Quantum Meet-in-the-Middle Attacks: Applications to Generic Feistel Constructions

Akinori Hosoyamada[1] and Yu Sasaki[1]

NTT Secure Platform Laboratories {hosoyamada.akinori,sasaki.yu}@lab.ntt.co.jp

Abstract. This paper shows that quantum computers can significantly speed-up a type of meet-in-the-middle attacks initiated by Demiric and Selçuk (DS-MITM attacks), which is currently one of the most powerful cryptanalytic approaches in the classical setting against symmetric-key schemes. The quantum DS-MITM attacks are then demonstrated against 6 rounds of the generic Feistel construction supporting an n-bit key and an n-bit block, which was attacked by Guo et al. in the classical setting with data, time, and memory complexities of $O(2^{3n/4})$. The complexities of our quantum attacks depend on the adversary's model and the number of qubits available to the adversary. When the adversary has an access to quantum computers for performing offline computations but online queries are made in a classical manner (so called Q1 model), the attack complexities are $O(2^{n/2})$ classical queries, $O(2^n/q)$ quantum computations by using about q qubits. Those are balanced at $\tilde{O}(2^{n/2})$, which significantly improves the classical attack. Technically, we convert the quantum claw finding algorithm to be suitable in the Q1 model, and apply it to reduce the cost of finding a match in the MITM stage. The attack is then extended to the case that the adversary is further allowed to make superposition queries (so called Q2 model). The attack approach is drastically changed from one in the Q1 model; the attack is based on 3-round distinguishers with Simon's algorithm and then appends 3 rounds for key recovery. This can be solved by applying the combination of Simon's and Grover's algorithms recently proposed by Leander and May.

Keywords: post-quantum cryptography · Demiric-Selçuk meet-in-the-middle attack · Feistel network · Grover's algorithm · claw finding algorithm · classical query model

1 Introduction

1.1 Background

Post-quantum cryptography is a hot topic in the current symmetric-key cryptographic community. It has been known that Grover's quantum algorithm [Gro96] and its generalized versions [BBHT98, BHMT02] reduce the cost of the exhaustive search on a k-bit key from 2^k to $2^{k/2}$. Whereas Grover's algorithm is quite generic, post-quantum security of specific constructions has also been evaluated, which includes key recovery attacks against Even-Mansour constructions [KM12], distinguishers against 3-round Feistel networks [KM10], key recovery attacks against multiple encryptions [Kap14], forgery attacks against CBC-like MACs [KLLN16a], key recovery attacks against FX constructions [LM17], and so on. Given those advancement of the quantum attacks, NIST announced that they take into account the post-quantum security in the coming standardization process for the light-weight cryptography [MBTM17]. It is now important to investigate how quantum computers can impact to the symmetric-key cryptography.

It is also possible to view the quantum attacks in approach-wise. That is, several researchers converted the well-known cryptanalytic approaches in the classical setting to

ones in the quantum setting. Several examples are quantum differential cryptanalysis [KLLN16b], quantum meet-in-the-middle attacks [Kap14, HS17], quantum universal forgery attacks [KLLN16a], and so on.

At the present time, one of the most powerful cryptanalytic approaches in the classical setting is a type of the meet-in-the-middle attacks initiated by Demiric and Selçuk [DS08]. The attacks are often called meet-in-the-middle attacks, while we call them the *DS-MITM attacks* in order to distinguish them from the simple and traditional meet-in-the-middle attacks that separate the attack target into two independent parts. The DS-MITM attacks are powerful. For example, one of the current best attacks against AES-128 is the DS-MITM attacks [DFJ13], which can often be applied to other SPN-based ciphers as well. The DS-MITM attacks are also effective against Feistel networks [GJNS14] and their variants [GJNS16]. Considering those facts, it is of great interest to check whether quantum computers can significantly speed-up the DS-MitM attacks.

A pioneering work of quantum attacks against symmetric-key cryptography by Kuwakado and Morii [KM12] and a remarkable work by Kaplan et al. [KLLN16a] demonstrate that security of symmetric-key primitives drops to a linear to the output size when adversaries are allowed to make quantum queries, in which the adversaries pass superposition states to oracles and receive the results also as superposition states. However, assuming such a strong ability may be too unrealistic. Security against the adversaries who only make queries through a classical network but have access to quantum computers in their local environment is more realistic. Kaplan et al. [KLLN16a] called the former and the latter settings *Q2 model* and *Q1 model*, respectively.

Given the above background, our target in this paper is a quantum version of the DS-MITM attacks. As a demonstration, we improve on the classical DS-MITM attack against generic 6-round Feistel constructions proposed by Guo et al. [GJNS14]. Our main focus is the Q1 model, while we also discuss further speed-up in the Q2 model.

1.2 Simple Quantum Attacks against Feistel Network

Before we explain the summary of our results, we explain that simple applications of the quantum attacks do not strongly impact to the security of the Feistel network. We start by introducing the target Feistel network analyzed in this paper.

Target Feistel Network. This paper presents cryptanalysis against Feistel network that is typically analyzed in the context of generic attacks. Namely, our target is a balanced Feistel network whose block size is n bits, and the round function consists of XORing an $n/2$-bit subkey followed a public function $F : \{0,1\}^{n/2} \mapsto \{0,1\}^{n/2}$. Subkeys in each round are independently chosen, thus the key size for r rounds is $nr/2$ bits. The public function F can be different in different rounds. To avoid making the paper unnecessarily complicated, we denote the public function in all rounds by an identical notation F.

Classical Attacks against Feistel Network. Generic attacks in the classical setting against the class of Feistel networks have been studied by many papers in various approaches; the impossible differential attack [Knu02], the all-subkeys recovery attack [IS12, IS13], the DS-MITM attack [GJNS14], the dissection attack [DDKS15], and so on. The number of attacked rounds depends on the assumed key size. Considering that the block size is n bits and thus the adversaries can obtain the full codebook with 2^n queries and memory, let us discuss the case that the adversaries can spend up to 2^n computations. In this setting, the best attack is the DS-MITM attack [GJNS14] that recovers the key up to 6 rounds with $O(2^{3n/4})$ complexities in all of data, time, and memory.

Application of Grover's Algorithm and Parallelization. The most simple quantum attack is applying Grover's algorithm [Gro96] to exhaustive key search. Let k denote the key length. With a quantum computer and Grover's algorithm, the exhaustive search can be performed in time $O(2^{k/2})$. Furthermore, if $O(n2^p)$ qubits are available to the adversary, the Grover search can be parallelized [GR03], and the cost of the exhaustive search is reduced in time $O(2^{(k-p)/2})$. Thus, by applying the parallelized Grover search to the r-round Feistel network, key recovery attacks can be performed in time $O(2^{nr/4-p/2})$ with $O(1)$ classical queries, using $O(n2^p)$ qubits.

For 6 rounds ($r = 6$), the key can be recovered in time $O(2^n)$, using $\tilde{O}(2^n)$ qubits. This does not have any advantages. Strictly speaking, the exhaustive search can be performed without guessing the last-round subkey, but the attack still does not have an advantage over the classical DS-MITM attack.

