Quantum meet-in-the-middle attack on Feistel construction

Yinsong Xu¹,²,³ · Zheng Yuan¹,²,³

Received: 13 July 2022 / Accepted: 19 October 2022 / Published online: 22 March 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
Inspired by Hosoyamada and Sasaki (in: International conference on security and cryptography for networks, pp 386–403. Springer, 2018), we propose a new quantum meet-in-the-middle (QMITM) attack on \( r \)-round (\( r \geq 7 \)) Feistel construction to reduce the time complexity, which is based on Guo et al. (Des Codes Cryptogr 80(3):587–618, 2016) classical meet-in-the-middle (MITM) attack. In our attack, we adjust the size of truncated differentials to balance the complexities between constructing the tables and querying firstly and introduce a quantum claw finding algorithm to solve the collision search problem in classical MITM attack. The total time complexities of our attack are only \( O\left(2^{2n/3} \cdot n\right) \), \( O\left(2^{19n/24} \cdot n\right) \) and \( O\left(2^{(r-5)n/4} \cdot n\right) \), when \( r = 7, r = 8 \) and \( r > 8 \), lower than classical and quantum attacks. Moreover, our attack belongs to Q1 model and is more practical than other quantum attacks.

Keywords Quantum meet-in-the-middle attack · Feistel construction · Quantum claw finding algorithm · Q1 model.

1 Introduction

A Feistel network is a scheme that builds \( n \)-bit permutations from smaller, usually \( n/2 \)-bit permutations or functions [11]. The Feistel-based design approach is widely used in block ciphers. In particular, a number of current and former international or national block cipher standards such as DES [6], Triple-DES [18] and Camellia [2] are Feistel ciphers.
Feistel ciphers have many constructions, and the analyzed target construction in this paper is displayed in Fig. 1. An $n$-bit state is divided into $n/2$-bit halves denoted by $a_i$ and $b_i$, and the state is updated by iteratively applying the following two operations:

$$a_{i+1} \leftarrow b_i \oplus F(k_i \oplus a_i), \quad b_{i+1} \leftarrow a_i,$$

where $F$ is a public function and $k_i$ is a subkey with $n/2$ bits. Note that the target Feistel construction is also called Feistel-2 in Ref. [14], or called Feistel-KF in Ref. [19]. For brevity, we simply call it the Feistel construction.

For the target Feistel construction, Guo et al. [14] absorbed the idea of meet-in-the-middle attack proposed by Demirci and Selçuk [7] and applied it on the attack of the Feistel construction. They firstly computed all the possible sequences constructed from a $\delta$-set such that a pair of messages satisfy the proposed 5-round distinguisher. Then, they collected enough pairs of plaintexts and corresponding ciphertexts and guessed the subkeys out of the distinguisher to compute the corresponding sequence for each pair. Finally, if the computed sequence from each pair of collected plaintexts belongs to the precomputed sequences from the distinguisher, the guessed subkeys are correct; otherwise, they are wrong. In their attack on 6-round Feistel construction, its time complexity is $O(2^{3n/4})$. Later, Zhao et al. combined Guo et al.’s work and pairs sieve procedure to attack on 7-round Feistel construction with $O(2^n)$ time. Besides, some other attacks penetrate up to 6 rounds, such as impossible differentials [25], all-subkey recovery [16, 17] and integral-like attacks [29].

In the quantum setting, cryptanalysts hope to take advantage of quantum computing to further reduce attack complexity. For symmetric cryptosystems, Grover’s algorithm [13] can provide a quadratic speedup over exhaustive search on keys. In addition, some other quantum algorithms have been applied to the analysis of block ciphers and achieved good results, such as Simon’s algorithm [28] and Bernstein–Vazirani algorithm [3]. Many block ciphers have been evaluated for the security in the quantum setting, for example, against Even–Mansour cipher [23], MACs [20, 26], AEZ [27], AES-COPA [31], FX construction [24] and so on. To evaluate the adversary’s ability, Kaplan divides these quantum attacks into two attack models: Q1 model and Q2 model [21].

1. **Q1 model.** The adversary is allowed only to make classical online queries and performs quantum offline computation.
2. **Q2 model.** The adversary can make quantum superposition online queries for cryptographic oracle and performs quantum offline computation.
Obviously, the adversary is more practical in Q1 model and more powerful in Q2 model.

Many researches have analyzed the security of the Feistel construction in Q1 or Q2 model. In 2010, Kuwakado et al. [22] proposed a quantum 3-round Feistel distinguisher and used Simon’s algorithm to recover subkeys in Q2 model. Its time complexity only needs $O(n)$ because of Simon’s algorithm. Based on Kuwakado et al.’s work, Dong et al. [9, 10] used Grover’s algorithm to search the last $r - 3$ round subkeys of $r$-round Feistel construction, inspired by Leander and May’s work [24]. Its time complexity is $O(2^{0.25nr-0.75n})$.

Different from Kuwakado et al.’s quantum 3-round Feistel distinguisher, Xie et al. [30] and Ito et al. [19] proposed new quantum Feistel distinguishers, respectively. Xie et al. [30] used Bernstein–Vazirani algorithm instead of Simon’s algorithm to recover subkeys. However, this modification causes a slight increase in complexity with $O(n^2)$ time. Ito et al. [19] proposed a new 4-round Feistel distinguisher and used Grover’s algorithm to search the last $r - 4$ rounds subkeys similarly to Dong et al.’s attack. Besides, Dong et al. [8] and Bonnetain et al. [4] proposed quantum slide attacks on Feistel construction by Simon’s algorithm, respectively.

The quantum attacks listed above are all Q2 models; Hosoyamada et al. [15] proposed a new quantum attack on 6-round Feistel construction that belongs to Q1 model. They gave a quantum claw finding algorithm to find a match between two tables, where two tables are constructed by two phases of Guo et al.’s attack: one table from precomputation and the other one from collected pairs. If there exists one match, the guessed subkeys are correct.

**Contribution.** Inspired by Hosoyamada et al.’s work [15], we propose a new quantum meet-in-the-middle attack (QMITM) on $r$-round Feistel construction to reduce time complexity in Q1 model. Guo et al.’s attack is to calculate the same state value in the cryptographic algorithm according to the distinguisher and the queried data, and store them in two tables, respectively. Then, they need to find the same element from the two tables and recover the key according to the found element. The main complexity of their attack consists of constructing the tables and querying the tables. Therefore, we try to reduce these complexities with the help of quantum algorithms in our attack. Firstly, we use Grover’s algorithm to speed up searching in the phase of constructing the tables. And we adjust the truncated differentials to balance the complexities between the distinguisher and the queried data. Finally, we use quantum claw finding algorithm [5] to find the same element, and recover the keys. In summary, the total time complexities of our attack are only $O(2^{2n/3} \cdot n)$, $O(2^{19n/24} \cdot n)$ and $O(2^{(r-5)n/4} \cdot n)$, when $r = 7$, $r = 8$ and $r > 8$, lower than classical and quantum attacks. The detailed comparison with other attacks is shown in Table 1.

