Improved quantum attack on Type-1 Generalized Feistel Schemes and Its application to CAST-256

Boyu Ni¹ and Xiaoyang Dong²

¹ Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, School of Mathematics, Shandong University, Jinan 250100, China.
² Institute for Advanced Study, Tsinghua University, Beijing 100084, China. xiaoyangdong@tsinghua.edu.cn

Abstract. Generalized Feistel Schemes (GFS) are important components of symmetric ciphers, which have been extensively researched in classical setting. However, the security evaluations of GFS in quantum setting are rather scanty. In this paper, we give more improved polynomial-time quantum distinguishers on Type-1 GFS in quantum chosen-plaintext attack (qCPA) setting and quantum chosen-ciphertext attack (qCCA) setting. In qCPA setting, we give new quantum polynomial-time distinguishers on (3d−3)-round Type-1 GFS with branches \(d \geq 3\), which gain \(d−2\) more rounds than the previous distinguishers. Hence, we could get better key-recovery attacks, whose time complexities gain a factor of \(2^{(d−2)n}\). In qCCA setting, we get (3d−3)-round quantum distinguishers on Type-1 GFS, which gain \(d−1\) more rounds than the previous distinguishers.

In addition, we give some quantum attacks on CAST-256 block cipher. We find 12-round and 13-round polynomial-time quantum distinguishers in qCPA and qCCA settings, respectively, while the best previous one is only 7 rounds. Hence, we could derive quantum key-recovery attack on 19-round CAST-256. While the best previous quantum key-recovery attack is on 16 rounds. When comparing our quantum attacks with classical attacks, our result also reaches 16 rounds on CAST-256 with 128-bit key under a competitive complexity.

Keywords: Generalized Feistel Scheme, Quantum attack, Simon’s algorithm, CAST-256

1 Introduction

Feistel block ciphers are featured by the efficient Feistel network, whose encryption and decryption process are based on similar operations. This design has been extensively researched [1–4] and adopted by many standard block ciphers, such as DES, Triple-DES, Camellia [5], GOST [6]. Feistel network was also generalized to form Generalized Feistel Networks (GFN), which adopts more branches and different operations between the branches. At CRYPTO 1989, Zheng et al. [7] summarised some Generalized Feistel Networks as 3 types GFN, namely Type-1, Type-2 and Type-3 GFN. In addition, some other Generalized Feistel Networks were invented by Anderson and Biham [8], Lucks [9] and Schneier and Kelsey [10]. Many important primitives employed GFN, such as block ciphers CAST-256 [11], CLEFIA [12], Simpira [13] as well as hash functions MD5 and SHA-1. The Generalized Feistel Network inherits the advantages of Feistel Network. Besides, it allows a small round function to construct a cipher with a larger block size, which is beneficial to lightweight implementations. In this paper, we focus on Type-1 and Type-2 GFN.

As the important block cipher CAST-256 and CLEFIA adopt Type-1 and Type-2 GFN, respectively, we also denote Type-1 GFN as CAST256-like GFN and denote Type-2 GFN as CLEFIA-like GFN.

Classically, Luby and Rackoff [14] proved that a three-round Feistel scheme is a secure pseudo-random permutation. At Asiacrypt 2000, Moriai and Vaudenay [15] studied some Generalized Feistel Schemes (GFS) and proved a 7-round 4-branch CAST256-like GFS and 5-round 4-branch CLEFIA-like GFS are secure pseudo-random permutations. Later, Hoang and Rogaway [16] improved and generalised the provable-security analysis of Type-1, Type-2 and Type-3 GFN. Generic attacks on those constructions are also widely studied, such as birthday attack [17], meet-in-the-middle attack [18], differential attacks [19,20], and Patarin et al.’s attacks [21–23].

Recently, the security evaluation of symmetric ciphers against quantum adversaries becomes a hot topic in communities. In the 2000s, it was a common belief that quantum attacks on symmetric primitives are of minor concern, as they mainly consist of employing Grover’s algorithm [24] to generically speed up search (sub-)problems. However, Kuwakado and Morii [25] found the first the polynomial-time quantum distinguisher on 3-round Feistel using quantum period finding algorithm, i.e., Simon’s algorithm [26]. Later, various quantum attacks against symmetric primitives were invented, such as key-recovery attacks
against Even-Mansour constructions [27], forgery or key-recovery attacks against block cipher based MACs [28,29], key-recovery attacks against FX constructions [30], and so on.

According to Zhandry’s work [31], there are two different models for quantum cryptanalysis against symmetric ciphers, i.e., standard security (also denoted as Q1) and quantum security (also denoted as Q2). In Q1 model, the adversaries could only collect data classically and processes them with local quantum computers. While in Q2 model, the adversaries could query the oracle with quantum superpositions of inputs, and obtain the corresponding superposition of outputs. The Q2 model is theoretically interesting. Moreover, as stated by Ito et al. [32], “the threat of this attack model becomes significant if an adversary has access to its white-box implementation. Because arbitrary classical circuit can be converted into quantum one, the adversary can construct a quantum circuit from the classical source code given by the white-box implementation”. In this paper, we assume that the adversaries come from Q2 model.