Application of Quantum Dissection Attacks. Consider an iterated block cipher, i.e., the cipher which is constructed as $E_k^r = E_{1,K_1} \circ E_{2,K_2} \circ \cdots \circ E_{r,K_r}$, where each E_i is an n-block cipher with m-bit key, and subkeys in $k = (K_1, \ldots, K_r)$ are independently chosen. E_k^r is an n-bit block cipher with mr-bit key, and the iterated construction is one of the simplest ways to handle a long key only by using a block cipher for short keys.

Kaplan proposed quantum meet-in-the-middle attacks and quantum dissection attacks to recovery the key against the iterated construction [Kap14]. For $r = 2$, the quantum meet-in-the-middle attack can recover the full key in time $O(2^{2m/3})$, using $O(2^{2m/3})$ qubits. For $r = 4$, the quantum dissection attack can recover the full key in time $O(2^{2m/3+n/2})$, using $O(2^{2m/3})$ qubits.

These attacks can be applied to Feistel networks, as Dinur et al. [DDKS15] applied the dissection attack to Feistel networks in the classical setting. For example, 6-round Feistel networks can be regarded as two iterations of the 3-round Feistel construction. Thus, applying the quantum meet-in-the-middle attack, we can recover the full key in time $O(2^n)$, using $O(2^n)$ qubits. Again, this approach does not have any advantage over the classical DS-MitM attack.

1.3 Our Contributions

In this paper, we show that quantum computers can significantly speed-up the DS-MITM attacks in both of the Q1 and Q2 models.

For the Q1 model (classical queries with quantum offline computations), to find a match between the offline and online phases, we need to solve a variant of claw finding problem. Normally, a claw between functions f' and g is defined to be a pair (x, y) such that $f'(x) = g(y)$, and there exist quantum algorithms [BHT97, Amb03, Zha05, Tan09] to find a claw assuming both of f' and g are quantum accessible. However, we need to find a pair (x, y) such that $f(x, y) = g(y)$, and g must be implemented in a classical manner in our Q1 model attack. Thus we describe a quantum algorithm to find a pair (x, y) that satisfies $f(x, y) = g(y)$ in such a situation.

We then apply the above algorithm in the Q1 model to improve the classical DS-MITM attack by Guo et al. [GJNS14] against the 6-round generic Feistel construction. The data complexity, or the number of classical queries, is reduced from $O(2^{3n/4})$ of the classical attack to $O(2^{n/2})$. The time complexity of our attack depends on the parameter q that corresponds to the number of qubits available to the attacker. In fact, the time complexity T is given by a tradeoff curve $Tq = 2^n$, where $q \leq 2^{n/2}$. Hence, the quantum attack outperforms the classical attack when $q > 2^{n/4}$. In particular, all parameters are balanced at $\tilde{O}(2^{n/2})$, which improves previous $O(2^{3n/4})$ in the classical setting.

We then further analyze the attack complexity against the 6-round generic Feistel construction in the Q2 model (quantum queries with quantum offline computations). The attack approach is quite different from the one in the Q1 model. We use the distinguisher

against 3-round Feistel network by Kuwakado and Morii [KM10] as a base, and then append 3 more rounds for key recovery. The 3-round distinguisher uses Simon's algorithm [Sim97] whereas the 3-round key recovery requires to use Grover's algorithm [Gro96]. The combination of those two algorithms has recently been studied by Leander and May [LM17], which leads to significant speed-up in our setting. In this attack, the time complexity and the data complexity D are always identical, which again depends on the parameter q. The actual tradeoff is $Tq^{1/2} = 2^{3n/4}$.

Tradeoffs between T and q, and between D and q in the classical setting and quantum settings are compared in Fig. 1. As shown in Fig. 1, the data complexity in two quantum

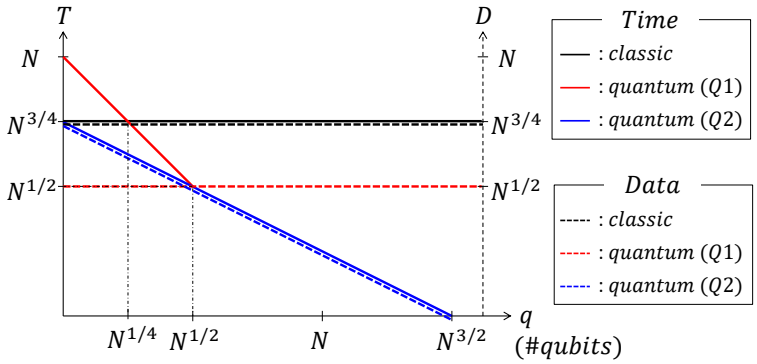

Figure 1: Time and Data Complexities in the Classical and Quantum Attacks (plotted in the logarithmic scale). N stands for 2^n. The complexities of the classical attack, which needs classical memory of size $O(2^{n/2})$, are independent from the number of qubits available.

settings are always better than one in the classical setting. At the very particular point $q = 2^{n/2}$, the time complexity in the Q1 and Q2 models match. For other q, the Q2 model generally outperforms the Q1 model.

1.4 Paper Outline

The paper is organized as follows. Section 2.1 explains attack models and quantum algorithms related to this work. Section 3 extends the previous quantum claw finding algorithm to the case that one function is evaluated only in the classical manner. Section 4 improves the previous DS-MitM attack against 6-round Feistel network by applying the theory in Sect. 3. Section 5 discusses the attack on Feistel network when the adversaries can make superposition queries.

2 Preliminaries

This section gives attack models and a summary of the quantum algorithms that are related to our work. Throughout the paper, we assume a basic knowledge of the quantum circuit model. For a public function $F : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$, we assume that a quantum circuit which calculates F, $C_F : |x\rangle |y\rangle \mapsto |x\rangle |y \oplus F(x)\rangle$ is available, and C_F runs in a constant time.

2.1 Attack Models

2.1.1 Online Query

As for an online oracle, which calculates an encryption of a keyed block cipher for example, we can consider two natural attack models when an adversary has a quantum computer. In the first model, an adversary can make only classical queries, whereas in the second model, the adversary is allowed to make quantum superposition queries. The former is called *Q1 model* and the latter is called *Q2 model* by Kaplan et al [KLLN16b] and Chailloux et al. [**?**].

In Q2 model, ability of an adversary is very strong, and in some cases the symmetric key cryptosystems that are proven to be secure in the classical setting are broken in polynomial time [KM10, KM12, KLLN16a, Bon17]. In addition, in Q2 model other symmetric key cryptosystems are also broken much faster than in the classical setting, though time complexity of the attack is still exponential [LM17]. Thus the Q2 model is a very interesting setting and new results in this setting give us new insight. However, it implicitly assumes that keyed functions are implemented on quantum circuits and data are communicated in quantum superposition states, which is relatively unrealistic.

On the other hand, Q1 model is the same as the classical model except that adversaries have a quantum computer, which is relatively realistic. Although Q1 model is weaker than Q2 model, we can still significantly speed up classical attacks [Kap14, KLLN16b, HS17].

2.1.2 Offline Quantum Computation

If we want to access some data or to operate table look-up in running a quantum circuit without any observation, we have to set all data on the quantum circuit so that data can be accessed in quantum superposition states. In particular, if we want to implement random access to stored data, we need as many qubits (or width of the quantum circuit) as the data size. Thus, quantum memory for random access is physically equivalent to quantum processor. In fact we regard them as identical.

