protection. Similarly, Akrom's review [5] of quantum support-vector machines underscores the advantages of

quantum computing in machine learning. These findings indicate that quantum computing is developing at a rapid pace and has emerged as a powerful tool for the field of cryptography. Thus, employing quantum algorithms to analyze the security of symmetric encryption algorithms is not only a critical approach to evaluating their resistance to quantum attacks but also provides valuable insights into the potential threats quantum computing poses to modern cryptography.

Until 2010, quantum attacks on symmetric ciphers were not considered a significant threat. However, when Kuwakado and others [6] first introduced a polynomial distinguisher for a three-round Feistel cipher under a quantum chosen-plaintext attack (qCPA) setting, this perspective changed. Since then, various quantum attacks on symmetric ciphers have been developed.

Zhandry [7], Kaplan [8], and others have proposed two different models for the quantum cryptanalysis of symmetric ciphers:

Standard Security (Q1 Model): A block cipher is standard secure against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from pseudorandom permutation (PRP or a PRF) by making only classical queries.

Quantum Security (Q2 Model): A block cipher is deemed quantum secure against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from PRP (or a PRF) even by making quantum queries.

This paper assumes that the attackers belong to the Q2 model. Recent studies have analyzed the security of symmetric ciphers under this model. In 2012, Kuwakado et al. [9] studied the quantum security of the EM cipher under the Q2 model, utilizing Simon's algorithm to construct an efficient distinguisher under the qCPA setting, thereby proving that the quantum version of the EM cipher is not secure. Subsequently, in 2015, Dinur et al. [10] combined meet-in-the-middle and partitioning attacks to enhance attacks on Feistel structures of more than four rounds. In 2016, Kaplan [11] utilized Simon's algorithm to break CBC-MAC, PMAC, and other symmetric cipher systems in polynomial time, demonstrating that Simon's algorithm could be used for slide attacks, providing exponential acceleration. Leander et al. [12] in 2017 first combined Grover's algorithm with Simon's algorithm to construct a quantum attack framework, which was later applied to the analysis of FX structures. Since then, the Grover-meets-Simon algorithm has been extensively utilized by numerous scholars in the quantum analysis of symmetric ciphers. Following Kaplan et al.'s [11] development of quantum slide attacks, Hosoyamada and colleagues [13] in 2019 expanded upon these techniques, introducing a related-key attack on the EM cipher structure and presenting a two-round key-recovery attack on the EM cipher structure.

Dong et al. [14] combined Grover's algorithm and Simon's algorithm to introduce a new quantum key-recovery attack on Feistel structures with varying rounds. In 2019, Ito [15] and others proposed a novel distinguisher for Feistel structures under a quantum-chosen ciphertext attack (qCCA) setting. This distinguisher can differentiate, in polynomial time, between four-round Feistel-F and Feistel-KF constructions and six-round Feistel-FK structures from random permutations. Subsequently, quantum key-recovery attacks for r-round Feistel-KF and Feistel-FK structures were performed, achieving key-recovery in $O(2^{\frac{(r-4)n}{4}})$ and $O(2^{\frac{(r-6)n}{4}})$ time, respectively. In the same year, Dong et al. [16] conducted a study on quantum distinguisher and key recovery attacks for two generalized Feistel structure (GFS) algorithms. They constructed $2d-1$ rounds of quantum distinguisher against d-branch Type-1 GFS and $2d+1$ rounds of quantum distinguisher against 2d-branch Type-2 GFS. Using these quantum distinguishers, key-recovery attacks were conducted on the Type-1 and Type-2 GFS ciphers over d^2-d+2 and $4d$ rounds, respectively, with time complexities of $O(2^{(\frac{1}{2d^2}-\frac{3}{2d}+2)\cdot\frac{n}{2}})$ and $O(2^{\frac{d^2n}{2}})$.

Ito et al. [17] also designed a polynomial-level quantum distinguisher under the qCPA setting for $3d-3$ round configurations of Type-1 GFS, along with a d^2-d+1 round version under the qCCA setting. Based on these distinguishers, key-recovery attacks were performed on r-round Type-1 GFS ciphers, with complexities of $O(2^{\left(\frac{d^2}{2}-\frac{3d}{2}+2\right)\cdot\frac{k}{2}+\frac{\left(r-d^2\right)k}{2}})$ and $O(2^{\frac{\left(r-\left(d^2-d+1\right)\right)k}{2}})$.

Ni et al. [18] introduce a $3d-3$ rounds of quantum distinguisher on Type-1 GFS under the Q2 model and also investigate quantum attacks against the CAST-256 block cipher. In 2020, Cid et al. [19] demonstrated a qCPA on contracting Feistel structures and studied related-key attacks on balanced Feistel structures. That same year, Hodzic et al. [20], based on Simon's algorithm, developed a construction for seven-round and eight-round quantum distinguishers for generalized Feistel structures under the qCPA setting. In 2021, Li et al. [21] examined the round functions and linear transformation P of Camellia, presenting a five-round quantum distinguisher and, under the qCPA setting, proposed a seven-round Camellia algorithm key-recovery attack with a complexity of 2^{24}.

Cui et al. [22] initially defined weakly periodic functions, thereby extending the application scope of Simon's algorithm and further constructing several variant Feistel structure distinguishers. They proposed quantum key-recovery attacks for Feistel variants. In 2022, Canale et al. [23] provided an automated periodic function search algorithm under a quantum computing model, implementing key-recovery attacks for the five-round Feistel-FK structure. In 2023, Xu et al. [24], based on the divisibility of branch output functions, proposed quantum attacks on two types of GFS under the qCPA setting. They constructed quantum distinguishers for an 8-round 4F and a 5-round 2F under the Q2 model, conducting 12-round and 7-round quantum key-recovery attacks, respectively. Additionally, they constructed a six-round 2F quantum distinguisher on a weak divisibility basis, performing an eight-round quantum key-recovery attack.

In the same year, Sun et al. [25] studied the security of Type-1 generalized Feistel structures, constructing a quantum distinguisher for a d-branch d^2-1 round Type-1 GFS structure under the qCCA setting. Furthermore, under the qCPA setting for the Type-1 block cipher CAST-256, they introduced a 17-round quantum distinguisher and constructed a quantum key-recovery attack with a complexity of $O(2^{\frac{37(r-17)}{2}})$.

The rapid advancement of quantum computing has necessitated the development and evaluation of post-quantum cryptographic (PQC) schemes to ensure secure communication in the quantum era. Recent studies have highlighted both the practical applications and potential vulnerabilities of PQC. For instance, Aslam et al. [26] proposed a quantum-resilient blockchain-enabled secure communication framework for connected autonomous vehicles, demonstrating the applicability of PQC in real-world IoT scenarios. Similarly, Sim et al. [27] investigated the side-channel vulnerabilities of lattice-based cryptographic schemes, particularly CRYSTALS-KYBER, and introduced a chosen-ciphertext clustering attack by leveraging the side-channel leakage of Barrett reduction. Their attack achieved a 100% success rate in recovering secret keys on ARM Cortex-M4 microcontrollers. These findings underscore the dual necessity of ensuring mathematical robustness and addressing side-channel resistance in PQC implementations, especially in resource-constrained environments such as IoT devices.

Based on the findings of prior research, it is apparent that studies conducted under the qCCA model remain inadequate, and numerous aspects remain unexplored. Studies demonstrate that numerous schemes proven secure under qCPA can be compromised by quantum algorithms like Simon's algorithm in qCCA settings. More significantly, the NIST Post-Quantum Cryptography Standardization Project has adopted the Q2 model (i.e., qCCA security) as a core evaluation criterion. Consequently, analysis under qCCA

conditions constitutes an essential benchmark for assessing the quantum security of block cipher algorithms.

The paper presents a five-round distinguisher for Camellia algorithm [28] under the qCCA setting. This is achieved by studying the round function and the characteristics of the key scheduling algorithm. Additionally, a key recovery attack model is proposed, and the complexity of nine rounds of Camellia recovery key under the quantum computing model is analyzed using the distinguisher.

2. Preliminaries

2.1. Notation and Acronyms

The notations used in this paper and their explanations are presented in Table 1. List of the acronyms and their definitions used in this paper in Table 2.

Table 1. Notations and Their Definitions.

Symbol	Description
X_i	Output on the left side of the i-th round in the Feistel structure
X_{i-1}	Output on the right side of the i-th round in the Feistel structure
F_i	Round function of the i-th round in the Feistel structure
k_i	Round key for the i-th round
$<<<$	Circular left shift operation
$>>>$	Circular right shift operation
$k_{i,j}$	The j-th byte of the round key for the i-th round
S_i	S-box substitution in the round function of the i-th round
$\cap$	Logical AND operation
$\cup$	Logical OR operation

Table 2. Acronyms and their definitions.