This paper is organized as follows: Sect. 2 provides a brief description of related quantum algorithms. And an overview of Guo et al.’s work is presented in Sect. 3. We propose a new quantum meet-in-the-middle attack on 7-round Feistel construction in Sect. 4. Then, the attack is furthered on $r$-round ($r > 7$) as shown in Sect. 5, followed by a conclusion in Sect. 6.
Table 1 Comparison with classical and quantum attacks on Feistel construction

| Ref  | Setting  | Round | Time                | Data       | Classical memory | Qubits |
|------|----------|-------|---------------------|------------|------------------|--------|
| [14] | Classical | 6     | $O(2^{3n/4})$       | $O(2^{3n/4})$ | $O(2^{3n/4})$    | -      |
| [32] | Classical | 7     | $O(2^n)$            | $O(2^n)$   | $O(2^{3n/4})$    | -      |
| [8]  | Q2       | 4     | $O(n)$              | $O(2^{n/2})$| $O(2^{n/2})$     | $O(n)$ |
| [4]  | Q2       | 4     | $O(n^2)$            | $O(2^{n/2})$| $O(2^{n/2})$     | $O(n)$ |
| [10] | Q2       | $r \geq 3$ | $O(n^{2.25n r - 0.75n})$ | $O(2^{n/2})$| $O(2^{n/2})$     | $O(n^2)$ |
| [19] | Q2       | $r \geq 4$ | $O(n^{2.25n r - n})$ | $O(2^{n/2})$| $O(2^{n/2})$     | $O(n^2)$ |
| [30] | Q2       | 3     | $O(n^2)$            | $O(2^{n/2})$| $O(2^{n/2})$     | $O(n)$ |
| Sect. 4 Q1 | 7     | $O(2^{2n/3} \cdot n)$ | $O(2^{2n/3})$ | $O(2^{n})$ | $O(2^{5n/6} \cdot n)$ |
| Sect. 5 Q1 | 8     | $O(2^{5n/6})$       | $O(2^{2n/3})$ | $O(2^{5n/6})$ | $O(2^{3n/4} \cdot n)$ |
| Sect. 5 Q1 | 8     | $O(21n/24 \cdot n)$ | $O(2^{2n/3})$ | $O(2^{5n/6})$ | $O(2^{5n/6} \cdot n)$ |
| Sect. 5 Q1 | $r > 8$ | $O(2^{(r-5)n/4} \cdot n)$ | $O(2^{2n/3})$ | $O(2^{5n/6})$ | $O(2^{5n/6} \cdot n)$ |

2 Related quantum algorithms

Grover’s algorithm. Grover’s algorithm, or the Grover search, is one of the most famous quantum algorithms, with which we can obtain quadratic speed up on database searching problems compared to the classical algorithms. It was originally developed by Grover [13].

Problem 1 Suppose a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is given as a black box, with a promise that there is $x$ such that $f(x) = 1$. Then, find $x$ such that $f(x) = 1$.

The process of solving the above problem is presented simply as below by Grover’s algorithm.

1. Start with a uniform superposition $|\varphi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0, 1\}^n} |x\rangle$.
2. The unitary transformation $(2 |\varphi\rangle \langle \varphi| - 1) \mathcal{O}_f$ applied on $|\varphi\rangle$ is iterated $\sqrt{2^n}$ times, where the quantum oracle $\mathcal{O}_f$ defined by the function $f$ is a quantum amplitude flip operation followed as below:

$$\mathcal{O}_f |x\rangle \rightarrow \begin{cases} - |x\rangle, & \text{if } f(x) = 1, \\ |x\rangle, & \text{otherwise}. \end{cases} \quad (1)$$

3. The final measurement gives the $x$ such that $f(x) = 1$ with an overwhelming probability.

In summary, Grover’s algorithm can solve Problem 1 with $O(2^{n/2})$ evaluations of $f$ using $O(n)$ qubits. If we have $2^p$ independent small quantum processors with $O(n)$ qubits, by parallel running $O(\sqrt{2^n \cdot 2^p})$ iterations on each small quantum processor, we
Algorithm 1 Quantum Claw Finding between Two Arbitrary Functions $f$ and $g$ [5]

**Input:** Sets $X$ and $Y$ of size $N$ and $M$, respectively.

**Output:** A pair of $(x, y) \in X \times Y$ such that $f(x) = g(y)$ if they exist.

1. Select a random subset $A \subseteq [N]$ of size $l$ ($l \leq \min(N, \sqrt{M})$).
2. Select a random subset $B \subseteq [M]$ of size $l^2$.
3. Sort the elements in $A$ according to their $f$-value.
4. For a specific $b \in B$, check if there is an $a \in A$ such that $(a, b)$ is a claw by using classical binary search on the sorted version of $A$. Combine this with quantum search on the elements of $B$ to find a claw in $A \times B$.
5. Apply amplitude amplification on steps 1-4.

In Algorithm 1, Step 3 requires classical sorting with $O(l \cdot \log l)$ comparisons and Step 4 needs $O(\sqrt{B} \cdot \log |A|) = O(l \log l)$ comparisons, since checking if there is an $A$-element colliding with a given $b \in B$ takes $O(\log |A|) = O(\log l)$ comparisons via binary search on the sorted $A$, and quantum search needs $O(\sqrt{|B|}) = O(l)$ to find a $B$-element that collides with an element occurring in $A$. In general, the time complexity of steps 1–4 is $O(l \log l)$. The probability of getting one claw $(x, y) \in A \times B$ in the first four steps is $p = l^3/2NM$. Therefore, the amplitude amplification of Step 5 requires an expected $O(\sqrt{NM/l^3})$ iterations of steps 1–4. In total, the time complexity of steps 1–5 is $O(\sqrt{NM/l^3} \log l)$. And it consumes $O(l \cdot \log(l))$ qubits.
When $N \leq M \leq N^2$, $O(\sqrt{NM \log l}) \approx O(N^{1/2}M^{1/4} \log N)$. And when $M > N^2$, $O(\sqrt{NM \log l}) \approx O(M^{1/2} \log N)$.

**Quantum random access memory.** A quantum random access memory (QRAM) is a quantum analogue of a classical random access memory (RAM), which uses $n$-qubit to address any quantum superposition of $2^n$ (quantum or classical) memory cells [12]. The QRAM is modeled as an unitary transformation $U_{QRAM}$ such that

$$\sum_i a_i |i\rangle_{addr} \xrightarrow{U_{QRAM}} \sum_i a_i |i\rangle_{addr} |D_i\rangle_{data},$$

where $\sum_i a_i |i\rangle_{addr}$ is a superposition of addresses and $D_i$ is the content of the $i$th memory cell.

### 3 Overview of classical meet-in-the-middle attack on 6-round feistel constructions

Our proposed quantum meet-in-the-middle attack on Feistel constructions is based on Guo’s work [14], so we first briefly introduce the framework of Guo’s attack.