Regardless of whether we use quantum computers or classical computers, the running time of an algorithm significantly depends on how a computational hardware is realized, when the algorithm needs exponentially many hardware resources. Thus we have to pay attention to data communication costs. In the quantum setting, Bernstein [Ber09] and Banegas and Bernstein [BB17] introduced two communication models, which they call *free communication model* and *realistic communication model*. The free communication model assumes that we can operate a unitary operation on any pairs of qubits. On the other hand, the realistic communication model assumes that 2^p qubits are arranged as a $2^{p/2} \times 2^{p/2}$ mesh, and a unitary operation can be operated only on a pair of qubits that are within a constant distance.

In this paper, for simplicity, we estimate the time complexity of quantum algorithms in the free communication model. A quantum hardware in the free communication model which has $O(N)$ qubits can simulate a quantum hardware in the free communication model which has $O(\sqrt{N})$ qubits, with time overhead $O(\sqrt{N})$ [BBG+13]. Note that this does not suggest that our proposed attacks do not work in the realistic communication model. We design our algorithms so that small quantum processors (of size polynomial in n) parallelly runs without any communication between each pair of small processors. Hence if the realistic communication model is applied, time complexity increases by a factor of polynomial in n.

2.2 Related Quantum Algorithms

2.2.1 Grover's Algorithm

Grover's quantum 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 database search algorithms. It was originally found by Grover [Gro96] and generalized later [BBHT98, BHMT02]. Let us consider the following problem:

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

Grover's algorithm (or its generalization) can solve the above problem with $O(2^{u/2})$ evaluations of ϕ using $O(u)$ qubits, if ϕ is given as a quantum oracle. The algorithm is composed of iterations of an elementary step which operates $O(1)$ evaluation of ϕ, and can easily be parallelized [GR04].

If we can use a quantum computer with $O(u2^p)$ qubits, we regard it as 2^p independent small quantum processors with $O(u)$ qubits. Then, by parallelly running $O(\sqrt{2^u/2^p})$ iterations on each small quantum processor, we can find x such that $\phi(x) = 1$ with high probability. This parallelized algorithm runs in time $O(\sqrt{2^u/2^p} \cdot T_\phi)$, where T_ϕ is the time needed to evaluate ϕ once.

2.2.2 Quantum Amplitude Amplification

There is an important generalization of the above Grover's algorithm, named *quantum amplitude amplification*, which was proposed by Brassard et al [BHMT02]. Let us consider Problem 1 below, and define "good subspace" G and "bad subspace" B by $G := \phi^{-1}(1)$ and $B := \phi^{-1}(0)$. Then the following proposition holds.

Proposition 1 (Quantum amplitude amplification [BHMT02])**.** *Suppose that there is a quantum algorithm $\mathcal{A}$ without observation such that $\mathcal{A}|0^u\rangle = \sum_{x\in G} \alpha_x |x\rangle + \sum_{x\in B} \beta_x |x\rangle$. Let $a := \sum_x \alpha_x^2$. We call the probability that we get a good element $x \in G$ when we observe the quantum state $\mathcal{A}|0^u\rangle$ good probability. Then, there is a quantum algorithm that runs in time $O(\sqrt{a}(T_\mathcal{A} + T_\phi))$ that finds a good element $x \in G$ with constant probability. Here, $T_\mathcal{A}$ and T_ϕ are the time that is required to run $\mathcal{A}$ and make a query to ϕ, respectively.*

If $\mathcal{A}$ is an Hadamard gate, quantum amplitude amplification matches the Grover search, and thus this technique is a generalization of the Grover search. In the similar way as the Grover search, quantum amplitude amplification can be parallelized. Using $O(u2^p)$ qubits, or $O(2^p)$ small quantum processors of size u, we can find a good element in time $O(\sqrt{a/2^p}(T_\mathcal{A} + T_\phi))$.

2.2.3 Simon's Algorithm

The above quantum algorithms we introduced are exponential time algorithms. Here we introduce a quantum algorithm that can solve a problem in polynomial time. The problem is defined as follows:

Problem 2. *Let $\phi : \{0,1\}^u \to \{0,1\}^u$ be a function such that there is a unique secret value s that satisfies $\phi(x) = \phi(y)$ if and only if $x = y$ or $x = y \oplus s$. Then, find s.*

Suppose ϕ is given as a quantum oracle. Then, Simon's algorithm [Sim97] can solve the above problem with $O(n)$ queries, using $O(n)$ qubits. We have to solve a system of linear equations after making queries, which requires $O(n^3)$ arithmetic operations. Since any classical algorithm needs exponential time to solve this problem, Simon's algorithm

obtains exponential speed-up from classical algorithms. The algorithm can be applied to the problem of which condition " $\phi(x) = \phi(y)$ if and only if $x = y$ or $x = y \oplus s$" is replaced with the weaker condition "$\phi(x \oplus s) = \phi(x)$ for any x", under the assumption that ϕ satisfies some good properties [KLLN16a].

2.2.4 Quantum Claw Finding Algorithms

Let us consider two functions $f : \{0,1\}^u \to \{0,1\}^\ell$ and $g : \{0,1\}^v \to \{0,1\}^\ell$. If there is a pair $(x, y) \in \{0,1\}^u \times \{0,1\}^v$ such that $f(x) = g(y)$, then it is called a *claw* of the functions f and g. Now we consider the following problem:

Problem 3. *Let u, v be positive integers such that $u \geq v$. Suppose that two functions $f : \{0,1\}^u \to \{0,1\}^\ell$ and $g : \{0,1\}^v \to \{0,1\}^\ell$ are given as black boxes. Then, find a claw of f and g.*

This problem, called *claw finding problem*, has attracted researchers' attention. It is known that, given f and g as quantum oracles, this problem can be solved with $O(2^{(u+v)/3})$ queries in the case $v \leq u < 2v$, and $O(2^{u/2})$ queries in the case $2v \leq u$ [BHT97, Amb03, Zha05, Tan09]. Quantum claw finding algorithms and their generalizations already have some applications in attacks against symmetric-key cryptosystems [Kap14, MS17].

3 Claw Finding between Classical and Quantum Functions

Quantum claw finding algorithms are useful, though, they cannot be applied if one of target functions, say g, is not quantum accessible. For example, if we need some information from a classical online (or keyed) oracle to calculate $g(y)$, then we have to use other algorithms, even if we have a quantum computer.

Sections 3 and 4 focus on the Q1 model. Hence, this section considers how to find a claw of functions f, g where g can be evaluated only classically. We are particularly interested in the case that there exists only a single claw of f and g, and show that the following proposition holds.

Proposition 2. *Suppose that f can be implemented on a quantum circuit C_f using $O(u)$ qubits, g can be evaluated only classically, and we can use a quantum computer with $O((u+v)2^p)$ qubits. Assume that there exist only a single claw of f and g. Then we can solve Problem 3 in time*

$$O\left(T^C_{g,all} + 2^{\frac{u}{2}+v-\frac{p+p_L}{2}} \cdot T^Q_f + 2^{v-p_L+p}\right), \tag{1}$$

where $T^C_{g,all}$ is the time to calculate the pair $(y, g(y))$ for all y, T^Q_f is the time to run C_f once, and p_L is a parameter that satisfies $p_L \leq \min\{p, n\}$. We also use $O(2^v)$ classical memory.