Acronym	Definition
IoT	Internet of Things
qCPA	Quantum Chosen-Plaintext Attack
qCCA	Quantum Chosen-Ciphertext Attack
GFS	Generalized Feistel Structure
SP	Substitution–Permutation Network
PQC	Post-Quantum Cryptographic

2.2. Brief Description of Camellia Algorithm

The Camellia algorithm [28], jointly designed by NTT and Mitsubishi Electric in 2000, is known for its high security and efficient performance on both hardware and software platforms. It was selected as a winning algorithm in the European NESSIE project in 2003, recommended in Japan's CRYPTREC initiative the same year, became an IETF standard in 2004, and adopted as an ISO/IEC standard in 2005.

Camellia is based on a Feistel structure with a block length of 128 bits and supports key lengths of 128, 192, and 256 bits, corresponding to 18, 24, and 32 rounds, respectively.

2.2.1. Camellia Encryption Transformation

The encryption transformation of Camellia involves differing initial and final whitening keys, with FL/FL^{-1} functions inserted every six rounds. For the 128-bit key version, the process consists of three six-round Feistel structures and two rounds of FL/FL^{-1} functions. Below, Camellia with a 128-bit key is described in Figure 1.

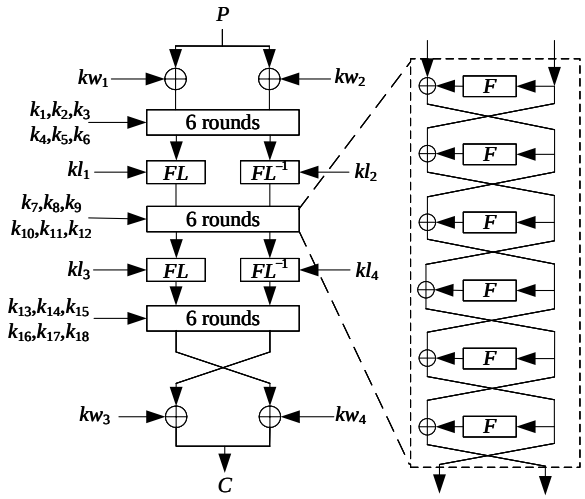


Figure 1. Encryption process of Camellia.

The plaintext M is 128 bits, the whitening key $kw_i (1 \leq i \leq 4)$ is 64 bits, and the round key $k_i (1 \leq i \leq 18)$ is 64 bits. The key $kl_i (1 \leq i \leq 4)$ utilized in each FL/FL^{-1} function is 32 bits, and the final output ciphertext C is 128 bits. The specific encryption process is as follows:

(1) Plaintext Whitening
A 128-bit plaintext M undergoes an XOR operation with the whitening key $kw_1 \| kw_2$, resulting in two parts: the left 64 bits X_{-1} and the right 64 bits X_0, such that $M \oplus (kw_1 \| kw_2) = X_{-1} \| X_0$.

(2) Round Iteration
For each round of the Feistel structure, let X_i denote the left output of the i-th round, and X_{i-1} denote the right output of the i-th round. For $i = 1, 2, \ldots, 18$, excluding $i = 6$ and $i = 12$, the i-th round transformation is performed as follows:

$$X_i = X_{i-2} \oplus F(X_{i-1}, k_i) \tag{1}$$

For $i = 6, 12$, the transformation is as follows:

$$\begin{aligned} X'_i &= X_{i-2} \oplus F(X_{i-1}, k_i) \quad X_i = FL\left(X'_i, kl_{i/3-1}\right) \\ X_{i-1} &= FL^{-1}(X_{i-1}, kl_{i/3}) \end{aligned} \tag{2}$$

(3) Pre-whitening of ciphertext output
The final round output $X_{18} \| X_{17}$ is XORed with the whitening keys $kw_3 \| kw_4$, producing the whitened ciphertext $C = (X_{18} \| X_{17}) \oplus (kw_3 \| kw_4)$.
The FL and FL^{-1} transformations are defined as follows:

$$\begin{aligned} &FL : F_2^{64} \to F_2^{64}, \\ &(X_L \| X_R, kl_L \| kl_R) \mapsto Y_L \| Y_R, \\ &Y_R = ((X_L \cap kl_L) \ll 1) \oplus X_R, \quad Y_L = (Y_R \cup kl_R) \oplus X_L \\ &FL^{-1} : \mathrm{F}_2^{64} \to \mathrm{F}_2^{64}, \\ &(Y_L \| Y_R, kl_R \| kl_L) \mapsto X_L \| X_R, \\ &X_L = (Y_R \cup kl_R) \oplus Y_L, \quad X_R = ((X_L \cap kl_L) \ll 1) \oplus Y_R. \end{aligned} \tag{3}$$

where $\cap$ represents bitwise logical "AND" operation; $\cup$ represents bitwise logical "OR" operation. In the Feistel structure, the function F during step (2) utilizes an SP structure design that incorporates round key XOR operations, S-box lookups, and the permutation **P**.

The final output of function F is formed by the outputs of eight S-boxes after undergoing the permutation **P**, as depicted in Figure 2. The specific steps involved are outlined below:

(1) Round Key XOR
A 64-bit input is divided into eight bytes. Each byte is then XORed with a corresponding round key byte before proceeding to the next step.

(2) S-Box Lookup
The XORed bytes sequentially query eight S-boxes in the order of $s1, s2, s3, s4, s2, s3, s4, s1$.

(3) Permutation **P**
The output from the S-boxes undergoes a linear transformation, described as follows.

$$\begin{aligned} y_1 &= x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7 \oplus x_8; y_5 = x_1 \oplus x_2 \oplus x_6 \oplus x_7 \oplus x_8 \\ y_2 &= x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7 \oplus x_8; y_6 = x_2 \oplus x_3 \oplus x_5 \oplus x_7 \oplus x_8 \\ y_3 &= x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_8; y_7 = x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_8 \\ y_4 &= x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7; y_8 = x_1 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \end{aligned} \tag{4}$$

The final output order is $(y_8, y_7, y_6, y_5, y_4, y_3, y_2, y_1)$. The diffusion layer **P** and its inverse $\mathbf{P}^{-1}$ have the following coefficient matrices:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}, \mathbf{P}^{-1} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{5}$$

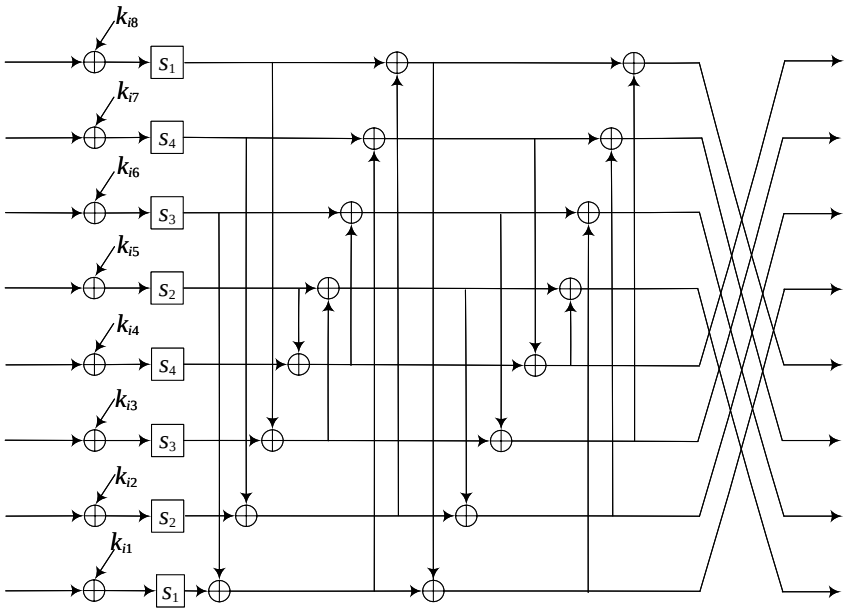


Figure 2. The round function of Camellia.