There are already some papers investigate Feistel schemes or GFS against Q2 adversaries. Besides Kuwakado and Morii [25]’s work, Ito et al. [32] extended the quantum distinguisher to 4-round Feistel construction under quantum chosen-ciphertext setting. Based on the Grover-meet-Simon algorithm by Leander and May [30], Hosoyamada et al. [33] and Dong et al. [34] introduced some quantum key-recovery attacks on Feistel schemes. Dong et al. [35] gave some quantum distinguishers and key-recovery attacks on some GFS. Dong et al. [36] and Bonnetain et al. [37] studied 2K-/4K-Feistel schemes against quantum slide attacks. Notably, Hosoyamada and Iwata [38] proved a tight quantum security bound of the 4-Round Luby-Rackoff construction recently.

Table 1. Rounds of quantum distinguishers on Type-1 GFS

<table>
<thead>
<tr>
<th>Source</th>
<th>Setting</th>
<th>$d$</th>
<th>$d$</th>
<th>$d$</th>
<th>$d$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[35]</td>
<td>qCPA</td>
<td>$2d-1$</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>Section 4.1</td>
<td>qCPA</td>
<td>$3d-3$</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>Section 4.2</td>
<td>qCPA</td>
<td>$3d-2$</td>
<td>7</td>
<td>10</td>
<td>13</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 2. Key-recovery attacks on Type-1 GFS ($d \geq 3$) in quantum settings

<table>
<thead>
<tr>
<th>Source</th>
<th>Distinguisher</th>
<th>Key-recovery rounds</th>
<th>Complexity ($\log$)</th>
<th>Trivial bound ($\log$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[35]</td>
<td>$2d-1$</td>
<td>$r \geq d^2$</td>
<td>$T + \frac{(r-d^2+2)}{2}$</td>
<td>$\frac{rn^2}{2}$</td>
</tr>
<tr>
<td>Section 4.1</td>
<td>$3d-3$</td>
<td>$r \geq d^2$</td>
<td>$T + \frac{(r-d^2)}{2}$</td>
<td>$\frac{rn^2}{2}$</td>
</tr>
<tr>
<td>$T$: ($\frac{1}{2}d^2 - \frac{3}{2}d + 2$) \cdot \frac{r}{2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Quantum attacks on CAST-256†

<table>
<thead>
<tr>
<th>Source</th>
<th>Setting</th>
<th>Distinguisher</th>
<th>Attacked rounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>[35]</td>
<td>qCPA</td>
<td>7</td>
<td>$2^{74}$</td>
</tr>
<tr>
<td>Section 5.1</td>
<td>qCPA</td>
<td>12</td>
<td>$2^{274}$</td>
</tr>
<tr>
<td>Section 5.2</td>
<td>qCCA</td>
<td>13</td>
<td>$2^{188}$</td>
</tr>
</tbody>
</table>

†: Note that for 256-bit key size version, the trivial bound is $2^{128}$ by Grover algorithm.

Our contribution

In this paper, we give some improve attacks on Type-1 Generalized Feistel Schemes (GFS) in Q2 model with quantum chosen-plaintext attack (qCPA) setting and quantum chosen-ciphertext attack (qCCA) setting, respectively.

First, in qCPA setting, we give new quantum polynomial-time distinguishers on ($3d-3$)-round Type-1 GFS with branches $d \geq 3$, which gain $d - 2$ more rounds than the previous distinguishers. Based on Leander and May’s algorithm [30], we could get better key-recovery attacks, whose time complexities gain a factor of $2^{\frac{d^2}{2}}$. Second, when considering qCCA setting, we get ($3d-3$)-round quantum distinguishers...
on Type-1 GFS, which gain $d - 1$ more rounds than the previous distinguishers. The distinguishers and the key-recovery attacks on Type-1 GFS are summarized in Table 1 and 2.

In addition, we also evaluate CAST-256 block cipher in qCPA and qCCA settings. We find 12-round and 13-round polynomial-time quantum distinguishers in qCPA and qCCA settings, respectively. Note that the best previous one is 7 rounds. Hence, we could derive quantum key-recovery attack on 19-round CAST-256. While the best previous quantum key-recovery attack is on 16 rounds. The results are summarized in Table 3. We also compare our quantum attacks with classical attacks in Table 4. When the key size of CAST-256 is 128, our result also reaches 16 rounds with a competitive complexity.