Below we give an algorithm to find a claw and confirm that it gives the upper bound Eq.(2), which suggests Proposition 2.

Algorithm. First, evaluate $g(y)$ for all y classically, and store each pair $(y, g(y))$ in a list L. For each $y \in \{0,1\}^v$, define a function $f_y : \{0,1\}^u \to \{0,1\}$ by $f_y(x) = 1$ if and only if $f(x) = g(y)$. Given C_f and the list L, we can implement f_y on a quantum circuit that runs in time $O(T^Q_f)$ using $O(u+v)$ qubits. Note that the parallelized Grover search on f_y, which parallelly runs $O(2^{p-p_L})$ independent small processors, can find x_0 such that $f_y(x_0) = 1$ (if there exists) in time $O(2^{u/2-(p-p_L)/2} \cdot T^Q_f)$. Let C^{Grover}_y denote this quantum circuit of size $O((u+v)2^p)$. Then, run the following procedure:

1. For $0 \leq i \leq 2^{v-p_L} - 1$, do:
2. Run $C^{Grover}_{(i\|j)}$ parallelly for $0 \leq j \leq 2^{p^L} - 1$ (see Fig. 2).
3. If a pair $(x, (i\|j))$ such that $f_{(i\|j)}(x) = 1$ is found, then return the pair $(x, (i\|j))$.

In the above procedure, i, j are elements in $\{0,1\}^{v-p_L}$ and $\{0,1\}^{p^L}$, respectively, and $i\|j \in \{0,1\}^v$.

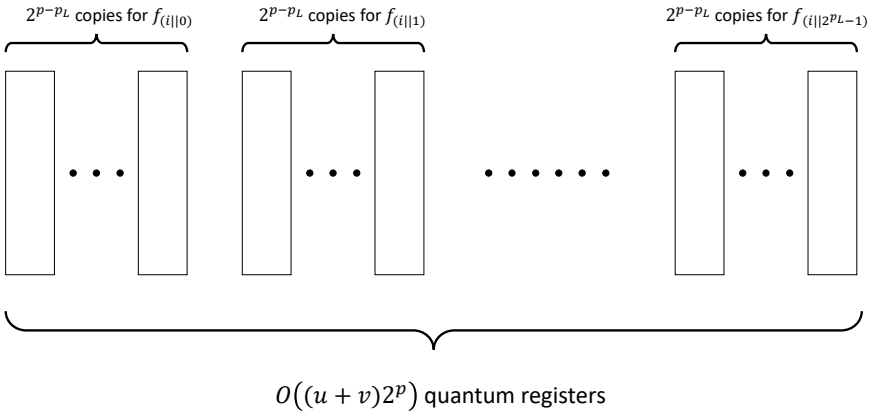

Figure 2: How to use $O(2^p)$ qubits

Complexity analysis. To evaluate $g(y)$ and store it for every y, we need $O(T^C_{g,all})$ time and $O(2^v)$ classical memory. In Step 2 of the procedure, the parallelized Grover search on $f_{(i\|j)}$ requires time $O(2^{u/2-(p-p_L)/2}T^Q_f)$ for each i and j as stated above. In Step 3 of the procedure, we need time $O(2^p)$ to check whether a pair $(x, (i\|j))$ such that $f_{(i\|j)}(x) = 1$ exists. Thus, the total running time is $O(T^C_{g,all} + 2^{v-p_L} \cdot (2^{u/2-p/2+p_L/2}T^Q_f + 2^p)) = O(T^C_{g,all} + 2^{u/2+v-p/2-p_L/2} \cdot T^Q_f + 2^{v-p_L+p})$.

As for the number of qubits, for a fixed i, we use $O((u+v)2^{p-p_L})$ qubits for the parallelized Grover search on $f_{(i\|j)}$ for each $0 \leq j \leq 2^{p_L} - 1$. Thus the total number of qubits we use is $O((u+v)2^{p-p_L}) \cdot 2^{p_L} = O((u+v)2^p)$.

3.1 Variation of Claw Finding

Next, we consider the following variant of the claw finding problem.

Problem 4. *Suppose that functions $f : \{0,1\}^u \times \{0,1\}^v \to \{0,1\}^\ell$ and $g : \{0,1\}^v \to \{0,1\}^\ell$ are given as black boxes, with promise that there is a unique pair $(x, y) \in \{0,1\}^u \times \{0,1\}^v$ such that $f(x, y) = g(y)$. Then, find such a pair (x, y).*

Again, we assume that g can be evaluated only classically, and f can be implemented on a quantum circuit. Problem 4 appears to be different from Problem 3, however, we can also solve it by applying our algorithm introduced above with a slightly modification of the definition of f_y as follows:

$$f_y(x) = 1 \text{ if and only if } f(x, y) = g(y).$$

With this small modification, we can find the pair (x, y) such that $f(x, y) = g(y)$ with the same complexity as in Proposition 2. The next section treats this variant problem to attack Feistel networks instead of the original claw finding problem with the conditions $p \leq v$ and $2^v \leq T^C_{g,all}$.

Corollary 1. *Suppose that f can be implemented on a quantum circuit C_f using $O(u+v)$ qubits, g can be evaluated only classically, and we can use a quantum computer with $O((u+v)2^p)$ qubits, where $p \leq v$. Assume that there is a unique claw of f and g. Then we can solve Problem 3 in time*

$$O\left(T^C_{g,all} + 2^{\frac{u}{2}+v-p} \cdot T^Q_f\right), \tag{2}$$

where $T^C_{g,all} \geq 2^v$ is the time to calculate the pair $(y, g(y))$ for all y and T^Q_f is the time to run C_f once. We also use $O(2^v)$ classical memory.

The above algorithms assume an ideal situation that we are given a quantum circuit that calculates f without error. However, in real applications, having some error might be inevitable. (e.g. we use Grover's algorithm as a subroutine a few times to calculate f). If the error is small, the above algorithm can still be used by replacing the Grover search on each f_y with quantum amplitude amplification explained in Sect 2.2.2. The following paragraph explains how to deal with this situation.

Suppose that our goal is to find x such that $f(x, y_0) = g(y_0)$ for a fixed $y_0 \in \{0,1\}^v$. Then assume that we only have a quantum circuit C'_f that calculates f with some error:

$$C'_f : |x\rangle |y_0\rangle |0^{w+\ell}\rangle \mapsto \sqrt{1-\delta^2_{x,y_0}} |x\rangle |y_0\rangle |\psi_{x,y_0}\rangle |f(x,y_0)\rangle + \delta_{x,y_0} |x\rangle |y_0\rangle |\mathsf{garbage}\rangle,$$

for some w-qubit state $|\psi_{x,y_0}\rangle$, which contains some necessary intermediate information to calculate $f(x, y_0)$, and $(w+\ell)$-qubit state $|\mathsf{garbage}\rangle$, which corresponds to unnecessary information. Here we assume that f is calculated by using the Grover search that normally contains some small error, and we now assume that $\delta_{x,y_0} = O(1/2^{u/2})$. Let $U_{\oplus y_0}$ denote an operation $|z\rangle \mapsto |z \oplus y_0\rangle$. With the notations in Proposition 1, we put $\mathcal{A} := C'_f(H^u \otimes U_{\oplus y_0} \otimes I_{w+\ell})$, which suggests that