2.2.2. Key Expansion Algorithm

The round keys used in the encryption process are generated from a 256-bit initial key $k_{L(128)} \| k_{R(128)}$. Initially, $k_{L(128)} \| k_{R(128)}$ is input into the Feistel structure with round constants $\Sigma 1, \Sigma 2, \Sigma 3, \Sigma 4, \Sigma 5, \Sigma 6$. After four rounds, 128-bit $k_{A(128)}$ is generated, and after

six rounds, 128-bit $k_{B(128)}$ is produced. The structure of the algorithm is shown in Figure 3. The six round constants involved in generating $k_{A(128)}$ and $k_{B(128)}$ are:

$$\begin{aligned}
\Sigma_1 &= 0xA09E667F3BCC908B \\
\Sigma_2 &= 0xB67AE8584CAA73B2 \\
\Sigma_3 &= 0xC6EF372FE94F82BE \\
\Sigma_4 &= 0x54FF53A5F1D26F1C \\
\Sigma_5 &= 0x10E527FADE682D1D \\
\Sigma_6 &= 0xB05688C2B3E6C1FD
\end{aligned} \tag{6}$$

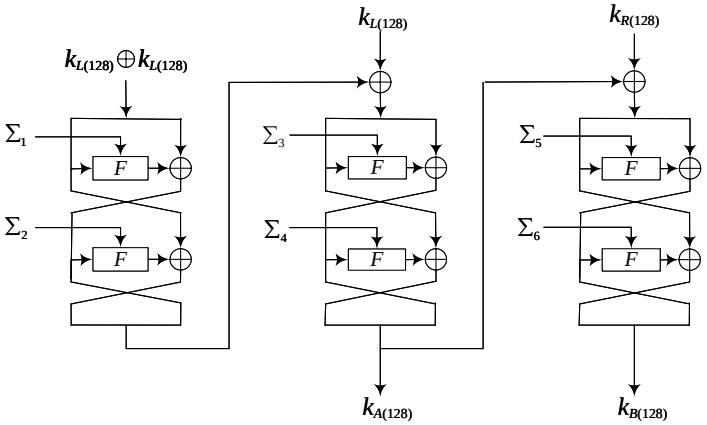


Figure 3. The key expansion algorithm of Camellia.

In the 128-bit version of the master key, the 256-bit initial key is defined as: $k_{L(128)} \| k_{R(128)} = k \| 0$. The round keys for each round are derived by shift transformations of the initial key K_L and K_A, as summarized in Table 3.

Table 3. Wheel keys for each wheel.

Round	Round Key	Round Key Value	Round	Round Key	Round Key Value
Pre-whitening	$kw_{1(64)}$	$(k_L <<< 0)_{L(64)}$	F (Round10)	$k_{10(64)}$	$(k_L <<< 60)_{R(64)}$
Pre-whitening	$kw_{2(64)}$	$(k_L <<< 0)_{R(64)}$	F (Round11)	$k_{11(64)}$	$(k_A <<< 60)_{L(64)}$
F (Round1)	$k_{1(64)}$	$(k_A <<< 0)_{L(64)}$	F (Round12)	$k_{12(64)}$	$(k_A <<< 60)_{R(64)}$
F (Round2)	$k_{2(64)}$	$(k_A <<< 0)_{R(64)}$	FL	$kl_{3(64)}$	$(k_L <<< 77)_{L(64)}$
F (Round3)	$k_{3(64)}$	$(k_L <<< 15)_{L(64)}$	FL^{-1}	$kl_{4(64)}$	$(k_L <<< 77)_{R(64)}$
F (Round4)	$k_{4(64)}$	$(k_L <<< 15)_{R(64)}$	F (Round13)	$k_{13(64)}$	$(k_L <<< 94)_{L(64)}$
F (Round5)	$k_{5(64)}$	$(k_A <<< 15)_{L(64)}$	F (Round14)	$k_{14(64)}$	$(k_L <<< 94)_{R(64)}$
F (Round6)	$k_{6(64)}$	$(k_A <<< 15)_{R(64)}$	F (Round15)	$k_{15(64)}$	$(k_A <<< 94)_{L(64)}$
FL	$kl_{1(64)}$	$(k_A <<< 30)_{L(64)}$	F (Round16)	$k_{16(64)}$	$(k_A <<< 94)_{R(64)}$
FL^{-1}	$kl_{2(64)}$	$(k_A <<< 30)_{R(64)}$	F (Round17)	$k_{17(64)}$	$(k_L <<< 111)_{L(64)}$
F (Round7)	$k_{7(64)}$	$(k_L <<< 45)_{L(64)}$	F (Round18)	$k_{18(64)}$	$(k_L <<< 111)_{R(64)}$
F (Round8)	$k_{8(64)}$	$(k_L <<< 45)_{R(64)}$	Post-whitening	$kw_{3(64)}$	$(k_A <<< 111)_{L(64)}$
F (Round9)	$k_{9(64)}$	$(k_A <<< 45)_{L(64)}$	Post-whitening	$kw_{4(64)}$	$(k_A <<< 111)_{R(64)}$

2.3. Related Algorithms

In this section, we offer a concise overview of classical quantum algorithms, specifically Simon's algorithm, Grover's algorithm, and the Grover-meets-Simon algorithm.

2.3.1. Simon's Algorithm

Given a Boolean function $f : \{0,1\}^n \to \{0,1\}^n$, which is guaranteed to satisfy $f(x) = f(y) \Leftrightarrow x \oplus y \in \{0, s\}$, it means the function has a period s and we need to find s. Classically, the optimal time to find period s is $O\left(2^{n/2}\right)$. Nevertheless,

Simon [3] introduced an algorithm that significantly expedites this process, requiring only $O(n)$ queries to determine s. This algorithm comprises the following five steps:

(1) Initialize two n-bit quantum registers in state $|0\rangle^{\otimes n}|0\rangle^{\otimes n}$ and apply the Hadamard transform to the first register to obtain the corresponding superposition state.

$$H^{\otimes n}|0\rangle|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|0\rangle \tag{7}$$

(2) Conduct a quantum query on function f and map it to the current state.

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle \tag{8}$$

(3) Measure the second register, reducing the first register to the following state:

$$\frac{1}{\sqrt{2}}(|z\rangle + |z \oplus s\rangle) \tag{9}$$

(4) Apply the Hadamard transform to the first register to obtain

$$\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{y\cdot z}(1 + (-1)^{y\cdot s})|y\rangle \tag{10}$$

(5) In this superposition state, the amplitudes corresponding to $y \cdot s = 1$ are zero. As a result, for any measurement of y, it is always true that $y \cdot s = 0$. By iterating this process $O(n)$ times, a set of linear equations can be constructed. Solving this system of equations results in determining the value of s.

At ISIT2010, Kuwawkado et al. [6] presented a quantum distinguishing attack on a three-round Feistel cipher constructed using Simon's algorithm. As illustrated in Figure 4, α_0 and α_1 are arbitrary constants.

$$\begin{aligned} &f : \{0,1\} \times \{0,1\}^n \to \{0,1\}^n \\ &b, x \to \alpha_b \oplus X_2, (X_3, X_2) = E(\alpha_b, x) \\ &f(b,x) = F_2(F_1(\alpha_b) \oplus x) \end{aligned} \tag{11}$$

Let f be a periodic function satisfying $f(b,x) = f(b \oplus 1, x \oplus F_1(\alpha_0) \oplus F_1(\alpha_1))$. Subsequently, the period $s = 1 \| F_1(\alpha_0) \oplus F_1(\alpha_1)$ can be obtained in polynomial time by employing Simon's algorithm.

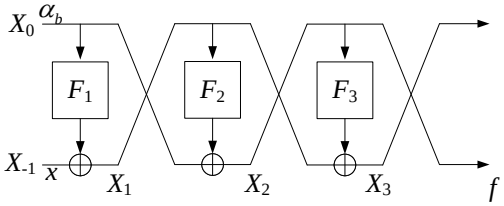


Figure 4. The periodic function of 3-round Feistel structure under qCPA setting.

2.3.2. Grover's Algorithm

When dealing with an unordered set of $N = 2^n$ elements, Grover's algorithm [2] is employed to pinpoint a unique element that fulfills certain criteria. Specifically, a quantum oracle O is used, which performs the operation $O|x\rangle = (-1)^{f(x)}|x\rangle$, where $f(x) = 0$ for all x except x_0 within the range $0 \leq x < 2^n$, and $f(x_0) = 1$. The goal is to determine x_0. The most efficient classical algorithm for searching this unordered data has a time complexity

of $O(N)$. However, Grover's algorithm, executed on a quantum computer, dramatically reduces this to merely $O(\sqrt{N})$ operations. The algorithm proceeds as follows:

(1) Initialize an n-bit register $|0\rangle^{\otimes n}$ and apply the Hadamard transform to the first register to achieve the corresponding superposition state, as shown in Equation (12):

$$H^{\otimes n}|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle = |\varphi\rangle \tag{12}$$

(2) Construct a quantum oracle $O : |x\rangle \rightarrow (-1)^{f(x)}|x\rangle$, if x is the correct state, then $f(x) = 1$; otherwise, $f(x) = 0$.