## 2 Notations

- $x^i_0$ the $j$th branch in the input;
- $x^i_j$ the $j$th branch in the output of $i$th round, $i \geq 1$, $j \geq 1$;
- $d$ the branch number of CAST256-like GFS;
- $2d$ the branch number of RC6/CLEFIA-like GFS;
- $R^i$ the $i$th ($i \geq 1$) round function of Type-1 (CAST256-like) GFS, the input and output are $n$-bit string, $n$-bit key is absorbed by $R^i$;
- $R^i_j$ the $j$th ($1 \leq j \leq d$) round function in the $i$th ($i \geq 1$) round function of Type-2 (RC6/CLEFIA-like) GFS, the input and output are $n$-bit string, $n$-bit key is absorbed by $R^i_j$;

## 3 Related works

### 3.1 Simon’s algorithm

Given a function $f \{0, 1\}^n \rightarrow \{0, 1\}^n$, that is known to be invariant under some $n$-bit XOR period $a$, find $a$. In other words, find $a$ by given: $f(x) = f(y) \iff x \oplus y \in \{0^n, a\}$.

Classically, the optimal time to solve the problem is $O(2^{n/2})$. However, Simon [26] gives a quantum algorithm that provides exponential speedup and only requires $O(n)$ quantum queries to find $a$. The algorithm includes five quantum steps:

I. Initializing two $n$-bit quantum registers to state $|0\rangle^\otimes n|0\rangle^\otimes n$, one applies Hadamard transform to the first register to attain an equal superposition:

$$H^\otimes n|0\rangle|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0, 1\}^n} |x\rangle|0\rangle. \quad (1)$$

II. A quantum query to the function $f$ maps this to the state

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

III. Measuring the second register, the first register collapses to the state:

$$\frac{1}{\sqrt{2}} (|z\rangle + |z \oplus a\rangle).$$

IV. Applying Hadamard transform to the first register, we get:

$$\frac{1}{\sqrt{2}} \sum_{y \in \{0, 1\}^n} (-1)^{y \cdot z}(1 + (-1)^{y \cdot a})|y\rangle.$$
The vectors $y$ such that $y \cdot a = 1$ have amplitude 0. Hence, measuring the state yields a value $y$ that $y \cdot a = 0$.

Repeat $O(n)$ times, one obtains $a$ by solving a system of linear equations. However, Kaplan et al. [28] and Santoli [42] showed that Simon’s promise may be weakened at the cost of computing many vectors $y$ that $y \cdot a = 0$. At Asiacrypt 2017, Leander and May [30] assume that $f(x)$ behaves as a random periodic function with period $a$, and show that any function value $f(x)$ has only two preimages with probability at least $\frac{1}{2}$. Moreover, they show that $l = 2(n + \sqrt{n})$ repetitions of the Simon’s algorithm are sufficient to compute $a$. The probability is greater than $\frac{4}{3}$ that it contains at least $n - 1$ linearly independent vectors $y$ that are orthogonal to $a$ (Lemma 4, [30]).

At ISIT 2010, Kuwakado and Morii [25] introduced a quantum distinguish attack on 3-round Feistel scheme using Simon’s algorithm. As shown in Figure 1, the promise may be weakened at the cost of computing many vectors $y$.

3.2 Hosoyamada and Sasaki’s Method to Truncate Outputs of Quantum Oracles

As shown in Figure 1, in Kuwakado and Morii’s quantum distinguisher, one has to truncate the output 2n bits of $E$ to obtain the right half n bits, namely $x_2^n$. However, Kaplan et al. [28] and Hosoyamada et al. [33] pointed out that in quantum setting it is not trivial to truncate the entangled 2n qubits to n qubits, since the usual truncation destroys entanglements.

At SCN 2018, Hosoyamada and Sasaki [33] introduced a method to simulate truncation of outputs of quantum oracles without destroying quantum entanglements. Let $O : |x⟩|y⟩|z⟩|w⟩ → |x⟩|y⟩|z ⊕ O_L(x, y)|w ⊕ O_R(x, y))$ be the encryption oracle $E$, where $O_L$, $O_R$ denote the left n bits and right n bits of the complete encryption, respectively. The goal is to simulate oracle $O_R : |x⟩|y⟩|w⟩ → |x⟩|y⟩|w ⊕ O_R(x, y))$. Hosoyamada and Sasaki first simulate a tweaked $O_R$, i.e., $O'_R : |x⟩|y⟩|w⟩|0^n⟩ → |x⟩|y⟩|w ⊕ O_R(x, y)|0^n⟩$. Let $|+⟩ := H^n|0^n⟩$, where $H^n$ is an n-bit Hadamard gate. Thus, $O|x⟩|y⟩|+⟩|w⟩ → |x⟩|y⟩|+⟩|w ⊕ O_R(x, y))$. Then, they define $O'_R := (I ⊗ H^n) ◦ \text{Swap} ◦ O ◦ \text{Swap} ◦ (I ⊗ H^n)$, where $\text{Swap}$ is an operator that swaps last 2n-qubits: $|x⟩|y⟩|z⟩|w⟩ → |x⟩|y⟩|w⟩|z⟩$. $O'_R$ can get the period $s$ times, one obtains $\text{Swap} ◦ O ◦ \text{Swap} ◦ (I ⊗ H^n)$.