#### 3.1 Attack idea

The MITM attack generally consists of the distinguisher and the key-recovery parts as illustrated in Fig. 2. Suppose that a truncated differential is specified to the entire cipher and the plaintext difference $\Delta P$ propagates to the input difference $\Delta X$ of the distinguisher with probability $p_1$. Similarly, from the other direction, the ciphertext difference $\Delta C$ propagates to the output difference $\Delta Y$ of the distinguisher with probability $p_2$. Generally, the attack consists of two phases: precomputation and queried-data analysis.

In the pre-computation phase, the adversary firstly enumerates all the possible differential characteristics that can satisfy the truncated differential of the distinguisher. Suppose that there exist $N_c$ such characteristics. Let $(X, X')$ be the input pair values for each characteristic with difference $\Delta X$. Then, the adversary generates a $\delta$-set that contains $X_1, X_2, \ldots, X_\delta$, where $X_i = X \oplus i$ $(i = 1, 2, \ldots, \delta)$. Let $Y_1, Y_2, \ldots, Y_\delta$ be the corresponding values at the output of the distinguisher. And the differences $\Delta_i$ between $Y$ and $Y_i$ for $i = 1, 2, \ldots, \delta$ make up a sequence, called $\Delta$-sequence ($\Delta$-sequence $= \Delta_1 || \Delta_2 || \cdots || \Delta_\delta$). Note that the size of the difference $\Delta_i$ is $\lambda$ bits. In the end, $N_c \Delta$-sequences of the size $\lambda\delta$ bits would be stored in Table $T_\delta$.

In the queried-data analysis phase, the adversary collects $(p_1p_2)^{-1}$ pairs of plaintexts with the difference $\Delta P$ and their corresponding ciphertexts with the difference $\Delta C$. One pair of collected pairs, with high probability, satisfies $\Delta X$ and $\Delta Y$ at the input and output of the distinguisher, respectively. Thus, for each of $(p_1p_2)^{-1}$ paired values, the adversary guesses subkeys for the key recovery rounds such that $\Delta X$
Quantum meet-in-the-middle attack on Feistel construction

and $\Delta Y$ appear after the first and the last key recovery parts, respectively. Then, for each pair and the guessed subkeys, $P$ is modified to $P_0$ and the other $P_1, P_2, \ldots, P_\delta$ are computed by generated $\delta$-set and the guessed subkeys. And these plaintexts are queried to obtain their corresponding ciphertexts. Next, the adversary partially decrypt these ciphertexts with the guessed subkeys, and the $\Delta$-sequence is computed at the output of the distinguisher. Finally, those $\Delta$-sequences are matched the table $T_\delta$, if the analyzed pair is a right pair and the guessed subkeys are correct, then a match will be found. Otherwise, a match will not be found as long as $(p_1 p_2)^{-1} N_c \times 2^{-\lambda \delta} \ll 1$.

3.2 Application on 6-round feistel construction

Pre-computation phase. Guo et al. [14] give the 5-round distinguisher, which is illustrated in Fig. 3(a). The input and output differences of the 5-round distinguisher are defined as $0||X$ and $X'||0$, where $X, X' \in \{0, 1\}^{n/2}, X \neq X'$ and the block size is $n$. For a given $X, X'$, the 5-round differential characteristics can be fixed to

\[
(0||X) \xrightarrow{1st R} (X||0) \xrightarrow{2nd R} (Y||X) \xrightarrow{3rd R} (X'|||Y) \xrightarrow{4th R} (0||X') \xrightarrow{5th R} (0||X'),
\]

where $Y$ represents the output difference of the 2nd round-function $F_{i+2}$ and has $2^{n/2}$ possible values. And the output difference of the 3rd round-function $F_{i+3}$ is $X''(X'' = X \oplus X')$.

For one choice of $X, X'$, the number of the 5-round differential characteristics satisfying such input and output differences is $2^{n/2}$, i.e., the number of corresponding $\Delta$-sequence for the left-half of the distinguisher’s output is $2^{n/2}$ proved by the Proposition 1 in Ref. [14]. To compute $\Delta$-sequence, we need to know the input values of
the middle 3 rounds round-functions \(F_{i+2}, F_{i+3}, \) and \(F_{i+4}\). According to the proof of Lemma 1 in Ref. [14], for each \(Y\), both input and output differences of \(F_{i+2}, F_{i+3}, \) and \(F_{i+4}\) are fixed, which suggests that the paired values during the round function are fixed to one choice on average. Thus, the adversary needs to construct three tables that record the input values of \(F_{i+2}, F_{i+3}, \) and \(F_{i+4}\), for each \(Y\) and one given \(X, X'\), respectively. Finally, the computed \(\Delta\)-sequences are stored in Table \(T_\delta\) for each \(Y\) and one given \(X, X'\), whose complexity is \(2^{n/2}\) in both time and memory.

To balance the complexities between the pre-computation phase and the queried-data analysis phase, Guo et al. iterate the above analysis for \(2^{n/4}\) different choices of \(X'\). They assume that the values of \(X'\) differ in the last \(n/4\) bits and are the same in the remaining \(n/4\) bits. Hence, the entire complexity of the pre-computation phase is \(O(2^{3n/4})\) in both time and memory.

4.1 Pre-computation phase

Similar to the classical MITM attack on 6-round Feistel, we need to calculate the \(\Delta\)-sequence based on the distinguisher and choose a fixed \(X\) value. But, \(2^{n/3}\) choices...
Fig. 3 (a) The 5-round distinguisher, (b) 6-round Feistel construction for the key recovery, (c) 7-round Feistel construction for the key recovery. $v_i$ and $F_i$ represent the input value and the round-function of each round, respectively. $X'' = X' \oplus X$ and $X' \neq X$. $Z$ is the difference at $v_7$

Table 2 Variables and notations

| Variables and notations | Explanations |
|-------------------------|--------------|
| $n$                     | Both the block length and the subkey size are $n$ bits |
| $X, Y, X', Z$           | $X$ (resp. $Y$) is the right (resp. left) half input difference of the 1st (resp. 3rd) round in the 5-round distinguisher. And $X'$ is the right half output difference of the distinguisher. $Z$ is the output difference at $v_7$ |
| $F_i, F^T_i, F^O_i$     | $F_i$ represents the $i$th round function, and $F^T_i$ (resp. $F^O_i$) indicates the input (resp. output) value of the $i$th round function |
| $t_{i+2}, t_{i+3}, t_{i+4}$ | $t_{i+2}, t_{i+3}$ and $t_{i+4}$ are the input values of $F_{i+2}, F_{i+3}$ and $F_{i+4}$, respectively |
| $k_i$                   | $k_i$ is the $n/2$-bit subkey of the $i$th round |
| $v_i, v_{i-1}$          | $v_i$ (resp. $v_{i-1}$) represents the left (resp. right) half input of the $i$th round in the cipher |
| $\Delta$                | $\Delta$ is loaded in front of the variable as a prefix, indicating the difference of the variable |
| $\Delta$-sequence      | $\Delta$-sequence is composed of $\delta$ differences at $v_{i+5}$ for each $X'$ and $Y$ |
| $T_\delta$              | $T_\delta$ is a lookup table constructed in pre-computation phase, which consists of $\Delta$-sequences |
| $P, C$                  | $P$ (resp. $C$) represents the plaintext (resp. ciphertext) |

of $X'$ are considered, to balance the complexities between the pre-computation phase and the queried-data analysis phase. Without loss of generality, assume that the values of $X'$ differ in the last $n/3$ bits and are zero in the remaining $n/6$ most significant bits (MSBs). We define a function $\mathcal{F} : X' \times Y \rightarrow \Delta$-sequence, where $X', Y \in \{0, 1\}^{n/2}$, $0 \leq X' \leq 2^{n/3} - 1$, $0 \leq Y \leq 2^{n/2} - 1$ and $\Delta$-sequence $\in \{0, 1\}^{\delta n/2}$. 