(3) Define the Grover iteration: $(2|\varphi\rangle\langle\varphi| - I)O$, and iterate this operation $R \approx \pi\sqrt{2^n}/4$ times:

$$[(2|\varphi\rangle\langle\varphi| - I)O]^R|\varphi\rangle \approx |x_0\rangle \tag{13}$$

(4) Return x_0.

2.3.3. Grover-Meets-Simon

During the 2017 ASIACRYPT conference, Leander et al. [12] presented a quantum key recovery attack approach that integrates Grover's algorithm with Simon's algorithm, specifically targeting the FX structure, depicted in Figure 5. The FX structure fulfills the given equation:

$$\mathrm{Enc}(x) = E_{k_0}(x + k_1) + k_2 \tag{14}$$

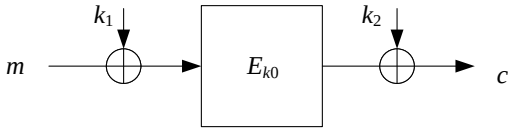


Figure 5. The structure of FX.

Reference [12] constructs the function $f(k, x) = \mathrm{Enc}(x) + E_k(x) = E_{k_0}(x + k_1) + k_2 + E_k(x)$. When the correct key guess $k = k_0$, it holds that $f(k, x) = f(k, x + k_1)$. However, for $k \neq k_0$, the function is not periodic. Under the qCPA setting, Reference [10] employs a combination of Simon's algorithm and Grover's algorithm to attack the FX structure.

Based on the work of Leander [12], Hosoyamada et al. [29] and Dong et al. [14] added a few rounds behind the three-round Feistel structured distinguisher shown in Figure 4 for recovering the keys of the Feistel encryption algorithm for the r rounds, with a time complexity of $O\left(2^{(r-3)n/2}\right)$.

3. Construction of Periodic Functions

This chapter presents a brief description of the periodic function of four-round Feistel structure proposed by ITO et al. [15]. Additionally, a periodic function for the Camellia algorithm is constructed and verified to be correct.

3.1. The Periodic Function of Four-Round Feistel Structure

At RSA 2019, Ito et al. introduced the design of periodic functions for a four-round Feistel cipher. They developed a quantum distinguisher and presented a key recovery attack against the Feistel structure. We will now describe the method for constructing a periodic function for a four-round Feistel structure, as proposed in Reference [15]. Figure 6 illustrates the structure of the periodic function.

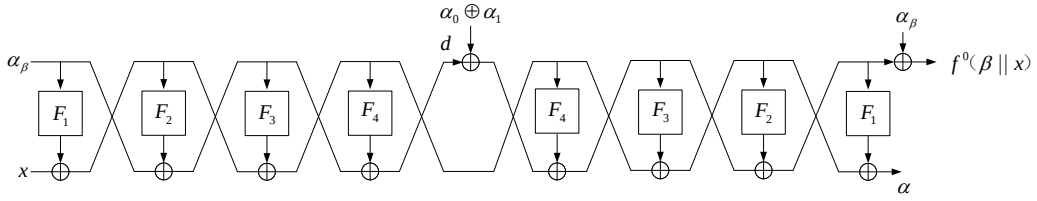


Figure 6. The periodic function of 4-round Feistel structure.

The four-round Feistel-F encryption structure is denoted as FF_4, and its corresponding decryption structure is represented by FF_4^{-1}. The round functions of Feistel-F are signified by $F_1, \ldots, F_4 \in \mathrm{Func}(n/2)$. The plaintext input $(a, b) \in \left(\{0,1\}^{n/2}\right)^2$ to FF_4, which outputs the ciphertext $(c, d) \in \left(\{0,1\}^{n/2}\right)^2$. The encryption structure $(a, b) \mapsto (c, d)$ is defined as follows:

$$\begin{aligned} c &= a \oplus F_1(b) \oplus F_3(b \oplus F_2(a \oplus F_1(b))) \\ d &= b \oplus F_2(a \oplus F_1(b)) \oplus F_4(a \oplus F_1(b) \oplus F_3(b \oplus F_2(a \oplus F_1(b)))) \end{aligned} \tag{15}$$

The decryption structure $(c, d) \mapsto (a, b)$ is defined as follow:

$$\begin{aligned} a &= c \oplus F_3(d \oplus F_4(c)) \oplus F_1(d \oplus F_4(c) \oplus F_2(c \oplus F_3(d \oplus F_4(c)))) \\ b &= d \oplus F_4(c) \oplus F_2(c \oplus F_3(d \oplus F_4(c))) \end{aligned} \tag{16}$$

For the input plaintext (α_β, x), the encryption and decryption structures can be further simplified.The simplified structure is depicted in Figure 7. The function $f^o(\beta \| x)$ is described as:

$$\begin{aligned} f^o(\beta \| x) = \alpha_0 \oplus \alpha_1 \oplus F_2(x \oplus F_1(\alpha_\beta)) \oplus F_2(x \oplus F_1(\alpha_\beta) \oplus F_3(\alpha_\beta \oplus F_2(x \oplus F_1(\alpha_\beta))) \\ \oplus F_3(\alpha_\beta \oplus \alpha_0 \oplus \alpha_1 \oplus F_2(x \oplus F_1(\alpha_\beta)))) \end{aligned} \tag{17}$$

Theorem 1. *Define* $Z_{(\beta \| x)} = F_1(\alpha_\beta) \oplus x$. *Then,* $Z_{(\beta \| x)}$ *is a periodic function with a period* $s = 1 \| (F_1(\alpha_0) \oplus F_1(\alpha_1))$.

Proof of Theorem 1. $Z_{((\beta \| x) \oplus s)} = x \oplus F_1(\alpha_0) \oplus F_1(\alpha_1) \oplus F_1\left(\alpha_{(\beta \oplus 1)}\right) = x \oplus F_1(\alpha_\beta) = Z_{(\beta \| x)}$. □

It is straightforward to demonstrate that $Z_{(\beta \| x)}$ is a periodic function. The output function $f^\circ(\beta \| x)$ can be described as follows:

$$f^o(\beta \| x) = \alpha_0 \oplus \alpha_1 \oplus F_2\left(Z_{\beta \| x}\right) \oplus F_2\left(Z_{\beta \| x} \oplus F_3\left(\alpha_0 \oplus F_2\left(Z_{\beta \| x}\right)\right) \oplus F_3\left(\alpha_1 \oplus F_2\left(Z_{\beta \| x}\right)\right)\right) \tag{18}$$

It can be deduced that $f^o(\beta \| x)$ is a periodic function with a period of s.

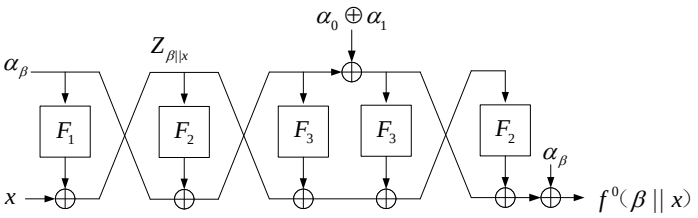


Figure 7. The equivalent structure of 4-round Feistel structure periodic function.

3.2. Construction of Periodic Functions for Camellia

In this section, we aim to construct a periodic function for Camellia, leveraging the periodic function construction methods outlined in the preceding section. To commence, we construct the five-round periodic function structure as illustrated in Figure 8.

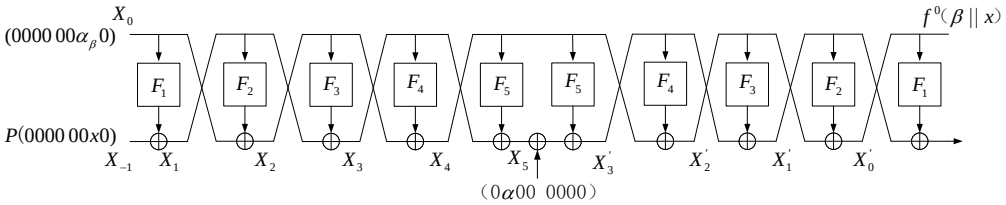


Figure 8. The construction of Camellia's periodic function.

In the illustrated structure, F_5 and F_1 do not participate in the construction of the periodic function $f^o(\beta \| x)$, thus the structure can be further simplified as shown in Figure 9.