$$\begin{aligned}\mathcal{A}|0^{u+v+w+\ell}\rangle &= C'_f\left(\sqrt{1/2^u}\sum_x |x\rangle |y_0\rangle |0^{w+\ell}\rangle\right)\\ &= \sqrt{1/2^u}\sum_x \left(\sqrt{1-\delta^2_{x,y_0}} |x\rangle |y_0\rangle |\psi_{x,y_0}\rangle |f(x,y_0)\rangle + \delta_{x,y_0} |x\rangle |y_0\rangle |\mathsf{garbage}\rangle\right),\end{aligned}$$

where H^u is the Hadamard operator of dimension 2^u and $I_{w+\ell}$ is the identity operator of dimension $2^{w+\ell}$. Now we run the quantum amplitude amplification with the above $\mathcal{A}$, defining that the "good" element corresponds to a state such that the last ℓ-qubits match $g(y_0)$ (i.e., define $\phi : \{0,1\}^{u+v+w+\ell} \to \{0,1\}$ as $\phi(\alpha) = 1$ if and only if the last ℓ-bits of α is $g(y_0)$, then a quantum circuit of ϕ can be implemented since we already know $(y_0, g(y_0))$.) Then, the good probability is approximately equals to $1/2^u \cdot (1-\delta^2) \approx 1/2^u$, which suggests that we can find x such that $f_{y_0}(x) = 1$ in time $O(2^{u/2})$. Thus our algorithm works even if f can be calculated with a small error.

4 Quantum DS-MitM Attacks against Feistel Ciphers

In this section, we show that quantum computers can significantly speed-up the DS-MitM attacks even under the limitation that queries are made only in a classical manner. To demonstrate it, we improve on the previous key recovery attack against 6-round Feistel ciphers presented by Guo et al. [GJNS14].

4.1 Classical DS-MITM Attack on 6-Round Feistel Ciphers

4.1.1 Overview of DS-MITM Attacks

We first briefly introduce the framework of the DS-MITM attack. The attack generally consists of the distinguisher and the key-recovery parts as illustrated in Fig. 3. A truncated differential characteristic is specified to the entire cipher and suppose that the plaintext difference ΔP propagates to the input difference to the distinguisher ΔX with probability p_1. Similarly, the ciphertext difference ΔC propagates to the output difference of the distinguisher ΔY with probability p_2 when decryption is performed. The attack is composed of two parts: *distinguisher analysis* and *queried-data analysis*.

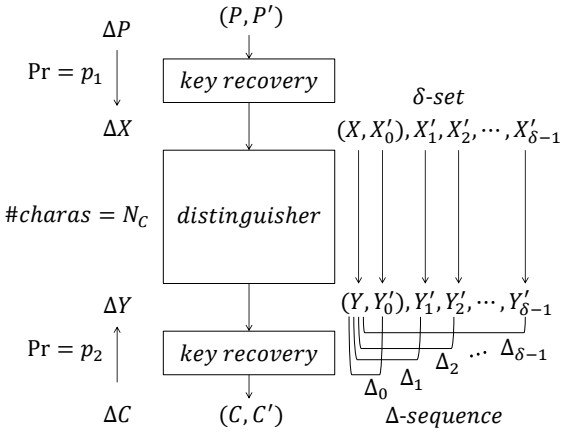

Figure 3: Overview of DS-MITM Attacks.

In the distinguisher analysis, the attacker enumerates all the possible differential characteristics that can satisfy the specified truncated differential. Suppose that there exist N_c such characteristics. For each of them, input paired values to the distinguisher are expected to be fixed uniquely. Let (X, X'_0) be the paired values. Then, the attacker generates a set of texts called δ-*set* by generating $\delta - 1$ new texts $X'_i \leftarrow X'_0 \oplus i$ for $i = 1, 2, \cdots, \delta - 1$. Suppose that the corresponding value at the output of the distinguisher can be computed. Let $Y, Y'_0, Y'_1, Y'_2, \cdots, Y'_{\delta-1}$ be the corresponding values at the output of the distinguisher. The attacker then computes the differences between Y and Y'_i for $i = 0, 1, \cdots, \delta - 1$ and makes a sequence of δ output differences at the output of the distinguisher. This sequence is called Δ-*sequence*. Note that the difference between Y and Y'_i may be able to be computed only partially, say γ bits. Thus the bit-size of the sequence is $\gamma\delta$. In the end, the Δ-sequence of the size $\gamma\delta$ bits is computed for each of the N_c characteristics and stored in a list L.

In the queried-data analysis, the attacker makes queries to collect $(p_1 p_2)^{-1}$ paired values having the plaintext difference ΔP and the ciphertext difference ΔC. One pair, with a good probability, satisfies ΔX and ΔY at the input and output of the distinguisher, respectively. Thus for each of $(p_1 p_2)^{-1}$ paired values, the attacker guesses subkeys for the key-recovery rounds such that ΔX and ΔY appear after the first and the last key recovery parts, respectively. Then, one of the paired texts (corresponding to P') is modified to P'_i so that the δ-set is generated at the input to the distinguisher, and those are queried to the oracle to obtain the corresponding ciphertext C'_i. The attacker then processes C'_i with the guessed subkeys for the last key-recovery part, and the Δ-sequence is computed at the output of the distinguisher. Finally, those are matched the list L. 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 $(p_1p_2)^{-1}N_c \times 2^{-\gamma\delta} \ll 1$.

4.1.2 Application to 6-Round Feistel Ciphers

Guo et al. [GJNS14] applied the DS-MITM attack to 6-round Feistel ciphers. The attack needs to solve the following problems.

Problem 5. *Let $F : \{0,1\}^{n/2} \mapsto \{0,1\}^{n/2}$ be a public function and Δ be a fixed difference.*

- *For a given output difference Δ_o, find all v such that $F(v) \oplus F(v \oplus \Delta) = \Delta_o$?*
- *For a given input difference Δ_i, find all v such that $F(v) \oplus F(v \oplus \Delta_i) = \Delta$?*

In the classical attack, those problems can be solved only with 1 access to the precomputed table of size $2^{n/2}$. The procedure is rather straightforward. Readers are refer to the paper by Guo et al. [GJNS14] for the exact procedure.

Distinguisher Analysis. The core of the attacks is the 5-round distinguisher explained below. The input and output differences for the 5 rounds are defined as $0\|X$ and $Y\|0$, respectively, where $X, Y \in \{0,1\}^{n/2}, X \neq Y$. For a given X, Y, the number of the 5-round differential characteristics satisfying those input and output differences is $2^{n/2}$. In fact, by representing the $n/2$-bit difference of the second round-function's output as Z, the 5-round differential characteristics can be fixed to

$$(0\|X) \xrightarrow{\text{1stR}} (X\|0) \xrightarrow{\text{2ndR}} (Z\|X) \xrightarrow{\text{3rdR}} (Y\|Z) \xrightarrow{\text{4thR}} (0\|Y) \xrightarrow{\text{5thR}} (Y\|0),$$

which is illustrated in the left-half of Fig. 4.