\textregistered Springer
For a given $X'$ and $Y$, assume that a pair of plaintexts $(m, m \oplus 0||X)$ satisfy the differential characteristic of the distinguisher, $t_{i+2}$, $t_{i+3}$ and $t_{i+4}$ are the input values of $F_{i+2}$, $F_{i+3}$ and $F_{i+4}$. Let us consider a new pair of plaintexts $(m, m \oplus 0||j)(j \in \{1, 2, \ldots, \delta\})$ and compute the corresponding $\Delta_j$ that is the difference at $v_{i+5}$.

Since the output difference $\Delta F^O_{i+1}$ of the first round-function $F_{i+1}$ in the distinguisher is always 0. So, $\Delta v_{i+2} = \Delta v_i = j$. In the second round, due to the input value $F^T_{i+2} = t_{i+2}$ and $\Delta F^T_{i+2} = j$, then $\Delta F^O_{i+2} = F_{i+2}(t_{i+2}) \oplus F_{i+2}(t_{i+2} \oplus j)$. And in the 3rd round, $\Delta F^O_{i+3} = F_{i+3}(t_{i+3} \oplus \Delta F^O_{i+2})$ because of $F^T_{i+3} = t_{i+3}$ and $\Delta F^T_{i+3} = \Delta F^O_{i+2}$. In the next round, $\Delta v_{i+4} = \Delta F^O_{i+3} \oplus j$, the output difference of 4th round is $\Delta F^O_{i+4} = F_{i+4}(t_{i+4}) \oplus F_{i+4}(t_{i+4} \oplus \Delta F^O_{i+3} \oplus j)$. Finally, we can get the difference $\Delta_j$ at $v_5$,

$$\Delta_j = \Delta v^j_{i+5}$$
$$= \Delta F^O_{i+4} \oplus \Delta F^O_{i+2}$$
$$= F_{i+4}(t_{i+4}) \oplus F_{i+4}(t_{i+4} \oplus \Delta F^O_{i+3} \oplus j) \oplus \Delta F^O_{i+2}$$

$$= F_{i+4}(t_{i+4}) \oplus F_{i+4}(t_{i+4} \oplus \Delta F^O_{i+3} \oplus j) \oplus F_{i+2}(t_{i+2}) \oplus F_{i+2}(t_{i+2} \oplus j)$$

By repeating this procedure for different choices of $j$, we can get

$$\Delta - \text{sequence} = \Delta_1||\Delta_2||...||\Delta_\delta$$

for a given $X'$ and $Y$.

From Eq. 3, to calculate $\Delta_j$, we need to know the values of $t_{i+2}$, $t_{i+3}$ and $t_{i+4}$ corresponding to the given $(X, X', Y)$. According to Lemma 1 in Ref. [14], there exists one state value (one solution) that satisfies such input–output difference in each of the middle three rounds for a given $(X, X', Y)$. As $Y$ takes at most $2^{n/2}$ different values, $t_{i+2}$, $t_{i+3}$ and $t_{i+4}$ can assume only $2^{n/2}$ different values. Therefore, we need to search for eligible $t_{i+2}$, $t_{i+3}$ and $t_{i+4}$ from a space of size $2^{n/2}$ for a given $(X, X', Y)$, where $t_{i+2}$, $t_{i+3}$ and $t_{i+4}$ satisfy

$$F_{i+2}(t_{i+2}) \oplus F_{i+2}(t_{i+2} \oplus X) = Y,$$
$$F_{i+3}(t_{i+3}) \oplus F_{i+3}(t_{i+3} \oplus Y) = X \oplus X',$$

So far, the above process is the calculation process of the function $F$. In the classical setting, the final $\Delta$-sequences are stored in a table $T_\delta$ with $O(2^n)$ computations and $O(2^n)$ classical memory. With quantum parallelism and superposition, we can reduce the time complexity to $O(2^{n/4})$ with $O(n)$ qubits, which is shown as below.
1. Prepare the superposition state

\[ |\varphi_1\rangle = \sum_{X' = 0}^{2^{n/3} - 1} \frac{1}{\sqrt{2^{n/3}}} |X'\rangle_1 \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3. \] (6)

where the bit length of \( X' \) is \( n/2 \) and \( X' \neq X \).

2. Do Grover search for \( t_{i+2}, t_{i+3} \) and \( t_{i+4} \) which satisfy Eq. 5 for each \((X', Y)\), and store them in the new registers, i.e.,

\[
\begin{align*}
&\sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3 \sum_{t_{i+2} = 0}^{2^{n/2} - 1} |t_{i+2}\rangle_4 \rightarrow \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3 |t_{i+2}\rangle_4, \\
&\sum_{X' = 0}^{2^{n/3} - 1} \frac{1}{\sqrt{2^{n/3}}} |X'\rangle_1 \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3 \sum_{t_{i+3} = 0}^{2^{n/2} - 1} |t_{i+3}\rangle_5 \rightarrow \\
&\sum_{X' = 0}^{2^{n/3} - 1} \frac{1}{\sqrt{2^{n/3}}} |X'\rangle_1 \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3 \sum_{t_{i+3} = 0}^{2^{n/2} - 1} |t_{i+3}\rangle_5, \\
&\sum_{X' = 0}^{2^{n/3} - 1} \frac{1}{\sqrt{2^{n/3}}} |X'\rangle_1 \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |X\rangle_3 \sum_{t_{i+4} = 0}^{2^{n/2} - 1} |t_{i+4}\rangle_6 \rightarrow \\
&\sum_{X' = 0}^{2^{n/3} - 1} \frac{1}{\sqrt{2^{n/3}}} |X'\rangle_1 \sum_{Y = 0}^{2^{n/2} - 1} \frac{1}{\sqrt{2^{n/2}}} |Y\rangle_2 |t_{i+4}\rangle_6. \quad (7)
\end{align*}
\]

And we obtain

\[ |\varphi_2\rangle = \sum_{X', Y = 0}^{2^{n/3} - 1, 2^{n/2} - 1} 2^{-5n/12} |X'\rangle_1 |Y\rangle_2 |X\rangle_3 |t_{i+2}\rangle_4 |t_{i+3}\rangle_5 |t_{i+4}\rangle_6 \] (8)

after Grover search with time \( O(2^{n/4}) \). Since registers 1–6 are independent and tensors at the beginning, searching for \( t_{i+2}, t_{i+3} \) and \( t_{i+4} \) would be independent of each other, although the result is an entangled state.