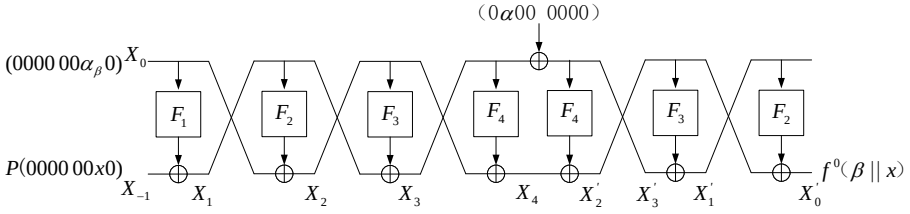


Figure 9. The quivalent structure of Camellia periodic function.

Let the inputs for the two branches be $(\mathbf{000000}\alpha_\beta\mathbf{0})$ and $\mathbf{P}(\mathbf{000000}x\mathbf{0})$, where $\mathbf{0}$ represents a sequence of eight zero bits. The variables α_β and x are both byte variables, both of which are located at the sixth byte. Byte indexing starts at 0 in this article. The input $\mathbf{P}(\mathbf{000000}x\mathbf{0})$ is obtained from $(\mathbf{000000}x\mathbf{0})$ through a permutation $\mathbf{P}$.

After the first round function transformation, we obtain the output X_1 as

$$X_1 = F_1(X_0) \oplus X_{-1} = F_1\left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \oplus \mathbf{P}(\mathbf{000000}x\mathbf{0}) = \mathbf{P}(\mathbf{000000} * \mathbf{0}) \tag{19}$$

where $* = s_1(\alpha_\beta) \oplus x, s_i$ is the s transformation of the i-th round function. Given the characteristics of Camellia algorithm's P permutation matrix, we obtain the output X_2 as:

$$X_2 = F_2(X_1) \oplus X_0 = \mathbf{P}(\mathbf{00}\Delta\Delta\Delta\Delta\Delta\mathbf{0}) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \tag{20}$$

Within this structure, each symbol Δ is distinct and consists solely of bytes related to $*$. The output obtained above continues to participate in the round function operation, and we can obtain the subsequent output as shown in the following equations:

$$\begin{aligned} X_3 &= X_1 \oplus F_3(X_2) = \mathbf{P}(\mathbf{000000} * \mathbf{0}) \oplus F_3\left(\mathbf{P}(\mathbf{00}\Delta\Delta\Delta\Delta\Delta\mathbf{0}) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right)\right) \\ &= \mathbf{P}(\mathbf{000000} * \mathbf{0}) \oplus F_3\left((\Delta\Delta\Delta\Delta\Delta\Delta\Delta\Delta) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right)\right) \\ &= \mathbf{P}(\mathbf{000000} * \mathbf{0}) \oplus F_3(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta) \\ &= \mathbf{P}(\mathbf{000000} * \mathbf{0}) \oplus \mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta) \\ &= \mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta) \end{aligned} \tag{21}$$

The symbol ? is used to indicate a byte where the periodicity of the function cannot be determined.

$$\begin{aligned} X_4 &= F_4(X_3) \oplus X_2 = F_4(\mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta)) \oplus \mathbf{P}(\mathbf{00}\Delta\Delta\Delta\Delta\Delta\mathbf{0}) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \\ &= F_4(\Delta\Delta?????\Delta) \oplus \mathbf{P}(\mathbf{00}\Delta\Delta\Delta\Delta\Delta\mathbf{0}) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \end{aligned} \tag{22}$$

$$\begin{aligned} X_2' &= F_4(X_3 \oplus (\mathbf{0}\alpha\mathbf{000000})) \oplus X_4 = F_4((\Delta\Delta?????\Delta) \oplus (\mathbf{0}\alpha\mathbf{000000})) \oplus X_4 \\ &= \mathbf{P}\left(\Delta\alpha'?????\Delta\right) \oplus \mathbf{P}(\Delta\Delta?????\Delta) \oplus \mathbf{P}(\mathbf{00}\Delta\Delta\Delta\Delta\Delta\mathbf{0}) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \\ &= \mathbf{P}\left(\Delta\alpha'\Delta\Delta\Delta\Delta\Delta\Delta\right) \oplus \left(\mathbf{000000}\alpha_\beta\mathbf{0}\right) \end{aligned} \tag{23}$$

$$X_1' = F_3(X_2') \oplus X_3' = \mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta) \oplus X_3 \oplus (\mathbf{0}\alpha\mathbf{000000}) = \mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta) \quad (24)$$

$$X_0' = X_2' \oplus F_2(X_1') = \mathbf{P}(\Delta\alpha'\Delta\Delta\Delta\Delta\Delta\Delta) \oplus (\mathbf{000000}\alpha_\beta\mathbf{0}) \oplus S_2\mathbf{P}(\mathbf{P}(\Delta\Delta\Delta\Delta\Delta\Delta?\Delta)) \quad (25)$$

Performing the $\mathbf{P}^{-1}$ operation on both sides of the above equation, we obtain:

$$\begin{aligned} \mathbf{P}^{-1}X_0' &= (\Delta\alpha'\Delta\Delta\Delta\Delta\Delta\Delta) \oplus \mathbf{P}^{-1}(\mathbf{000000}\alpha_\beta\mathbf{0}) \oplus S_2(\Delta\Delta?????\Delta) \\ &= (\Delta\alpha'\Delta\Delta\Delta\Delta\Delta\Delta) \oplus (\mathbf{00}\alpha_\beta\alpha_\beta\alpha_\beta\alpha_\beta\mathbf{0}\alpha_\beta) \oplus (\Delta\Delta?????\Delta) \\ &= (\Delta\Delta??????) \end{aligned} \quad (26)$$

As derived above, the 0 th and first bytes of $\mathbf{P}^{-1}X_0'$ relate only to the variable $*$; thus they can be used to construct a periodic function. Define the output byte $\mathbf{P}^{-1}(X_0')_1$ as the function $f^o(\beta, x)$, i.e., $f^o(\beta, x) = (\mathbf{P}^{-1}(X_0')_1)$. The theorem states the following:

Theorem 2. *The period of the function* $f^o(\beta, x)$ *is* $s = 1\|s_1(\alpha_0) \oplus s_1(\alpha_1)$.

$$f^o(\beta, x) = f^o(\beta \oplus 1, x \oplus s_1(\alpha_0) \oplus s_1(\alpha_1))$$

Proof of Theorem 2. When $\beta = 0$, for $f^o(\beta\|x), * = s(\alpha_0) \oplus x = s(\alpha_{0\oplus 1}) \oplus x \oplus s(\alpha_0) \oplus s(\alpha_1) = s(\alpha_0) \oplus x$. Since the value of $f^o(\beta\|x)$ relates only to $*$, it follows that $f^o(0\|x) = f^o(0 \oplus 1\|x \oplus s(\alpha_0) \oplus s(\alpha_1))$.

When $\beta = 1$, for $f^o(\beta\|x), * = s(\alpha_1) \oplus x = s(\alpha_{1\oplus 1}) \oplus x \oplus s(\alpha_0) \oplus s(\alpha_1) = s(\alpha_1) \oplus x$. Since the value of $f^o(\beta\|x)$ relates only to $*$, it follows that $f^o(1\|x) = f^o(1 \oplus 1\|x \oplus s(\alpha_0) \oplus s(\alpha_1))$. □

It can be demonstrated that the function $f^o(\beta\|x)$ exhibits periodicity. In accordance with Theorem 2 of the study [15], a distinguisher against the five-round Camellia can be constructed using the function $f^o(\beta\|x)$.

During the research process, we attempted to construct distinguishers with a higher number of rounds. However, distinguishers exceeding five rounds could not be theoretically verified for their correctness. Based on the method proposed in this paper, we consider the five-round distinguisher to be an appropriate choice. On the other hand, the five-round distinguisher offers a balance between complexity and theoretical verifiability, making it the most suitable choice based on the method proposed in this paper. This selection ensures that the distinguisher is both practical to implement and theoretically sound, which is critical for the reliability of our approach.

3.3. Experimental Validation

In this subsection, we conducted targeted experiments to validate the correctness of the periodic functions developed in the preceding section. In accordance with the Camellia algorithm criteria outlined in the literature [28], we have constructed the five-round periodic function structure presented in Section 3.2 on the Python 3.7 environment. Our primary aim was to verify the equation $f^\circ(\beta, x) = f^\circ(\beta \oplus 1, x \oplus s_1(\alpha_0) \oplus s_1(\alpha_1))$. We conducted correctness verification by setting different input parameters. Based on a local computer, we selected 2^{32} sets of distinct parameter data for periodicity testing, all of which confirmed the correctness of the periodic. As illustrated in Figure 10, the periodic function is demonstrated to be correct with a specific set of data.