3.3 Grover’s algorithm

Given an unordered set of $N = 2^n$ items, Grover’s algorithm is to find the unique element that satisfies some condition. In other words, given a quantum oracle $O$ which performs the operation $O|x⟩ = (-1)^{f(x)}|x⟩$, where $f(x) = 0$ for all $0 ≤ x < 2^n$ except $x_0$, for which $f(x_0) = 1$, find $x_0$. While the best
classical algorithm for a search over unordered data requires $O(N)$ time, Grover’s algorithm performs the search on a quantum computer in only $O(\sqrt{N})$ operations, a quadratic speedup. The steps of the algorithm are as follows:

1. Initialization of a $n$-bit register $|0\rangle^{\otimes n}$. Apply the Hadamard transform to the first register to attain an equal superposition that can be given as follows:
   \[ H^{\otimes n}|0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle = |\varphi\rangle. \tag{2} \]

2. Construct an oracle $O: |x\rangle \rightarrow (-1)^{f(x)}|x\rangle$, where $f(x) = 1$ if $x$ is the correct state; otherwise, $f(x) = 0$.
3. Define the Grover iteration as $(2|\varphi\rangle\langle \varphi| - I)O$, and apply it $R \approx \frac{\pi}{4} \sqrt{2^n}$ times:
   \[ [(2|\varphi\rangle\langle \varphi| - I)O]^R|\varphi\rangle \approx |x_0\rangle. \]
4. return $x_0$.

### 3.4 Combining Grover and Simon’s algorithms

At Asiacrypt 2017, Leander and May [30] gave a quantum key-recovery attack on FX-construction shown in Figure 2: $Enc(x) = E_{k_0}(x + k_1) + k_2$. They introduce the function $f(k, x) = Enc(x) + E_k(x) = E_{k_0}(x + k_1) + k_2 + E_k(x)$. For the correct key guess $k = k_0$, we have $f(k, x) = f(k, x + k_1)$ for all $x$. However, for $k \neq k_0$, $f(k, \cdot)$ is not periodic. They combine Simon and Grover algorithm to attack FX ciphers in the quantum-CPA model with complexity roughly $2^{32}$.

Based on Leander and May’s work, Hosoyamada and Sasaki [33], and Dong and Wang [34] appended several rounds to the 3-round Feistel distinguisher in Figure 1 to recover the keys of an $r$-round Feistel cipher in time $O(2^{(r-3)n/2})$. See the previous papers [33,34] for details.

#### 3.5 Ito et al.’s attack on Feistel cipher

At RSA 2019, Ito et al.’s [32] gives new quantum distinguisher on 4-round Feistel cipher in quantum chosen-ciphertext attack (qCCA) setting.

As shown in Figure 3, plaintext $(\alpha_0, x)$ is first encrypted by 4-round Feistel to get the ciphertext $(d, c)$, then a new tweaked ciphertext $(d \oplus \alpha_0 \oplus \alpha_1, c)$ is decrypted by the inverse 4-round Feistel to get new plaintext. Then, Ito et al. defined function:
\[ f^O = \alpha_0 \oplus \alpha_1 \oplus F_2(x \oplus F_1(\alpha_2)) \]
\[ \oplus F_2(x \oplus F_1(\alpha_3) \oplus F_3(\alpha_3 \oplus F_2(x \oplus F_1(\alpha_2)))) \]
\[ \oplus F_3(\alpha_3 \oplus \alpha_0 \oplus \alpha_1 \oplus F_2(x \oplus F_1(\alpha_2))), \]

where \( \beta \) is 0 or 1, and \( \alpha_3 \) is constant. Hence, \( f^O \) has period \( s = 1 || F_1(\alpha_0) \oplus F_1(\alpha_1) \).

Combining with Leander and May’s algorithm [30], Ito et al. gave key-recovery attacks on Feistel using the new distinguisher. In addition, in the key-recovery attacks, they did not actually compute the period of \( f^O \), instead, they distinguish \( f \) by checking the dimension of the space spanned by the vectors given by the Simon’s algorithm. Thus, there will not be a problem if there are several partial periods or periods other than \( s \) because it distinguishes \( f \) without computing \( s \). In this paper, we also use this method to launch our key-recovery attacks on Type-1 GFS and CAST-256. For more details, we refer the readers to Ito et al.’s paper [32].

4 Quantum attack on Type-1 GFS

As shown in Figure 4, the input of the Type-1 cipher is divided into \( d \) branches, i.e. \( x_j^0 \) for \( 1 \leq j \leq d \), each of which has \( n \)-bit, so the blocksize is \( d \times n \). \( R^i \) is the round function that absorbs \( n \)-bit secret key \( k_i \) and \( n \)-bit input.

Dong et al. [35] first considered some quantum distinguishers and key-recovery attacks on Type-1 GFS. In this section, we give some improved polynomial-time quantum distinguishers on Type-1 GFS in quantum chosen-plaintext attack (qCPA) setting and quantum chosen-ciphertext attack (qCCA) setting, respectively. Based on those distinguishers, some improved key-recovery attacks are achieved.