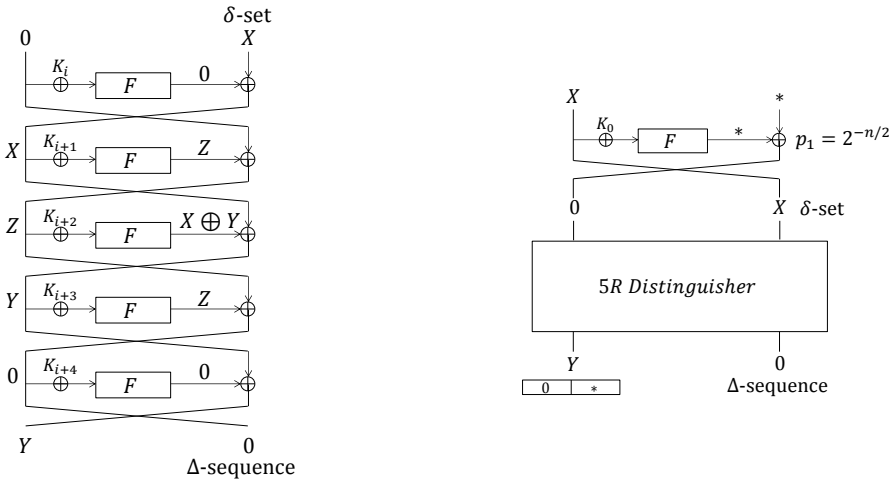

Figure 4: Left: $|Z|$ Differential Characteristics in the 5-Round Distinguisher. Right: 1-Round Extension for Key-Recovery.

For each Z, both input and output differences of F in the middle 3 rounds are fixed, which suggests that the paired values during F are fixed to one choice on average. Guo et al. showed that by generating a δ-set at the right half of the distinguisher's input, the corresponding Δ-sequence can be computed for the right-half of the distinguisher's output. Readers are referred to the paper by Guo et al. [GJNS14] for the complete analysis. The computed Δ-sequences are stored in the list L. Note that the size of δ is very small. Indeed,

$p_1 = 2^{-n/2}$, $p_2 = 1$, $N_c = 2^{n/2}$ and $\gamma = n/2$. Hence, $\delta = 3$ is sufficient to filter out all the wrong candidates.

To balance the complexities between the distinguisher analysis and the queried-data analysis, Guo et al. iterated the above analysis for $2^{n/4}$ different choices of Y. More precisely, the $n/4$ MSBs of Y are always set to 0 and $n/4$ LSBs of Y are exhaustively analyzed. The complexity of the procedure for each choice of Y is $O(2^{n/2})$ both in time and memory. Hence, the entire complexity of the distinguisher part is $O(2^{3n/4})$ in both time and memory.

Queried-Data Analysis. Guo et al. appended 1-round before the 5-round distinguisher to achieve the 6-round key-recovery attack, which is illustrated in the right-half of Fig. 4. By propagating the input difference to the distinguisher, $0\|X$, in backwards, ΔP is set to $X\|*$ where $*$ can be any $n/2$-bit difference. The probability p_1 that a randomly chosen plaintext pair with the difference $X\|*$ satisfies the difference $0\|X$ after 1 round is $2^{-n/2}$.

The attacker collects the pairs that satisfy the truncated differential in Fig. 4 by using the structure technique. Namely, the attacker prepares 2 sets of $2^{n/2}$ plaintexts in which the first and the second sets have the form $\{(c\|0), (c\|1), \cdots, (c\|2^{n/2}-1)\}$ and $\{(c \oplus X\|0), (c \oplus X\|1), \cdots, (c \oplus X\|2^{n/2}-1)\}$, respectively, where c is a randomly chosen $n/2$-bit constant. About 2^n pairs exist whereas only $O(2^{n/4})$ pairs satisfy ΔC in the corresponding ciphertexts. By iterating this procedure $O(2^{n/4})$ times for different choices of c, the attacker collects $O(2^{n/2})$ pairs satisfying the truncated differential in Fig. 4. In summary, with $O(2^{3n/4})$ queries (and thus the time complexity of $O(2^{3n/4})$ memory accesses), $O(2^{n/2})$ pairs are obtained, in which one pair will satisfy the probabilistic differential propagation in the first round.

For each pair, the input and output differences of F in the first round are fixed, which will fix K_0 uniquely. The attacker then modifies the left-half of the plaintext such that δ-set with $\delta = 3$ is generated at the right-half of the input to the distinguisher. The right-half of the plaintext is also modified to ensure that the left-half of the input to the distinguisher is not affected. The modified plaintexts are then queried to obtain the corresponding ciphertexts. The attacker computes the corresponding Δ-sequence and matches L; the list computed during the distinguisher analysis. A match recovers K_0 and Z. The other subkeys are trivially recovered from the second round one by one.

Summary of Complexity. In the distinguisher analysis, both of the time and memory complexities are $O(2^{3n/4})$. In the queried-data analysis, the data and time complexities are $O(2^{3n/4})$ and it uses a memory of size $O(2^{n/2})$ to collect the pairs with the structure technique.

Remarks. Solving Problem 5 is rather straightforward in the classical attack with $O(2^{n/2})$ memory, whereas this is a crucial problem to the quantum adversaries. This is because the efficient table look-up cannot be executed in quantum computers.

4.2 Quantum DS-MITM Attack on 6-Round Feistel Ciphers

We now convert the classical DS-MITM attack on 6-round Feistel ciphers into quantum one, in which the adversary has access to a quantum computer to perform offline computations whereas queries are made in the classical manner. The attack complexity becomes $O(2^{n/2})$ queries, $O(2^{n/2})$ offline quantum computations by using $O(2^{n/2})$ qubits.

The main idea is to introduce quantum operations to reduce the complexity of the distinguisher analysis. We show that the claw finding algorithm in Sect. 3 can be used to find a match between the distinguisher and the queried-data analyses. This enables us

to adjust the tradeoff between the complexities in the distinguisher and the queried-data analyses, and thus the data complexity can also be reduced.

4.2.1 Adjusted Truncated Differentials

After the careful analysis, we determined to analyze all $2^{n/2}$ choices of Y in the 5-round distinguisher during the distinguisher analysis part. In the classical attack, this increases the cost of the distinguisher analysis to $O(2^n)$, whereas it reduces the number of queries in the queried-data analysis. In the quantum attack, the increased cost of the distinguisher analysis can be reduced to its square root, i.e. $O(2^{n/2})$ and eventually the cost of two analyses are balanced.

4.2.2 Switching Online and Offline Phases

The claw finding algorithm in Sect. 3 matches the result of the quantum computation against the results collected in the classical method. Namely, the results of the queried-data analysis must be stored before the distinguisher analysis starts.

This can be easily done by switching the order of the two analyses. In fact, such a switch has already been applied by Darbez and Perrin [DP15, Appendix E] though their goal is to optimize the classical attack complexity, which is different from ours.

4.2.3 Queried-Data Analysis

Because queries are made in the classical manner, the procedure of the queried-data analysis remains unchanged from the classical attack by Guo et al. However, to directly apply the claw finding algorithm to the DS-MITM attack, we explicitly separate the procedure to collect $p_1^{-1} = 2^{n/2}$ pairs satisfying the truncated differentials (both ΔP and ΔC) and the procedure to compute Δ-sequences.