3. Compute \( \Delta\text{-}sequence \) according to Eq. 3 with \( \otimes \sum_{j=1}^{\delta} |j\rangle \) as input, and store them in a new register of \( \delta n/2 \) qubits.

\[
|\varphi_3\rangle = \sum_{X', Y = 0}^{X'=2^{n/3} - 1, Y=2^{n/2} - 1} 2^{-5n/12} |X'\rangle_1 |Y\rangle_2 |X\rangle_3 |\Delta\text{-}sequence\rangle_7 \]

\[
|\varphi_3\rangle = \sum_{X', Y = 0}^{X'=2^{n/3} - 1, Y=2^{n/2} - 1} 2^{-5n/12} |X'\rangle_1 |Y\rangle_2 |X\rangle_3 |\mathcal{F}(X', Y)\rangle_7, \quad (9)
\]
where we perform some reverse operations and discard some useless registers finally.

4. Do Grover search parallelly on \(|\psi_3\rangle\) for each \((X', Y)\) with \(2^{5n/6}\) parallel single quantum processors and store the measured \(|\Delta - \text{sequence}\rangle_7\) in the table \(T_\delta\) indexed by \((X', Y)\).

In summary, the entire quantum computation for the table \(T_\delta\) will cost \(O(2^{5n/12} \cdot n^2/4) \approx O(2^{2n/3})\) time according to the steps 2 and 4, with \(O(2^{5n/6} \cdot n)\) qubits and \(O(2^{5n/6})\) classical memory.

4.2 Queried-data analysis phase

Firstly, we choose two plaintext sets in the form of \(\{m||0, m||1, ..., m||2^{n/2} - 1\}\) and \(\{m \oplus X||0, m \oplus X||1, ..., m \oplus X||2^{n/2} - 1\}\), where \(m\) is a randomly chosen \(n/2\)-bit constant. \(2^n\) pairs of plaintexts can be generated, \(2^{5n/6}\) pairs of them will satisfy \(\Delta C = X'||Z\) in the corresponding ciphertexts for \(2^{n/3}\) choices of \(X'\) and \(2^{n/2}\) choices of \(Z\). By iterating this procedure \(2^{n/6}\) times for different choices of \(m\), \(2^n\) pairs satisfying \(\Delta P = X||*\) and \(\Delta C = X'||Z\) are collected. These pairs are stored in the table \(T_{PC}\) indexed by the item \(I\) (\(0 \leq I \leq 2^n - 1\)). We can see that the probability of that the plaintext difference \(\Delta P\) propagates to the input difference \(0||X\) of the distinguisher is \(p_1 = 2^{-n/2}\), and the probability of the ciphertext difference \(\Delta C\) propagates to the output difference \(X'||0\) is \(p_2 = 2^{-n/2}\). Therefore, one pair of \(2^n\) pairs would satisfy the entire differential characteristic of 7-round Feistel.

In this phase, we aim to compute the \(\Delta\)-sequence for each pair, and we define this computing process as a function \(\mathcal{G} : I \rightarrow \Delta\text{-sequence}\), where \(0 \leq I \leq 2^n - 1\) and \(\Delta\text{-sequence} \in \{0, 1\}^{5n/2}\).

For each pair, we should guess subkeys \(k_0\) and \(k_6\) to ensure that the differences \(\Delta P\) and \(\Delta C\) propagate to the input difference \(0||X\) and the output difference \(X'||0\) of the distinguisher, respectively. In other words, we assume that the differences \(\Delta P\) and \(\Delta C\) for each pair propagate to \(0||X\) and \(X'||0\), respectively. In this case, we can find the input values of the two round functions \(F_0\) and \(F_6\) according to the known input and output differences and then use the input values with the known plaintext and ciphertext values to deduce the guessed subkeys, called subkey candidates.

Suppose that one pair of \(2^n\) pairs is \((P, P') = (v_0||v_{-1}, v_0 \oplus X||v_{-1} \oplus *)\) and their corresponding ciphertexts \((C, C') = (v_6||v_7, v_6 \oplus X'||v_7 \oplus Z)\). Then, we need to search the input values \(F_0^T\) and \(F_6^T\) according to the input and output differences. Therefore, we compute all \(2^{n/2}\) possible values of \(F_0^T\) and \(F_6^T\) by Eq. 10 and store them in two tables \(T_0\) and \(T_6\) indexed by \((X, *)\) and \((X', Z)\), respectively. This step will cost \(O(2^{5n/6})\) time classically.

\[
F_0(F_0^T) \oplus F_0(F_0^T \oplus X) = *, \quad F_6(F_6^T) \oplus F_6(F_6^T \oplus X') = Z. \tag{10}
\]

When we get \(F_0^T\) and \(F_6^T\), we can obtain two subkeys candidates \(k_0\) and \(k_6\) by

\[
k_0 = F_0^T \oplus v_0, \quad k_6 = F_6^T \oplus v_6. \tag{11}
\]
Then, we construct a \( \delta \)-set \( \{ v_1||v_0 \oplus 1, v_1||v_0 \oplus 2, \ldots, v_1||v_0 \oplus \delta \} \), where \( v_1||v_0 \) is corresponding to the plaintext \( v_0||v_{-1} \) and can be obtained by encrypting with the subkey candidate \( k_0 \). So, with the knowledge of subkey candidate \( k_0 \), we compute the corresponding \( \Delta F_0^O \), modify \( v_{-1} \) so that the value of \( v_1 \) stays unchanged, and obtain the plaintexts corresponding to the \( \delta \)-set, i.e.,

\[
\Delta F_0^O = F_0(v_0 \oplus k_0) \oplus F_0(v_0 \oplus k_0 \oplus j), \quad j \in \{1, 2, \ldots, \delta\}
\]

And these plaintexts are queried to get ciphertexts \( \{ C_1, C_2, \ldots, C_\delta \} \). With the knowledge of subkey candidate \( k_6 \), these ciphertexts are decrypted partially to get the values \( \{ v_5^1, v_5^2, \ldots, v_5^\delta \} \) at \( v_5 \). Finally, the differences between \( v_5 \) (corresponding to ciphertext \( C \)) and \( v_5^j \) make up a sequence \( \Delta\text{-sequence} \). If this \( \Delta\text{-sequence} \) can be found in Table \( T_\delta \), the subkey candidates \( k_0 \) and \( k_6 \) are correct. Note that, \( p_1 = 2^{-n/2}, p_2 = 2^{-n/2}, N_c = 2^{n/2}, \lambda = n/2 \); hence, \( \delta = 4 \) is sufficient to filter out all the wrong candidates.