4.1 Quantum distinguishers on Type-1 GFS in qCPA setting

For the sake of clarity, we first give an example attack on Type-1 with \( d = 4 \), then extend to any case with \( d \geq 3 \). After that, the key-recovery attacks are given.

Example case of Type-1 with \( d = 4 \) in qCPA setting When \( d = 4 \), we have a 9-round quantum distinguisher as shown in Figure 5.

The encryption process of 9-round Type-1 GFS is \( E(x_1^0, x_2^0, \alpha_0, x) = (x_1^0, x_2^0, x_3^0, x_4^0) \), where \( x_1^0 \) and \( x_2^0 \) are constants and \( b = 0, 1 \), \( \alpha_0, \alpha_1 \) are also constants, \( \alpha_0 \neq \alpha_1 \). Suppose \( h(\alpha_0) = R^3(R^2(R^1(x_1^0) \oplus x_2^0) \oplus \alpha_0) \). Define \( f(b, x) = x_2^0 \oplus \alpha_b = R^6(R^5(R^4(h(\alpha_0) \oplus x) \oplus x_1^0) \oplus R^1(x_1^0) \oplus x_2^0) \oplus R^2(R^1(x_1^0) \oplus x_2^0) \).

Hence, we deduce
\[
\begin{align*}
f(0, x) &= R^6(R^5(R^4(h(\alpha_0) \oplus x) \oplus x_1^0) \oplus R^1(x_1^0) \oplus x_2^0) \oplus R^2(R^1(x_1^0) \oplus x_2^0) \\
&= f(1, x \oplus h(\alpha_0) \oplus h(\alpha_1)), \\
f(1, x) &= R^6(R^5(R^4(h(\alpha_1) \oplus x) \oplus x_1^0) \oplus R^1(x_1^0) \oplus x_2^0) \oplus R^2(R^1(x_1^0) \oplus x_2^0) \\
&= f(0, x \oplus h(\alpha_0) \oplus h(\alpha_1)).
\end{align*}
\]

Thus, \( f(b, x) = f(b \oplus 1, x \oplus h(\alpha_0) \oplus h(\alpha_1)) \), \( f(b, x) \) is a function with period \( s = 1 || h(\alpha_0) \oplus h(\alpha_1) \).
Fig. 5. 9-round distinguisher on Type-1 GFS with $d = 4$
Extending to any case of Type-1 with $d \geq 3$ in qCPA setting. We construct the quantum distinguisher on the $(3d − 3)$-round cipher. The intermediate state after the $i$th round is $x_i^j$ for $1 \leq j \leq d$, especially the output of the $(3d − 3)$th round is denoted as $x_1^{3d−3}||x_2^{3d−3}||...||x_d^{3d−3}$.

The encryption process of $(3d − 3)$-round Type-1 GFS is $E(x_1^0, ..., x_{3d−2}^0, a_0, x)$, where $b = 0, 1$, and $\alpha_0, \alpha_1$ are arbitrary constants, $\alpha_0 \neq \alpha_1$, and $x_0^0 = x$. All remaining branches $x_1^0, x_2^0, ..., x_{3d−2}^0$ are constants.

Similar to the attack on case $d = 4$, we also define $h(\alpha_b) = R^d−1(R^d−2(...R^3(2(R^2(2(x_1^0) \oplus x_2^0) \oplus x_3^0)...) \oplus x_{3d−2}^0))$.

Then, we define

$$f(b, x) = x_2^{3d−3} \oplus \alpha_b = R^{2d−2}(R^{2d−3}(...(h(\alpha_b) \oplus x)... \oplus R^{d−3}(... (R^2(2(x_1^0) \oplus x_2^0) \oplus x_3^0)...) \oplus x_{3d−2}^0))$$

(5)

With simple computation, we deduce $f(b, x) = f(b \oplus 1, x \oplus g(\alpha_0) \oplus g(\alpha_1))$. Therefore, function $f$ satisfies Simon’s promise with $s = 1||g(\alpha_0) \oplus g(\alpha_1)$. Thanks to Hosoyamada and Sasaki’s work [33] shown in Sect. 3.2, we could truncate outputs of quantum oracles with ease, and hence $f$ could be implemented as quantum oracle. The period $s$ of $f$ could be found using Simon’s algorithm.

---

**Fig. 6.** 16-round quantum key-recovery attack on Type-1 GFS with $d = 4$

Quantum key-recovery attacks on Type-1 GFS in qCPA setting. We give an example key-recovery attack on Type-1 GFS with $d = 4$ branches. Following the similar idea that combines Simon’s and Grover’s algorithms to attack Feistel structure [33, 34], we append 7 rounds under the 9-round distinguisher to launch the attack. As shown in Figure 6, there are $4n$-bit key needed to be guessed by Grover’s algorithm.
to compute \( x_2^3 \), which are highlighted in the red boxes of round functions. Note that, here we do not need Hosoyamada and Sasaki’s method to truncate the output of encryption oracle. We just need to implement a function \( h_D(k_{10}, k_{13}, k_{14}, k_{16}, x_{16}^3, x_{16}^2, x_{16}^1) = x_2^9 \), where \( k_{10}, k_{13}, k_{14}, k_{16} \) are round keys in \( R^{10}, R^{13}, R^{14}, R^{16} \), respectively. \( h_D \) decrypts \( (x_{16}^1, x_{16}^2, x_{16}^3, x_{16}^4) \) to get \( x_2^9 \) by guessing \( k_{10}, k_{13}, k_{14}, k_{16} \).