Precomputation for Collecting Pairs. The goal of this procedure is to collect $2^{n/2}$ pairs satisfying both $\Delta P = X\|*$ and $\Delta C = *\|0$. We again use the structure technique. Namely, we query 2 sets of $2^{n/2}$ plaintexts $\{(c\|0), (c\|1), \cdots, (c\|2^{n/2}-1)\}$ and $\{(c \oplus X\|0), (c \oplus X\|1), \cdots, (c \oplus X\|2^{n/2}-1)\}$. About 2^n pairs can be generated and $2^{n/2}$ of them have no difference in the right-half of the ciphertexts. The generated pairs are stored in the list L^{pre} indexed by the difference Y (the left-half of ΔC). In summary, this procedure requires $O(2^{n/2})$ classical queries, $O(2^{n/2})$ memory access and $O(2^{n/2})$ classical memory.

Generating Δ-sequences. The goal of this procedure is to generate Δ-sequences for all the pairs stored in L^{pre}. To make it consistent with the notations in Sect. 3, we define a classical function $g : \{0,1\}^{n/2} \to \{0,1\}^{\delta n/2}$ that takes the difference Y (the left-half of ΔC) as input and outputs the Δ-sequence as follows.

1. Pick up all the pairs in L^{pre} such that the difference Y matches the g's input.

2. Compute the Δ-sequences as in the classical attack by assuming that the probabilistic differential propagation in the first round is satisfied.

Then, the classical queried-data analysis becomes identical with computing $g(y)$ for all $y \in Y$. The cost of computing g for a single choice of y is 1. Hence, with the notation in Sect. 3, $T^C_{g,all}$ becomes $O(2^{n/2})$. After this phase, a list L with a classical memory that stores $O(2^{n/2})$ Δ-sequences is generated.

4.2.4 Quantum Distinguisher Analysis

The goal of the distinguisher analysis is to calculate Δ-sequences for all $2^{n/2}$ choices of Y and $2^{n/2}$ choices of Z in Fig. 4 in order to find a match with L. We define a quantum function $f : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}^{\delta n/2}$ that takes Z and Y as input and calculates the corresponding Δ-sequence. Given that L is computed before this analysis, the goal can be viewed as searching for a preimage Z such that $\exists Y, f(Z,Y) \in L$.

Issues to be taken into account. Note that in our situation, the function f might be incompletely defined. We want to define $f(Z,Y)$ to be the corresponding Δ-sequence to (Z,Y), however, to be precise, there are following two issues when Problem 5 is solved.

Issue 1. To calculate the corresponding Δ-sequence, we need input/output pairs of 2nd, 3rd, and 4th round functions that are compatible with the pair (Z,Y). Though there exists one suitable pair for each round function on average, there might be no pair or more than one pair that are compatible with the pair (Z,Y).

Issue 2. Suppose that issue 1 can be solved and we obtain two choices for the input and output values of the i-th round function where $i = 2,3,4$. To construct the paired values for the entire 5 rounds, we need to test 8 patterns (2 choices from each of the 2nd, 3rd, and 4th rounds).

These issues already exist even in the classical setting, but they are trivially solved. However, solving those issues in the quantum setting is non-trivial, and deserves careful attention. In what follows, for simplicity, we first describe the attack by assuming that the above issues are naturally solved as in the classical setting, and later explain how to deal with those issues.

Quantum procedures and complexity. Assume that $f(Z,Y)$ is uniquely determined for each (Z,Y). Remember that the goal of the quantum distinguisher analysis is to find Z such that $\exists Y, f(Z,Y) \in L$. As discussed in Corollary 1, suppose that a quantum circuit C_f that calculates $f(Y,Z)$ for a single choice of (Z,Y) in time T_f^Q can be implemented by using $O(n)$ qubits and we can use a quantum computer with $O(n2^p)$ qubits. Then the time complexity to find such Z becomes $O(2^{n/4+n/2-p} \cdot T_f^Q + 2^{n/2})$.

We construct C_f so that it runs the following steps:

1. Find the input/output pair of the 2nd round function F that has input difference X and output difference Z.
2. Find the input/output pair of the 3rd round function F that has input difference Z and output difference $X \oplus Y$.
3. Find the input/output pair of the 4th round function F that has input difference Y and output difference Z.
4. Construct a δ-set and calculate the corresponding Δ-sequence, using the result of Step1, 2, and 3,
5. Output the Δ-sequence obtained in Step 4.

Steps 1,2, and 3 correspond to Problem 5, which was solved using an efficient table look-up in the classical setting. However, in our circuit C_f, we use the Grover search to find the input/output pairs, since there is an obstacle that quantum computer cannot perform an efficient table look-up. Because the input and output sizes of the F are $n/2$ bits, we can run Steps 1,2, and 3 with Grover's algorithm in time $O(2^{n/4})$, using $O(n)$ qubits. The complexities for Steps 4 and 5 are much smaller than the application of Grover's algorithm.

Hence the above C_f runs in time $T_f^Q = O(2^{n/4})$, using $O(n)$ qubits. Note that C_f may return error with a small probability since we use the Grover search as subroutines for a few times. However we can deal with this error, as explained in Sect. 3.

As described in Corollary 1, if $O(n2^p)$ qubits are available ($p \leq n/2$), then we can find Z such that $f(Z, Y) \in L$ in time $O(2^{n/4+n/2-p+n/4} + 2^{n/2}) = O(2^{n-p})$. Complexities are balanced at $p = n/2$. In summary, we can find a match with time complexity $O(2^{n/2})$, using $O(n2^{n/2})$ qubits.

Dealing with the two issues. Next, we explain how to deal with issues 1 and 2. We regard that elements in $\{0,1\}^{n/2}$ are integers $0, 1, \ldots, 2^{n/2} - 1$

For a given input and output differences for the i-th round function $i = 2, 3, 4$, let $\alpha_1^{(i)}, \alpha_2^{(i)}, \alpha_3^{(i)}, \cdots$ $\beta_1^{(i)}, \beta_2^{(i)}, \beta_3^{(i)}, \cdots$ are the input and output vales satisfying the given differences, respectively, namely $F(\alpha_1^{(i)}) = \beta_1^{(i)}, F(\alpha_2^{(i)}) = \beta_2^{(i)}, F(\alpha_3^{(i)}) = \beta_3^{(i)}, \cdots$. In what follows, we assume that $\alpha_1^{(i)} < \alpha_2^{(i)}$ and $\beta_1^{(i)} < \beta_2^{(i)}$ for $i = 2, 3, 4$, without loss of generality.