In the classical setting, the computing process of \( G \) needs \( O(2^n) \) computations with \( O(2^n) \) classical memory. To reduce complexity, we do the following quantum operations.

1. Prepare the superposition state

\[
|\varphi_1\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1.
\]

2. Use \( |\varphi_1\rangle \) as the address to query the paired plaintexts and ciphertexts \( (P, P', C, C') \) from the table \( TP_C \) by QRAM,

\[
|\varphi_2\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |P, P', C, C'\rangle_2
\]

\[
= \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2,
\]

where \( P, P', C, C' \) is abbreviated as \( PC \).

3. Compute \( (X, \ast, X', Z) \) for each pair to obtain

\[
|\varphi_3\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, \ast, X', Z\rangle_3
\]
4. Do Grover search to find $F_0^I$ and $F_6^I$ satisfying Eq. 10 for each pair

$$\sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 \sum_{F_0=0}^{2^n/2-1} |F_0^I\rangle_4$$

$$\sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 |F_6^I\rangle_5$$

which costs $O(2^{n/4})$ time and obtains

$$|\varphi_4\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 |F_0^I\rangle_4 |F_6^I\rangle_5.$$  (17)

5. Compute the subkey candidates $k_0$ and $k_6$ with known plaintext $P$ and ciphertext $C$ according to Eq. 11, to get

$$|\varphi_5\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 |F_0^I\rangle_4 |F_6^I\rangle_5 |k_0\rangle_6 |k_6\rangle_7.$$  (18)

6. Compute the plaintexts corresponding to the $\delta$-set according to Eq. 12, to get

$$|\varphi_6\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 |F_0^I\rangle_4 |F_6^I\rangle_5 |k_0\rangle_6 |k_6\rangle_7 |P_1, P_2, ..., P_\delta\rangle_8.$$  (19)

7. Query the corresponding ciphertexts $\{C_1, C_2, ..., C_\delta\}$ for $\{P_1, P_2, ..., P_\delta\}$ by QRAM,

$$|\varphi_7\rangle = \sum_{I=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |PC\rangle_2 |X, *, X', Z\rangle_3 |F_0^I\rangle_4 |F_6^I\rangle_5 |k_0\rangle_6 |k_6\rangle_7$$

$$\otimes |P_1, P_2, ..., P_\delta\rangle_8 |C_1, C_2, ..., C_\delta\rangle_9.$$  (20)
Algorithm 2 Quantum Claw Finding between Two Functions $\mathcal{F}$ and $\mathcal{G}$

1. Select a random subset $A \subseteq \{(X', Y)\}$ of size $l (l \leq 2^{5n/6})$, where $|\{(X', Y)\}| = 2^{5n/6}$.
2. Select a random subset $B \subseteq \{I\}$ of size $l^2$.
3. Compute $\sum_{b \in B} \frac{1}{\sqrt{l}} \ket{b}_B \ket{\mathcal{G}(b)}$ according to Eqs. 13–21.
4. Sort the elements in $A$ according to their $\mathcal{F}$-value according to the table $T_\delta$.
5. For a specific $b \in B$, check if there is an $a \in A$ such that $(a, b)$ is a claw by using classical binary search on the sorted version of $A$. Combine this with quantum search on the elements of $B$ to find a claw in $A \times B$.
6. Apply amplitude amplification on steps 1-4 with $\sqrt{\frac{2^{5n/6}+n}{l^3}}$ iterations. And output the claw $((X', Y), I)$ such that $\mathcal{F}(X', Y) = \mathcal{G}(I)$.

Note that another table $T'_{PC}$ stores $2^{2n/3+1+\delta}$ plaintexts and corresponding ciphertexts, which is indexed by the plaintexts. So, we perform $\delta$ times of
\[
\sum_{l=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |P_i\rangle_8 \xrightarrow{U_{QRAM}} \sum_{l=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |P_i\rangle_8 |C_i\rangle_9 (1 \leq i \leq \delta).
\]

8. Compute the $\Delta$-sequence with the knowledge of subkey candidate $k_6$ and ciphertexts $\{C_1, C_2, ..., C_\delta\}$, to obtain
\[
|\varphi\rangle = \sum_{l=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |\Delta-sequence\rangle_10
\]
\[
= \sum_{l=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |\mathcal{G}(I)\rangle_10,
\]
where we perform some reverse operations and discard some useless registers.

For now, we need to find a claw $((X', Y), I)$ such that $\mathcal{F}(X', Y) = \mathcal{G}(I)$ from the table $T_\delta$ and the superposition state $\sum_{l=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |I\rangle_1 |\mathcal{G}(I)\rangle_10$, which can be implemented by Algorithm 2. When we find the claw $((X', Y), I)$, the corresponding subkeys candidates $k_0$ and $k_6$ are correct, and the other subkeys can be deduced with the knowledge of $X$, $X'$ and $Y$.

**Complexity.** Obviously, our attack needs $O(2^{2n/3})$ classical queries, which means that the data complexity is $O(2^{2n/3})$ and our attack belongs to Q1 model. And the time complexities of constructing the table $T_\delta$ and Algorithm 2 are $O(2^{2n/3})$ and $O(2^{2n/3} \cdot n)$, respectively. Therefore, the total time complexity of our attack is $O(2^{2n/3} \cdot n)$. Besides, our attack needs $O(2^{5n/6} + 2^n + 2^{2n/3+1+\delta}) \approx O(2^n)$ classical memory due to the tables $T_\delta$, $T_{PC}$ and $T'_{PC}$. And it consumes $O(2^{5n/6} \cdot n + 2^{n/2} \cdot n) \approx O(2^{5n/6} \cdot n)$ qubits for constructing $T_\delta$ and performing Algorithm 2 with $l = 2^{n/2}$.

**Discussion.** To find a collision (a claw) in one table with $N$ elements, Buhrman’s algorithm we used in this paper needs $O(N^{3/4})$ time, but Ambainis’s algorithm [1] only needs $O(N^{2/3})$ time. Someone may wonder why we don’t choose the optimal quantum claw finding algorithm. Because, we do not have $\mathcal{F}$-values and $\mathcal{G}$-values before
the attack and we need much time to compute their values even in the quantum setting. If we use Ambainis’s algorithm, we need to consider the time of computing $F$-values and $G$-values in each iteration of quantum walk, and the total time of finding a claw will be higher than Algorithm 2. The specific process of Ambainis’s algorithm is not shown here. For details, please refer to Ref. [1]. Buhrman’s algorithm is more flexible than Ambainis’s algorithm. In Sect. 4.1, we compute the $F$-values with quantum computation and store them classically. We can even compute $F$-values classically when $r > 7$ (see next section). For computing $G$-values, we treat the process as part of each iteration in Algorithm 2, like step 3. Its time complexity is $O(2^{n/4})$, and step 5 needs $O(2^{n/2} \cdot n)$ when $l = 2^{n/2}$ ($l \leq \min\{2^{5n/6}, \sqrt{2^n}\}$). Therefore, the time of computing $G$-values in each iteration can be negligible.