Hence, the 16-round quantum-key-recovery attack needs about \( 2^{9n} \) queries and \( O(n^3) \) qubits. If we attack \( r \) > 16 rounds, we need guess \((r - 12)n\) key bits by Grover’s algorithm. Thus, the time complexity is \( 2((r - 2)n)^{3/2} \).

Generally, for \( d \geq 3 \), we could get \((3d - 3)\)-round quantum distinguisher. We append \( d^2 - 3d + 3 \) rounds under the quantum distinguisher to attack \( r_0 = d^2 \) rounds Type-1 GFS. Similarly, we need to guess \((\frac{1}{2}d^2 - \frac{3}{2}d + 2)n\)-bit key by Grover’s algorithm. Thus, for \( r_0 \) rounds, the time complexity is \( (\frac{1}{2}d^2 - \frac{3}{2}d + 2) \cdot \frac{n}{2} \) queries, and \( O(n^2) \) qubits are needed. If we attack \( r > r_0 \) rounds, we need guess \((\frac{1}{2}d^2 - \frac{3}{2}d + 2)n + (r - r_0)n\) key bits by Grover’s algorithm. Thus, the time complexity is \( 2((\frac{1}{2}d^2 - \frac{3}{2}d + 2) \cdot \frac{n}{2} + (r - r_0)n)^{3/2} \).

### 4.2 Quantum distinguishers on Type-1 GFS in qCCA setting

**Example case of Type-1 with \( d = 4 \) in qCCA setting** When \( d = 4 \), we get 10-round quantum distinguisher as shown in Figure 7. Note that, for simplicity, we omit the last swap function.

The encryption process is \( E(x_4^1, x_4^0, x) = (x_{10}^1, x_{10}^2, x_3^1, x_4^1) \), where \( b = 0, 1 \), and \( a_0, a_1 \) are arbitrary constants, \( a_0 \neq a_1 \), and \( x_0^1 = x \). The branches \( x_4^0, x_4^2 \) are constants. The decryption is \( D(x_{10}^1, x_{10}^2, a_0 \oplus a_1, x_3^1, x_4^1) = (y_i^1, y_i^2, y_i^3, y_i^4) \), note that as shown in Figure 7, many internal states in the decryption phase (right side of Figure 7) are the same to that in the encryption phase (left side of Figure 7).

In the right side of Figure 7, following the red lines and blue lines, we define \( f(b, x) = y_1^0 = R^3(R^2(g(b, x) \oplus a_0 \oplus a_1) \oplus R^2(g(b, x)) \oplus x_4^2) \oplus x_4^2 \), where \( g(b, x) = x_4^2 \).

Then we try to compute the ANF of \( g(b, x) \). We first denote \( h(a_0) = R^8(R^8(h(a_0) \oplus x_4^2) \oplus a_0) \). Then, \( g(b, x) = x_4^2 = R^{12}(R^8(h(a_0) \oplus x_4^2) \oplus R^8(h(a_0) \oplus x_4^2) \oplus R^8(h(a_0) \oplus x_4^2) \oplus R^8(h(a_0) \oplus x_4^2) \oplus a_1) \). Through simple computation, we could deduce \( g(0, x) \oplus a_0 \oplus a_1 = g(1, x \oplus h(a_0) \oplus h(a_1)) \) and \( g(1, x) \oplus a_0 \oplus a_1 = g(0, x \oplus h(a_0) \oplus h(a_1)) \).

Hence, \( g(x) = R^3(h(a_0) \oplus x) \) is periodic, and the period is \( 1 || h(a_0) \oplus h(a_1) \). Then, we rewrite \( f(b, x) = R^8(g(b, x) \oplus x_4^2) \oplus x_4^2 \). Since \( x_4^2 = h(a_0) \oplus x_4^2 = h(a_0) \oplus x \) holds, \( x_4^2 \) is a function on \( b, x \) with the same period to \( g(b, x) \), i.e. the period is also \( 1 || h(a_0) \oplus h(a_1) \). Meanwhile, \( x_4^2 = R^3(h(a_0) \oplus x) \oplus x_4^0 \), thus it also has the same period \( 1 || h(a_0) \oplus h(a_1) \).

So \( f(b, x) \) is a function with period \( 1 || h(a_0) \oplus h(a_1) \).