The input plaintext group data are $(\mathbf{000000}\alpha_\beta\mathbf{0}), \mathbf{P}(\mathbf{000000}x\mathbf{0})$, where $x = (00000000)$, $\alpha_0 = (00000000)$, and $\alpha_1 = (00000001)$. When $\beta = 0$, for $f^\circ(\beta, x)$, the plaintext input comprises solely zeros. The output for the first byte, denoted as $(\mathbf{P}^{-1})X_0'$, is (01101100).

For $f^{\circ}(\beta \oplus 1, x \oplus s_1(\alpha_0) \oplus s_1(\alpha_1))$, the plaintext group data is updated to $(\mathbf{0000000}\alpha_1)$, $\mathbf{P}(\mathbf{000000}x \oplus s_1(\alpha_0) \oplus s_1(\alpha_1)0)$, and remarkably, the output for the first byte, again denoted as $(\mathbf{P}^{-1})X'_0$, remains unchanged at (01101100).

Given the experimental verification outlined above, Theorem 1 is confirmed to be accurate.

```
********************************************************************************
input: α_β = 0000 0000, x=0000 0000, α= 0000 0001
The first byte of the output from the periodic function (p^(-1) ) x_0^':0110 1100
********************************************************************************
input: α_β = 0000 0001, x = 1110 0101, α = 0000 0001
The first byte of the output from the periodic function (p^(-1) ) x_0^':0110 1100
```

Figure 10. Experimental result.

4. The Attack Model for Camellia

This section presents a key-recovery attack model for the Camellia algorithm under the qCCA setting, leveraging the five-round distinguisher proposed in Section 3.2. Here is an outline of our attack methodology:

1. Implement a nine-round encryption oracle, denoted as $\mathcal{E}$, and decrypt the input of the periodic function f_{in} over two rounds. The resulting intermediate value after the two rounds and the subkeys for the two rounds as the input to the circuit $\mathcal{E}$. Then, the output of $\mathcal{E}$ undergoes two rounds of decryption and is XORed with a constant.
2. Implement a quantum circuit $\mathcal{D}$ that computes the inverse of $\mathcal{E}$. Encrypt the results from the previous step over two rounds, inputting the intermediate value along with subkeys into the circuit $\mathcal{D}$. Subsequently, encrypt $\mathcal{D}$'s output over two rounds to acquire the periodic function output f_{out}.
3. Guess the keys for the two rounds preceding and succeeding the quantum circuits $\mathcal{E}$ and $\mathcal{D}$.
4. For each key guess, check its correctness with the following procedure.
   (1) Apply the five-round distinguisher to $\mathcal{E}$ and $\mathcal{D}$.
   (2) If the distinguisher returns "this is a random permutation", then judge that the guess is wrong. Otherwise, judge that the guess is correct.

Nine-Round Key-Recovery Attack on Camellia

A nine-round key-recovery attack is performed using the established attack model, as shown in Figure 11. The periodic function input f_{in} is $[\mathbf{P}(\mathbf{000000}x\mathbf{0}), (\mathbf{000000}\alpha_\beta\mathbf{0})]$, with the output f_{out} being $\mathbf{P}^{-1}((X_7)_1)$. To decrypt f_{in} through two rounds and retrieve the plaintext, the keys K_1 and $K_{2,\{2,3,4,5,7\}}$ need to be guessed. Subsequently, the ciphertext is obtained after nine rounds of encryption by the oracle. In addition, the keys to be guessed for the decryption of the ciphertext for two rounds are $K_{9,\{0,3,4,5,6\}}$ and $K_{8,7}$. After xOR the obtained intermediate state with the constant $[(\mathbf{00000000}), (\mathbf{0}\alpha\mathbf{000000})]$, it needs to guess the keys $K_{8,7}$ and $K_{9,\{0,3,5,6,7\}}$ to encrypt for two rounds to obtain the ciphertext. The plaintext is then obtained by nine rounds of decryption oracle. Finally, the keys K_1 and $K_{2,1}$ must be guessed in order to encrypt the plaintext for two rounds, thereby obtaining the output f_{out}.

Due to the characteristics of Camellia's key scheduling, where round keys for F (Round 1), F (Round 2), and F (Round 9) are derived from cyclic shifts of K_A, many key bits are repeated. We guess all key bits for Round 1 and bytes 1–5, 7 for Round 2. Round 9 requires guesses for 0 and 3–7 bytes of the key, with 45 bits being repetitions. As shown in Figure 12, the orange portion of the figure indicates the three-bit non-repeating key positions that must be guessed in Round 9. Thus, the actual key bits to guess are $K_{A[0\text{–}63]}, K_{A[72\text{–}111],[120\text{–}127]}, K_{L[37\text{–}44]}, K_{A[69,70,71]}$, totaling $64+48+8+3=123$ bits.

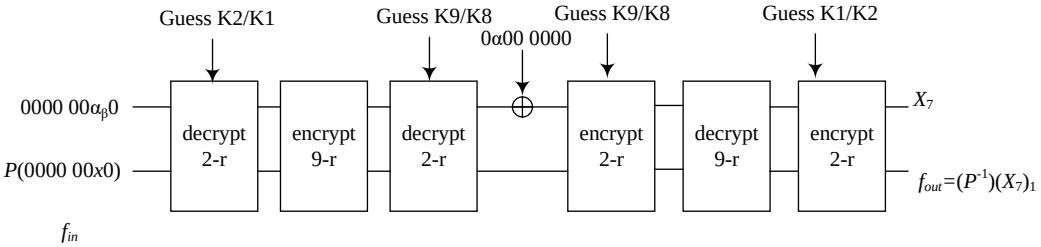


Figure 11. The key-recovery attack model of Camellia.

r	Key Bit Position																													
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
2	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
n	…																															
9	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76
	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108

Notes: Yellow parts represent duplicated key bits in the key schedule.
Orange parts indicate non-duplicated key bits in the round.

Figure 12. Camellia round key duplicate bits.

Define $g : F_2^{64} \times F_2^{48} \times F_2^8 \times F_2^{32} \times F_2^{(8+1)} \rightarrow F_2^8$ satisfying $(K_{A[0–63]}, K_{A[72–119]}, K_{L[37–44]}, K_{A[69,70,71]}, y) \rightarrow f(y)$, where $y = f^o(\beta, x)$. If the key guess is correct, the following holds true:

$$\begin{aligned} &g\left(K_{\mathrm{A}[0–63]}, K_{\mathrm{A}[72–119]}, K_{\mathrm{L}[37–44]}, K_{\mathrm{A}[69,70,71]}, y\right) = \\ &g\left(K_{\mathrm{A}[0–63]}, K_{\mathrm{A}[72–119]}, K_{\mathrm{L}[37–44]}, K_{\mathrm{A}[69,70,71]}, y \oplus s\right) \end{aligned} \tag{27}$$

If $f_{\mathrm{in}} \rightarrow f_{\mathrm{out}}$ is a periodic function, then the period of this function can be determined by inputting it into Simon's algorithm. The guess is correct; otherwise, the guess is incorrect.

As outlined in the literature [12], the formula for the number of quantum bits required for a key recovery attack is as follows:

$$\text{sum } = n_k + n_{\mathrm{in}} \times l + n_{\mathrm{out}} \times l, l = 2(\tilde{n} + \sqrt{n}) \tag{28}$$

where sum represents the total number of quantum bits required, n_k the length of the key, n_{in} the input length of the periodic function, n_{out} the output length of the periodic function, and $\tilde{n}$ the length of the period. For Camellia's guessed keys $K_A[0–63], K_A[72–119], K_L[37–44], K_A[69,70,71]$, we have that:

$$\begin{aligned} &n_k = 64 + 48 + 8 + 3 = 123, n_{\mathrm{in}} = 8 + 1 = 9, \tilde{n} = 8 + 1 = 9 \\ &n_{\mathrm{out}} = 8, l = 2(9 + \sqrt{9}) = 24, \text{ sum } = 123 + 9 \times 24 + 8 \times 24 = 531 \end{aligned} \tag{29}$$

The whole attack needs $123 + 9 \times 2(9 + \sqrt{9}) + 8 \times 2(9 + \sqrt{9}) = 531$ qubits. According to the methods described in References [14,16], the proof is provided as follows: Given an accurate guess of the key:

$$(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}K_{9,[69,70,71]}) \tag{30}$$

it holds that:

$$g(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}K_{9,[69,70,71]}, y) = g(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}K_{9,[69,70,71]}, y \oplus s). \tag{31}$$

Let the h be defined as:

$$\begin{aligned} h : F_2^{123} \times F_2^{(8+1)\times 24} &\to F_2^{8\times 24} \\ (K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_1, \ldots, y_{24}) &\\ \to g(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_1) \,||\, \ldots \,||& \\ g(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_{24}).& \end{aligned} \tag{32}$$

where $||$ denotes concatenation.