5 Quantum Meet-in-the-Middle Attack on $r$-round Feistel Construction

To attack $r$-round ($r > 7$) Feistel construction, we guess the subkeys of the last $r - 7$ rounds to compute $G$-values with queried plaintexts and ciphertexts and then to find a claw between the functions $F$ and $G$. If we find this claw, then the guessed subkeys are correct.

Pre-computation phase. Similar to the computing process in Sect. 4.1, we can construct the table $T_\delta$ classically without quantum computation. It costs $O(2^{5n/6})$ time and $O(2^{5n/6})$ classical memory. Or, it costs $O(2^{n/3})$ time, $O(2^{5n/6})$ memory and $O(2^{5n/6} \cdot n)$ qubits with quantum computation as same as Sect. 4.1.

Queried-data analysis phase. We choose the same paired plaintexts in Sect. 4.2 and query for their ciphertexts, which are also stored in the table $TCP_{\text{all}}$ indexed by $I$ ($0 \leq I \leq 2^{7n/6} - 1$). After querying, we do following operations.

1. Prepare the initial superposition state

$$|\varphi_1\rangle = \sum_{K=0}^{2^{(r-7)n/2}} \frac{1}{\sqrt{2^{(r-7)n/2}}} |K\rangle_1 \sum_{I=0}^{2^{7n/6}-1} \frac{1}{\sqrt{2^{7n/6}}} |I\rangle_2, \quad (22)$$

where $K = (k_7, k_8, \ldots, k_{r-1})$.

2. Query the paired plaintexts and ciphertexts from the table $TCP_{\text{all}}$ by QRAM,

$$|\varphi_2\rangle = \sum_{K=0}^{2^{(r-7)n/2}} \frac{1}{\sqrt{2^{(r-7)n/2}}} |K\rangle_1 \sum_{I=0}^{2^{7n/6}-1} \frac{1}{\sqrt{2^{7n/6}}} |I\rangle_2 |PC-all\rangle_3. \quad (23)$$

3. Decrypt the ciphertexts partially to the values at $v_7||v_6$ for each $K$,
\[ |\varphi_3\rangle = \sum_{K=0}^{2^{(r-7)n/2}} \frac{1}{\sqrt{2^{(r-7)n/2}}} |K\rangle_1 \sum_{l=0}^{2^{7n/6}-1} \frac{1}{\sqrt{2^{7n/6}}} |I\rangle_2 |PC - all\rangle_3 |C_{v7||v6}\rangle_4, \]

(24)

where \( C_{v7||v6} \) represents the values at \( v7||v6 \).

4. Do Grover search for that the difference \( \Delta \) of the corresponding \( C_{v7||v6} \) is all zero in the first \( n/6 \) bits, with \( O(2^{n/12}) \) time. Only these ciphertexts and their corresponding plaintexts are what we need.

\[ |\varphi_4\rangle = \sum_{K=0}^{2^{(r-7)n/2}} \frac{1}{\sqrt{2^{(r-7)n/2}}} |K\rangle_1 \otimes \sum_{0 \leq l' \leq 2^{7n/6}-1, |l'\rangle = 2^n} \frac{1}{\sqrt{2^n}} |I\rangle_2 |PC - all\rangle_3 |C_{v7||v6}\rangle_4 \]

(25)

5. For now, we just need to find a claw between \( \mathcal{F}(x) \) and \( \mathcal{G}(y) \), where \( x \in \{(X', Y)\} |\{(X', Y)\} = 2^{5n/6} \) and \( y \in \{(K, I')\} |\{(K, I')\} = 2^{n+(r-7)n/2} \).

(a) when \( r = 8 \), \( |\{(K, I')\}| = 2^{3n/2} \), and \( 2^{5n/6} < 2^{3n/2} < \left(2^{5n/6}\right)^2 \). Then, \( l = \min(2^{5n/6}, 2^{3n/4}) = 2^{3n/4} \) in Algorithm 1. The complexity of step 4 in Algorithm 1 is \( 2^{3n/4} \cdot 3/4n \). Since the complexity of computing \( \mathcal{G}(y) \) in Step 2 is \( 2^{n/3} = 2^{n/12+3n/8} \) according to Eqs. 13–21 and 25, it does not influence the entire complexity of Algorithm 1 due to \( 2^{n/3} < 2^{3n/4} \cdot 3/4n \). The time complexity of finding a claw is \( 2^{5n/12+3n/8} \cdot 5/6n = 2^{19n/24} \cdot 5/6n \).

i. If we construct the table \( T_5 \) classically without quantum computation, the total time complexity of our attack is \( O(2^{19n/24} \cdot 5/6n + 2^{5n/6} + 2^{2n/3}) \approx O(2^{5n/6}) \). And its memory is \( O(2^{7n/6} + 2^{5n/6}) \approx O(2^{7n/6}) \) for storing \( T_{PC-all} \) and \( T_5 \) and consumes \( O(2^{3n/4} \cdot n) \) qubits.

ii. Otherwise, its time complexity is \( O(2^{19n/24} \cdot 5n + 2^{2n/3}) \approx O(2^{19n/24} \cdot n) \), and its memory is \( O(2^{5n/6} \cdot n + 2^{3n/4} \cdot n) \approx O(2^{5n/6} \cdot n) \) qubits.

(b) when \( r > 8 \), \( 2^{(r-5)n/2} > \left(2^{5n/6}\right)^2 \). Then \( l = 2^{5n/6} \). The time of finding a claw is \( O(2^{(r-5)n/4} \cdot n) \). Therefore, the time complexity of our attack is \( O(2^{(r-5)n/4} \cdot n + 2^{5n/6} + 2^{2n/3}) \approx O(2^{(r-5)n/4} \cdot n) \). And its memory is \( O(2^{7n/6} + 2^{5n/6}) \approx O(2^{7n/6}) \) for storing \( T_5 \) and \( T_{PC-all} \) and consumes \( O(2^{5n/6} \cdot n) \) qubits.

Note that \( p_1 = 2^{-n/2} \), \( p_2 = 2^{-2n/3} \), \( N_c = 2^{n/2} \); hence, \( \delta = 4 \) is sufficient to filter out all the wrong candidates. Regardless of the value \( r \ (r \geq 7) \), the data complexity is \( O(2^{2n/3}) \).

6 Conclusion

To reduce the time complexity of classic and quantum attacks on \( r \)-round \( (r \geq 7) \) Feistel construction, we propose a new quantum meet-in-the-middle attack in Q1
model, which combines a quantum claw finding algorithm and Guo et al.’s meet-in-the-middle attack. Its time complexity only needs $O(2^{2n/3} \cdot n)$, $O(2^{19n/24} \cdot n)$ and $O(2^{(r−5)n/4} \cdot n)$ when $r = 7$, $r = 8$ and $r > 8$. Moreover, it belongs to the Q1 model and is more practical than other quantum attacks.