**Extending to any case of Type-1 with \( d \geq 3 \) in qCCA setting** We construct the new quantum distinguisher on the \((3d - 2)\)-round cipher. The intermediate state after the \((3d - 2)\)th round is denoted as \( x_{3d-2} \). The encryption process is \( E(x_0^1, x_0^0, ..., x_{3d-2}^0, a_o, x) = (x_{3d-2}^1, x_{3d-2}^2, ..., x_{3d-2}^d) \), where \( b = 0, 1 \), and \( a_0, a_1 \) are arbitrary constants, \( a_0 \neq a_1 \), and \( x_0^1 = x \). The branches \( x_0^1, x_0^2, ..., x_{3d-2}^d \) are constants. So we also denote encryption process as \( E(a_0, x) \) for simplicity. The decryption is \( D(x_{3d-2}^1, x_{3d-2}^2 \oplus a_0 \oplus a_1, ..., x_{3d-2}^d) = (y_1^1, y_2^1, ..., y_d^1) \). Define \( g(b, x) \):

\[
\begin{align*}
g(b, x) = x_{3d-2}^1 &= R^{3d-2}((R^{3d-1}(R^{d-1}(R^{d-1}(R^d(x_0^1) \oplus x_0^2) \oplus x_0^3) ... \oplus a_0) \oplus x) \oplus x_0^1) \oplus R^1(x_0^1) \oplus x_0^2) \oplus R^{d-3}(R^{d-3}(R^d(x_0^1) \oplus x_0^2) \oplus x \ominus a_0) \oplus x) \oplus x_0^3 \oplus R^d(x_0^1) \oplus x_0^2) \oplus x_0^3 \oplus R^d(x_0^1) \oplus x_0^2) \oplus x_0^3 \oplus R^d(x_0^1) \oplus x_0^2) \oplus x_0^3 \oplus R^d(x_0^1) \oplus x_0^2) \oplus a_0)
\end{align*}
\]

We also take the case \( d = 4 \) as an example shown in Figure 7, and we follow the red lines in the encryption process, and get

\[
\begin{align*}
x_{3d-2}^1 &= x_{3d-3}^1 = R^{2d-4}((x_{3d-3}^1) \oplus x_{3d-5}^1) \oplus x_{3d-5}^1), 3 \leq i \leq d, \\
x_{3d-2}^1 &= x_{3d-3}^1 = R^{2d-3}((x_{3d-3}^1) \oplus x_{3d-4}^1), \\
x_{3d-2}^1 &= R^{2d-2}(x_{3d-2}^1) \oplus x_{3d-3}^1
\end{align*}
\]

Following the red lines in the decryption process of Figure 7, and get

\[
\begin{align*}
y_0^1 &= R^d(R^{d-1}(R^{3d-3}(x_{3d-3}^1) \oplus a_0) \oplus a_1) \oplus x_0^3) \oplus R^{2d}(x_{3d-2}^1) \oplus x_{3d-2}^2)
\end{align*}
\]

We construct function \( f \) as following:
Fig. 7. 10-round distinguisher on CAST256-like GFS with $d = 4$
Improved quantum attack on Type-1 Generalized Feistel Schemes and Its application to CAST-256

$f : \{0, 1\} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$
b, x → y
where \((x_{3d-2}, x_{3d-2}, ..., x_{3d-2}) = E(α, x), \)
\((y_0, y_0, ..., y_0) = D(x_{3d-2}, x_{3d-2} \oplus α_0 \oplus α_1, ..., x_{3d-2}), \)
\(f(b, x) = R^1(R^{3d-2}(x_{3d-3} \oplus α_0 \oplus α_1) \oplus R^{3d-2}(x_{3d-2}) \oplus x_{3d-2}^2). \)

Quantum key recovery attacks on Type-1 GFS in qCCA setting

We add \(r - 3d + 2\) rounds to the \((3d - 2)\)-round distinguisher to launch the key-recovery attack by Leander and May's algorithm [30].

The attack processes are as follows:

1. Run the quantum circuit that takes the intermediate state value \((x_{3d-2}, x_{3d-2}, ..., x_{3d-2}) \oplus \alpha_0, x)\) after the first \((r - 3d + 2)\) rounds and the subkeys for the first \((r - 3d + 2)\) rounds as inputs, and decrypt the first \((r - 3d + 2)\) rounds get the plaintext. Then use the encryption oracle \(E\) encrypt the plaintext \((x_{3d-2}, x_{3d-2}, ..., x_{3d-2})\) to get the ciphertext \((x_{3d-2}, x_{3d-2}, ..., x_{3d-2})\).

2. Run the quantum circuit which takes the ciphertext \((x_{3d-2}, x_{3d-2}, ..., x_{3d-2})\) and the \((r - 3d + 2)\) rounds subkeys as input. Make quantum decryption query \(D\) of the ciphertext to get the plaintext, and use the the plaintext and subkeys to decrypt the last \((r - 3d + 2)\) rounds to get the intermediate state \((y_{3d-2}, y_{3d-2}, ..., y_{3d-2})\).

3. Guess the subkeys of the first \((r - 3d + 2)\) rounds. For each guessed subkey, use the the \((3d - 2)\) rounds distinguisher in \(E\) and \(D\) to check its correctness. If the distinguisher is a periodic permutation, then judge that the guess is correct. Otherwise judge that the guess is wrong.