The constructed quantum gate U_h satisfies the following mapping:

$$\begin{aligned} &|K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_1, \ldots, y_{24}, 0, \ldots, 0\rangle \\ \to\ &|K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_1, \ldots, y_{24}, \\ &h(K_1, K_{2,\{1,2,3,4,5,6\}}, K_{8,7}, K_{9,[69,70,71]}, y_1, \ldots, y_{24}))\rangle. \end{aligned} \tag{33}$$

By constructing a quantum algorithm A, we achieve a key recovery attack on the Camellia algorithm. The algorithm begins by initializing 531 qubits, all set to the initial state $|0\rangle$. Among these qubits, the first $123 + 8 + 1 \times 24 = 339$ qubits undergo a Hadamard transform $H^{\otimes 339}$, resulting in the following uniform superposition:

$$\sum_{k_1 \in F_2^{64}, k_2 \in F_2^{48}, k_8 \in F_2^8, k_9 \in F_2^3, y_1, \ldots, y_{24} \in F_2^{8+1}} |k_1, k_2, k_{8,7}, k_9\rangle |y_1 \ldots y_{24}\rangle |0\rangle \tag{34}$$

In the proof, k_1 represents the full key for the first round, k_2 represents $K_{2,\{1,2,3,4,5,6\}}$, k_8 represents $K_{8,7}$, k_9 represents $K_{9,[69,70,71]}$, and $y_1, \ldots, y_{24}$ are auxiliary qubits. The amplitude of this quantum state, which can be ignored, is given by:

$$2^{-(123+(8+1)\times 24)/2} = 2^{-339/2}. \tag{35}$$

Next, the superposition state is processed through the U_h, resulting in the following state:

$$\sum_{k_1 \in F_2^{64}, k_2 \in F_2^{48}, k_8 \in F_2^8, k_9 \in F_2^3, y_1, \ldots, y_{24} \in F_2^{8+1}} |k_1, k_2, k_{8,7}, k_9\rangle |y_1 \ldots y_{24}\rangle |h(k_1, k_2, k_{8,7}, k_9, y_1, \ldots, y_{24})\rangle \tag{36}$$

To extract periodic information, a Hadamard transform is applied again to the $|y_1 \ldots y_{24}\rangle$ register, resulting in the following superposition:

$$\begin{aligned} |\phi\rangle = \sum_{k_1 \in F_2^{64}, k_2 \in F_2^{48}, k_8 \in F_2^8, k_9 \in F_2^3, y_1, \ldots, y_{24} \in F_2^{8+1}} & |k_1, k_2, k_{8,7}, k_9\rangle \quad (-1)^{\langle u_1, y_1\rangle} |u_1\rangle \ldots (-1)^{\langle u_{24}, y_{24}\rangle} |u_{24}\rangle \\ & |h(k_1, k_2, k_{8,7}, k_9, y_1, \ldots, y_{24})\rangle \end{aligned} \tag{37}$$

If the guessed key (k_1, k_2, k_8, k_9) is correct, the period s will be orthogonal to $u_1, \ldots, u_{24}$ upon measuring $|\phi\rangle$. According to Lemma 4 from Reference [12], choosing $l = 2 \times (8 + 1 + 8 + 1) = 24$ ensures that the period s can be computed.

The classification of the quantum state is performed using a classifier B, defined as $B : F_2^{123+(8+1)\times 24} \to \{0, 1\}$. The classifier operates as follows. First, it checks the dimension of

$$\bar{U} = \mathrm{Span}(|u_1, \ldots, u_{24}\rangle). \tag{38}$$

If $\dim(\bar{U}) \neq 24$, the classifier outputs 0. Otherwise, the unique period s is computed using Lemma 4 from Reference [12]. For any given y, the classifier verifies the equality:

$$g(k_1, k_2, k_8, k_9, y) = g(k_1, k_2, k_8, k_9, y \oplus s). \tag{39}$$

If this equality holds, the classifier outputs 1; otherwise, it outputs 0. The classifier B divides the quantum state $|\phi\rangle$ into the "good" subspace $|\phi_1\rangle$ and the "bad" subspace $|\phi_0\rangle$, such that:

$$|\phi\rangle = |\phi_1\rangle + |\phi_0\rangle. \tag{40}$$

Here, $|\phi_1\rangle$ represents the projection onto the "good" subspace, while $|\phi_0\rangle$ represents the projection onto the "bad" subspace. The "good" subspace satisfies $B = 1$. Furthermore, the classifier defines a unitary operator S_B, which acts as follows:

$$|k_1, k_2, k_8, k_9\rangle|y_1 \dots y_{24}\rangle \rightarrow \begin{cases} -|k_1, k_2, k_8, k_9\rangle|y_1 \dots y_{24}\rangle, & \text{if } B = 1, \\ |k_1, k_2, k_{8,7}, k_9\rangle|y_1 \dots y_{24}\rangle, & \text{if } B = 0. \end{cases} \tag{41}$$

Grover's algorithm is then applied to amplify the probability of the classifier outputting 1. For the initial state $|\phi\rangle = A|0\rangle$, the angle to the "bad" subspace is θ, where:

$$\sin^2(\theta) = p \approx 2^{-61.5}. \tag{42}$$

Using Grover's algorithm, the following iterative operator is applied:

$$Q = -AS_0A^{-1}S_B. \tag{43}$$

The number of iterations required is:

$$t = \lceil \pi/(4\theta) \rceil = \lceil \pi/(4 \times 2^{-61.5}) \rceil = 2^{61.5}. \tag{44}$$

After these iterations, the final state is almost orthogonal to the "bad" subspace, and the probability of measuring a "good" state approaches 1. The entire attack requires 531 qubits, including 123 qubits for storing the key, 9 qubits for the input to the periodic function, 8 qubits for the periodic function's output, and additional qubits for periodic computation. The time complexity of the attack is:

$$T = O(2^{61.5}). \tag{45}$$

This approach successfully achieves a key recovery attack on the Camellia algorithm.

The comparison between the work presented in this paper and other methods is shown in Table 4.

Table 4. Comparison with other attack methods.

Reference	Target Cipher	Attack Model	Attack Type	Rounds of Key Recovery	Time Complexity	Quantum Resources (Qubits)
[30]	Luby–Rackoff	qCPA	Distinguishing attack	4	$O(2^{n/12})$	N/A
[15]	Feistel-F	qCCA	Distinguishing attack	4	Polynomial $O(\mathrm{n})$	N/A
[21]	Camellia	qCPA	Key recovery attack	7	$O(2^{24})$	456 qubits
[31]	MARS-like (4 branches)	qCCA	Key recovery attack	9	$O(2^{2n})$	N/A
This work	Camellia	qCCA	Key recovery attack	9	$O(2^{61.5})$	531 qubits

5. Conslusions and Future Work

Reference [15]studied the quantum distinguisher for a four-round Feistel structure under the Q2 model but did not consider the internal structure of the algorithm. The contribution of this paper is the first construction of a quantum key-recovery attack against Camellia under the qCCA model. Specifically, based on the round function and key scheduling features of Camellia, we propose a five-round qCCA distinguisher. Subsequently, leveraging this distinguisher, we present a nine-round quantum key-recovery attack based on the Grover-meets-Simon algorithm, achieving a time complexity of $2^{61.5}$.

Compared to previous quantum analyses of Camellia, our attack demonstrates better performance and makes a significant contribution to the study of the quantum security of block cipher algorithms. The ability of symmetric cryptographic algorithms to resist quantum attacks is garnering increasing attention. In addition to conducting quantum security analyses of existing symmetric cryptographic algorithms, researchers are also considering quantum security as a critical criterion in proposing new symmetric cryptographic designs. Future research will focus on constructing quantum distinguishers with an increased number of rounds under quantum computing models. Furthermore, by exploring quantum algorithms, the time complexity of quantum key-recovery attacks may be further reduced. Additionally, the effectiveness of symmetric cryptographic structural schemes against quantum attacks will be further investigated.