Furthermore, we hope to carry out quantum meet-in-the-middle attacks on more multi-round Feistel constructions. Because there are not only 5-round distinguisher, but also 7-round distinguisher, 8-round distinguisher, etc., combining these distinguishers with quantum claw finding algorithms, or even other quantum algorithms may achieve good attack results.

Acknowledgements This work was supported by the Open Fund of Advanced Cryptography and System Security Key Laboratory of Sichuan Province (Grant No. SKLACSS-202103), BUPT Excellent Ph.D. Students Foundation (CX2022140) and the 13th Five-Year Plan National Cryptography Development Fund (MMJJ20180217).

Data Availability All data generated or analyzed during this study are included in this published article.

References

1. Ambainis, A.: Quantum walk algorithm for element distinctness. SIAM J. Comput. 37(1), 210–239 (2007). https://doi.org/10.1137/S0097539705447311
2. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for step-reduced sha-2. In: Matsui, M. (ed.) Advances in Cryptology - ASIACRYPT 2009, pp. 578–597. Springer, Berlin (2009)
3. Bernstein, E., Vazirani, U.: Quantum complexity theory. SIAM J. Comput. 26(5), 1411 (1997)
4. Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: On quantum slide attacks. In: Paterson, K.G., Stébila, D. (eds.) Selected Areas in Cryptography - SAC 2019, pp. 492–519. Springer International Publishing, Cham (2020)
5. Buhrman, H., Dürr, C., Heiligman, M., Høyer, P., Magniez, F., Santha, M., de Wolf, R.: Quantum algorithms for element distinctness. SIAM J. Comput. 34(6), 1324–1330 (2005). https://doi.org/10.1137/S0097539702402780
6. Coppersmith, D.: The data encryption standard (des) and its strength against attacks. IBM J. Res. Dev. 38(3), 243–250 (1994). https://doi.org/10.1147/rd.383.0243
7. Demirci, H., Selçuk, A.A.: A meet-in-the-middle attack on 8-round aes. In: Nyberg, K. (ed.) Fast Software Encryption, pp. 116–126. Springer, Berlin (2008)
8. Dong, X., Dong, B., Wang, X.: Quantum attacks on some feistel block ciphers. Des. Codes Crypt. 88, 1–25 (2020)
9. Dong, X., Li, Z., Wang, X.: Quantum cryptanalysis on some generalized feistel schemes. Sci. China Inf. Sci. 62(2), 25201 (2019)
10. Dong, X., Wang, X.: Quantum key-recovery attack on feistel structures. Sci. China Inf. Sci. 61(10), 1–7 (2018)
11. Feistel, H., Notz, W., Smith, J.: Some cryptographic techniques for machine-to-machine data communications. Proc. IEEE 63(11), 1545–1554 (1975). https://doi.org/10.1109/PROC.1975.10005
12. Giovannetti, V., Lloyd, S., Maccone, L.: Quantum random access memory. Phys. Rev. Lett. 100, 160501 (2008). https://doi.org/10.1103/PhysRevLett.100.160501
13. Grover, L.K.: Quantum computers can search arbitrarily large databases by a single query. Phys. Rev. Lett. 79(23), 4709 (1997)
14. Guo, J., Jean, J., Nikolić, I., Sasaki, Y.: Extended meet-in-the-middle attacks on some feistel constructions. Des. Codes Crypt. 80(3), 587–618 (2016)
15. Hosoyamada, A., Sasaki, Y.: Quantum demiric-selçuk meet-in-the-middle attacks: applications to 6-round generic feistel constructions. In: International conference on security and cryptography for networks, pp 386–403. Springer (2018)
16. Isobe, T., Shibutani, K.: All subkeys recovery attack on block ciphers: extending meet-in-the-middle approach. In: Knudsen, L.R., Wu, H. (eds.) Selected Areas in Cryptography, pp. 202–221. Springer, Berlin (2013)
17. Isobe, T., Shibutani, K.: Generic key recovery attack on feistel scheme. In: Sako, K., Sarkar, P. (eds.) Advances in Cryptology - ASIACRYPT 2013, pp. 464–485. Springer, Berlin (2013)
18. ISO/IEC: Information technology–security techniques–encryption algorithms–part 3: block ciphers (2010)
19. Ito, G., Hosoyamada, A., Matsumoto, R., Sasaki, Y., Iwata, T.: Quantum chosen-ciphertext attacks against feistel ciphers. In: Matsui, M. (ed.) Topics in Cryptology - CT-RSA 2019, pp. 391–411. Springer International Publishing, Cham (2019)
20. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Breaking symmetric cryptosystems using quantum period finding. In: Crypto 2016-36th annual international cryptology conference, pp 207–237. Springer (2016)
21. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Quantum differential and linear cryptanalysis. IACR Trans. Symmet. Cryptol. 2016(1), 71–94 (2016). https://doi.org/10.13154/tosc.v2016.i1.71-94
22. Kuwakado, H., Morii, M.: Quantum distinguisher between the 3-round feistel cipher and the random permutation. In: 2010 IEEE international symposium on information theory, pp 2682–2685. IEEE (2010)
23. Kuwakado, H., Morii, M.: Security on the quantum-type even-mansour cipher. In: 2012 international symposium on information theory and its applications, pp 312–316. IEEE (2012)
24. Leander, G., May, A.: Grover meets simon–quantumly attacking the fx-construction. In: International conference on the theory and application of cryptography and information security, pp 161–178. Springer (2017)
25. Knudsen, L.R.: The security of feistel ciphers with six rounds or less. J. Cryptol. 15(3), 207–222 (2002). https://doi.org/10.1007/s00145-002-9839-y
26. Santoli, T., Schaffner, C.: Using simon’s algorithm to attack symmetric-key cryptographic primitives. arXiv preprint arXiv:1603.07856 (2016)
27. Shi, T., Jin, C., Guan, J.: Collision attacks against aez-prf for authenticated encryption aez. China Commun. 15(2), 46–53 (2018)
28. Simon, D.R.: On the power of quantum computation. SIAM J. Comput. 26(5), 1474–1483 (1997)
29. Todo, Y.: Upper bounds for the security of several feistel networks. In: Boyd, C., Simpson, L. (eds.) Information Security and Privacy, pp. 302–317. Springer, Berlin (2013)
30. Xie, H., Yang, L.: Using Bernstein-vazirani algorithm to attack block ciphers. Des. Codes Crypt. 87(5), 1161–1182 (2019)
31. Xu, Y., Liu, W., Yu, W.: Quantum forgery attacks on copsa, aec-copsa and marble authenticated encryption algorithms. Quantum Inf. Process. 20(4), 1–21 (2021)
32. Zhao, S., Duan, X., Deng, Y., Peng, Z., Zhu, J.: Improved meet-in-the-middle attacks on generic feistel constructions. IEEE Access 7, 34416–34424 (2019). https://doi.org/10.1109/ACCESS.2019.2900765

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.