For the \(r > 3d - 2\) round, there are \((r - 3d + 2)n\)-bit key needed to be guessed by Grover's algorithm. So the \(r\)-round quantum key-recovery attack needs about \(2^{\frac{r-3d+2}{2}n}\) time and \(O(n^2)\) qubits.

5 Quantum attacks on CAST-256 block cipher

CAST-256 block cipher is a first-round AES candidate. It is composed of 48 rounds, including 24 rounds Type-1 GFM and 24 rounds inverse Type-1 GFM, as shown in Figure 8, which is composed of 9-round Type-1 GFM and 3-round inverse Type-1 GFM. The block size is 128 bits, which are divided into four 32-bit branches and the key size can be 128, 192 or 256 bits. Each round function absorbs 37-bit subkey. Our attack is quit general and does not need any other details of the cipher.

In this section, we give two quantum attacks on CAST-256 block cipher in qCPA setting and qCCA setting, respectively.

5.1 Quantum attack on CAST-256 in qCPA setting

As shown in Figure 8, we construct 12-round quantum distinguisher on CAST-256, which includes 9-round Type-1 GFM and 3-round inverse Type-1 GFM. The distinguisher is very similar to the 9-round distinguisher in Section 4.1.

Suppose \(h(α) = R^1(R^1(x_1^0) \oplus x_0^0)\), Define \(f(b, x) = x_2^0 \oplus α_0 = R^b(R^f(R^4(h(α) \oplus x) \oplus x_0^0) \oplus R^1(x_1^0) \oplus x_2^0) \oplus R^2(R^4(x_1^0) \oplus x_2^0)\). As shown in Section 4.1, \(f(b, x) = f(b \oplus 1, x \oplus h(α) \oplus h(α_1))\), \(f(b, x)\) is a function with period \(s = 1\) when attacking CAST-256 when \(r > 12\) with \(2^{r-12} = 2^{18.5r-222}\) Grover iterations are needed.
Fig. 8. 12-round distinguisher on CAST-256

Fig. 9. 14-round attack on CAST-256
Improved quantum attack on Type-1 Generalized Feistel Schemes and Its application to CAST-256

Fig. 10. 13-round distinguisher on CAST-256
5.2 Quantum attack on CAST-256 in qCCA setting

We construct a 13-round quantum distinguisher in qCCA setting as shown in Figure 10. The distinguisher is very similar to the 10-round distinguisher of Type-1 GFS in Section 4.2.

The encryption process is \( E(x_1^0, x_2^0, x_0, x) = (x_1^{13}, x_2^{13}, x_3^{13}, x_1^{13}) \), where \( b = 0, 1 \) and \( \alpha_0, \alpha_1 \) are arbitrary constants, \( \alpha_0 \neq \alpha_1 \), and \( x_1^0 = x \). The branches \( x_1^0, x_2^0 \) are constants. The decryption is \( D(x_1^{13}, x_2^{13}, x_3^{13}, x_4^{13}) = (y_1^0, y_2^0, y_3^0, y_4^0) \). Note that as shown in Figure 10, many internal states in the decryption phase (right side of Figure 10) are the same to that in the encryption phase (left side of Figure 10).

In the right side of Figure 10, following the red lines and blue lines, we define \( f(b, x) = y_1^0 = R^2(g(b, x) \oplus \alpha_0 \oplus \alpha_1) \oplus x_1^0 \). The encryption process is \( R(x_1^0) = (x_1^0) \). Then we try to compute the ANF of \( g(b, x) \). We first denote \( h(\alpha_0) = R^3(R^2(R^1(x_0^0) \oplus x_2^0) \oplus \alpha_0 \oplus \alpha_1) \).

\[ h(\alpha_0) = R^3(R^2(R^1(x_0^0) \oplus x_2^0) \oplus \alpha_0 \oplus \alpha_1) \]

Thus, we have to guess all the subkeys in the first 13-round distinguisher. We thus also have the same period 1 for all the subkeys in the first 13 rounds, i.e. \( (r - 13) \times 37 \)-bit key. Thus, about \( 2^{(r-13) \times 37} = 2^{18.5r - 240.5} \) Grover iterations are needed.

6 Conclusion

In this paper, we give more improved polynomial-time quantum distinguishers on Type-1 GFS in quantum chosen-plaintext attack (qCPA) setting and quantum chosen-plaintext attack (qCCA) setting. First, we give new qCPA quantum distinguishers on \((3d - 3)\)-round Type-1 GFS with branches \( d \geq 3 \), which gain \( d - 2 \) more rounds than the previous distinguishers. Hence, we could get better key-recovery attacks, whose time complexities gain a factor of \( 2^{3d-3m} \). We also get \((3d - 3)\)-round qCCA quantum distinguishers on Type-1 GFS, which gain \( d - 1 \) more rounds than the previous distinguishers. In addition, we also discuss some quantum attacks on CAST-256 block cipher.

References