Author Contributions: Conceptualization, Y.L.; methodology, Y.L., J.L. and Q.W.; software, D.H.; validation, Y.L. and Q.W.; formal analysis, H.X.; investigation, Y.L. and Q.W.; writing—original draft preparation, Y.L.; writing—review and editing, Y.L. and Q.W.; visualization, Q.W.; supervision, Y.L.; project administration, Y.L.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fund of Yunnan Key Laboratory of Blockchain Application Technology grant No. 202305AG340008, the Beijing Natural Science Foundation grant No. 4234084, and the Key Laboratory of Equipment Data Security and Guarantee Technology, Ministry of Education, under grant No. 2024010102.

Data Availability Statement: The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: Authors Yanjun Li and Jian Liu were employed by the company China Electronics Technology Group Corporation. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **1999**, *41*, 303–332. [CrossRef]
2. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. In Proceedings of the 28th 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**, *26*, 1474–1483. [CrossRef]
4. Zhou, N.R.; Chen, Z.-Y.; Liu, Y.-Y.; Gong, L.-H. Multi-party semi-quantum private comparison protocol of size relation with d-level GHZ states. *Adv. Quantum Technol.* **2024**, 2400530 . [CrossRef]
5. Akrom, M. Quantum support vector machine for classification task: A review. *J. Multiscale Mater. Inform.* **2024**, *1*, 1–8. [CrossRef]
6. Kuwakado, H.; Morii, M. Quantum Distinguisher Between the 3-Round Feistel Cipher and the Random Permutation. In Proceedings of the 2010 IEEE International Symposium on Information Theory, Austin, TX, USA, 12–18 June 2010; pp. 2682–2685. [CrossRef]
7. Zhandry, M. How to Construct Quantum Random Functions. In Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science, New Brunswick, NJ, USA, 20–23 October 2012; pp. 679–687. [CrossRef]
8. Kaplan, M.; Leurent, G.; Leverrier, A.; Naya-Plasencia, M. Quantum differential and linear cryptanalysis. *arXiv* **2015**, arXiv:1510.05836. [CrossRef]
9. Kuwakado, H.; Morii, M. Security on the Quantum-Type Even-Mansour Cipher. In Proceedings of the 2012 International Symposium on Information Theory and Its Applications, Cambridge, MA, USA, 1–6 July 2012; pp. 312–316.
10. Dinur, I.; Dunkelman, O.; Keller, N.; Shamir, A. New Attacks on Feistel Structures with Improved Memory Complexities. In Proceedings of the Annual Cryptology Conference, Santa Barbara, CA, USA, 16–20 August 2015; Springer: Berlin, Germany, 2015; pp. 433–454. [CrossRef]
11. Kaplan, M.; Leurent, G.; Leverrier, A.; Naya-Plasencia, M. Breaking symmetric cryptosystems using quantum period finding. In *Advances in Cryptology, Proceedings of the 36th Annual International Cryptology Conference (CRYPTO 2016), Santa Barbara, CA, USA, 14–18 August 2016*; Proceedings, Part II 36; Springer: Berlin/Heidelberg, Germany, 2016; pp. 207–237. [CrossRef]

12. Leander, G.; May, A. Grover meets Simon–quantumly attacking the FX-construction. In *Advances in Cryptology, Proceedings of the 2017 23rd International Conference on the Theory and Applications of Cryptology and Information Security (ASIACRYPT 2017), Hong Kong, China, 3–7 December 2017*; Proceedings, Part II 23; Springer: Berlin, Germany, 2017; pp. 161–178. [CrossRef]
13. 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]
14. Dong, X.; Wang, X. Quantum key-recovery attack on Feistel structures. *Sci. China Inf. Sci.* **2018**, *61*, 102501. [CrossRef]
15. Ito, G.; Hosoyamada, A.; Matsumoto, R.; Sasaki, Y.; Iwata, T. Quantum chosen-ciphertext attacks against Feistel ciphers. In *Topics in Cryptology, Proceedings of the 2019 Cryptographers' Track at the RSA Conference (CT-RSA 2019), San Francisco, CA, USA, 4–8 March 2019*; Springer: Cham, Switzerland, 2019; pp. 391–411. [CrossRef]
16. Dong, X.; Li, Z.; Wang, X. Quantum cryptanalysis on some generalized Feistel schemes. *Sci. China Inf. Sci.* **2019**, *62*, 22501. [CrossRef]
17. Ito, G.; Iwata, T. Quantum Distinguishing Attacks Against Type-1 Generalized Feistel Ciphers. Cryptology ePrint Archive, 2019. https://eprint.iacr.org/2019/327 (accessed on 15 April 2025).
18. Ni, B.; Dong, X. Improved quantum attack on type-1 generalized Feistel schemes and its application to CAST-256. *J. Electron. Inf. Technol.* **2020**, *42*, 295–306.
19. Cid, C.; Hosoyamada, A.; Liu, Y.; Sim, S.M. Quantum cryptanalysis on contracting Feistel structures and observation on related-key settings. In *Progress in Cryptology, Proceedings of the 21st International Conference on Cryptology in India (INDOCRYPT 2020), Bangalore, India, 13–16 December 2020*; Proceedings 21; Springer: Cham, Switzerland, 2020; pp. 373–394. [CrossRef]
20. Hodžić, S.; Knudsen, L.R. A quantum distinguisher for 7/8-round SMS4 block cipher. *Quantum Inf. Process.* **2020**, *19*, 411. [CrossRef]
21. Li, Y.; Lin, H.; Liang, M.; Sun, Y. A new quantum cryptanalysis method on block cipher Camellia. *IET Inf. Secur.* **2021**, *15*, 487–495. [CrossRef]
22. Cui, J.; Guo, J.; Ding, S. Applications of Simon's algorithm in quantum attacks on Feistel variants. *Quantum Inf. Process.* **2021**, *20*, 1–50. [CrossRef]
23. Canale, F.; Leander, G.; Stennes, L. Simon's Algorithm and Symmetric Crypto: Generalizations and Automatized Applications. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 15–18 August 2022; Springer: Cham, Switzerland, 2022; pp. 779–808.
24. Xu, Y.; Du, X.; Jia, M.; Wang, X.; Zou, J. Quantum attacks on generalized Feistel networks based on the strong–weak separability. *Quantum Inf. Process.* **2023**, *22*, 375. [CrossRef]
25. Sun, H.W.; Cai, B.B.; Qin, S.J.; Wen, Q.Y.; Gao, F. Quantum Attacks on Type-1 Generalized Feistel Schemes. *Adv. Quantum Technol.* **2023**, *6*, 2300155. [CrossRef]
26. Aslam, A.M.; Bhardwaj, A.; Chaudhary, R. Quantum-resilient blockchain-enabled secure communication framework for connected autonomous vehicles using post-quantum cryptography. *Veh. Commun.* **2025**, *52*, 100880. [CrossRef]
27. Sim, B.Y.; Park, A.; Han, D.G. Chosen-ciphertext clustering attack on CRYSTALS-KYBER using the side-channel leakage of Barrett reduction. *IEEE Internet Things J.* **2022**, *9*, 21382–21397. [CrossRef]
28. Aoki, K.; Ichikawa, T.; Kanda, M.; Matsui, M.; Moriai, S.; Nakajima, J.; Tokita, T. Camellia: A 128-bit block cipher suitable for multiple platforms—Design and analysis. In *Selected Areas in Cryptography, Proceedings of the 7th Annual International Workshop (SAC 2000), Waterloo, ON, Canada, 14–15 August 2000*; Proceedings 7; Springer: Berlin/Heidelberg, Germany, 2001; pp. 39–56. [CrossRef]
29. Hosoyamada, A.; Sasaki, Y. Quantum Demiric-Selçuk meet-in-the-middle attacks: Applications to 6-round generic Feistel constructions. In *Security and Cryptography for Networks, Proceedings of the 11th International Conference (SCN 2018), Amalfi, Italy, 5–7 September 2018*; Proceedings 11; Springer: Cham, Switzerland, 2018; pp. 386–403. [CrossRef]
30. Hosoyamada, A.; Iwata, T. 4-Round Luby-Rackoff Construction is a qPRP. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, 8–12 December 2019; Springer: Cham, Switzerland, 2019; pp. 145–174. [CrossRef]
31. Qian, X.; You, Q.D.; Zhou, X.; Zhang, Y.; Zhao, X.J. Quantum attack on MARS-like Feistel schemes. *J. Cryptologic Res.* **2021**, *8*, 417–431. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.