As for issue 1, if there is no value satisfying (Z, Y), we simply put $f(Z, Y) :=\perp$. The problem is that there might be more than one pairs that are compatible with (Z, Y). In the classical setting, the number of solutions for a given (Z, Y) can be obtained easily bo looking-up a precomputation table, whereas in the quantum setting, we need to iterate Grover's algorithm multiple times, thus the complexity of this part increases as proportional to the number of α. We expect that the following property holds: for arbitrary $d \neq d' \in \{0,1\}^{n/2} \setminus \{0^{n/2}\}$, there is at most $N_{max} := \lceil 3(\frac{n}{2} - 1)/\log(\frac{n}{2} - 1) \rceil$ many bit-strings α that satisfies $F(\alpha) \oplus F(\alpha \oplus d) = d'$ and $\alpha < \alpha \oplus d$. This is a reasonable assumption since, for a random function $\phi : \{0,1\}^u \to \{0,1\}^u$, we have $\Pr[|\phi^{-1}(d')| \leq 3u/\log u] \geq 1 - 1/2^u$ (see Lemma 5.1 in [MU05]), and $F(\cdot) \oplus F(\cdot \oplus d)$ is an almost random function if F is random (and here we consider that the domain of $F(\cdot) \oplus F(\cdot \oplus d)$ is $\{x | x < x \oplus d\}$, of which cardinality is $2^{n/2-1}$). We consider N_{max}^3 divided cases, and associate a triplet (i, j, k) with each cases ($0 \leq i, j, k \leq N_{max} - 1$). In the case (i, j, k), we search for $\alpha_1^{(1)}, \alpha_1^{(2)}, \alpha_1^{(3)}$ from the sets of strings of which most significant $\log N_{max}$ bits are i, j, and k, respectively. Trying all N_{max}^3 cases increases the time complexity by a factor of N_{max}^3. However, we run the Grover search on the restricted domains for each case, which decreases the time complexity by a factor of $\sqrt{N_{max}}$. Consequently, dealing with issue 1 increases the time complexity by a factor of $N_{max}^{5/2} = O(n^{5/2})$.

As for issue 2, we choose one possibility among a constant number of choices before running quantum algorithm. If no solution is found under one choice, then we try another choice. Trying all possible choices increases the time complexity by a constant factor.

Eventually, we can deal with the issue by dividing the problem into $O(n^3)$ cases, which increases the time complexity by a factor of $O(n^{5/2})$.

4.2.5 Complexity Summary

- The queried-data analysis require $O(2^{n/2})$ classical queries, $O(2^{n/2})$ computations and $O(2^{n/2})$ classical memory.
- The quantum distinguisher analysis requires $O(n2^{n/2})$ qubits and $O(n^{5/2}2^{n/2})$ offline computations.

In the end, all the complexities are balanced at $\tilde{O}(2^{n/2})$, which is significantly smaller than the classical attack by Guo et al. that requires $\tilde{O}(2^{3n/4})$ queries and offline computations.

5 Attacks Using Quantum Queries

This section discusses quantum attacks in the Q2 model, where an adversary is allowed to make quantum superposition queries to online oracles. We show that we can recover full

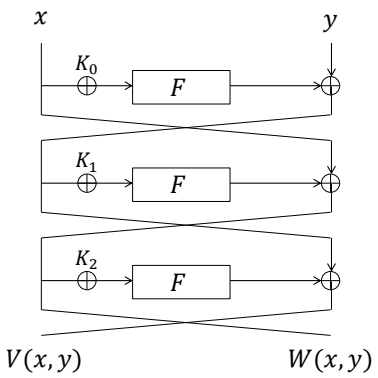

Figure 5: 3-round Feistel networks

keys of an r-round Feistel network ($r > 3$) in time $O(n^3 2^{n(r-3)/4})$, using $O(n^2)$ qubits. Our idea is, with the technique by Leander and May [LM17], to combine the trivial key-recovery attack using Grover search with the quantum distinguisher of 3-round Feistel network by Kuwakado and Morii [KM10], which was later generalized by Kaplan et al. [KLLN16a]. Again, we consider n-bit Feistel networks such that each $n/2$-bit round key is added before round function F.

Quantum distinguisher of 3-round Feistel networks. We briefly explain the quantum attack that distinguishes 3-round Feistel networks from a random permutation π [KM10, KLLN16a]. The attack works in the Q2 model, and runs in polynomial time due to Simon's algorithm.

Assume that we are given a quantum oracle that calculates $W(x, y)$, the right $n/2$-bits of the ciphertext which is encrypted with 3-round Feistel networks (see Fig. 5). Then, $W(x, y) = x \oplus F(K_1 \oplus y \oplus F(K_0 \oplus x))$ holds.

If we fix two different bit strings $\alpha, \beta \in \{0,1\}^{n/2}$ and define $f : \{0,1\}^{n/2+1} \to \{0,1\}^{n/2}$ by

$$f(b, x) = \begin{cases} W(\alpha, x) \oplus \beta \\ W(\beta, x) \oplus \alpha, \end{cases}$$

then simple calculation shows that

$$f\left((b, x) \oplus (1, F(K_0 \oplus \alpha) \oplus F(K_0 \oplus \beta))\right) = f\left(b, x\right),$$

i.e., f has a period $(1, F(K_0 \oplus \alpha) \oplus F(K_0 \oplus \beta))$.

On the other hand, if we are given a quantum oracle that calculates the right $n/2$-bits of $\pi(x, y)$ instead of $W(x, y)$ and construct such a function f, where π is a random permutation, then f does not have such a period with high probability. Thus, roughly speaking, we can distinguish 3-round Feistel networks from a random permutation π with high probability by using Simon's algorithm and by being given a quantum oracle that calculates $W(x, y)$, or the right $n/2$-bits of $\pi(x, y)$.

Simulating the half output oracle given the complete encryption oracle. As pointed out by Kaplan et al. [KLLN16a], if we are only given the complete encryption oracle (quantum oracle that returns n-bit output values $(V(x, y), W(x, y))$ or $\pi(x, y)$), then it is not trivial whether the above attack works. In the classical setting, if we are given the complete encryption oracle and only want the right half of outputs, then we can just truncate outputs of the complete oracle. However, in the quantum setting, answers to

queries are in quantum superposition states, of which right $n/2$-bits and left $n/2$-bits are entangled. Since the truncation destroys entanglements, it is not trivial to simulate the oracle that only produces the right half of the output from the complete encryption oracle. However, the simulation is still possible and we explain its procedure below.

Let $\mathcal{O} : |x\rangle|y\rangle|z\rangle|w\rangle \mapsto |x\rangle|y\rangle|z \oplus O_L(x,y)\rangle|w \oplus O_R(x,y)\rangle$ be the complete encryption oracle, where O_L, O_R denote the left and right $n/2$-bits of the complete encryption, respectively. Suppose that we want to simulate oracle $\mathcal{O}_R : |x\rangle|y\rangle|w\rangle \mapsto |x\rangle|y\rangle|w \oplus O_R(x,y)\rangle$. It suffices to make an operator that maps $|\psi\rangle|0^{n/2}\rangle$ to $(\mathcal{O}_R|\psi\rangle) \otimes |0^{n/2}\rangle$, using ancilla qubits. Let $|+\rangle := H^{n/2}|0^{n/2}\rangle$, where $H^{n/2}$ is an $n/2$-bit Hadamard gate. Then $\mathcal{O}|x\rangle|y\rangle|+\rangle|w\rangle = |x\rangle|y\rangle|+\rangle|w \oplus O_R(x,y)\rangle$ holds for any $x, y, w \in \{0,1\}^{n/2}$. Therefore,

$$(I \otimes H^{n/2}) \cdot \mathsf{Swap} \cdot \mathcal{O} \cdot \mathsf{Swap} \cdot (I \otimes H^{n/2})|\psi\rangle \otimes |0^{n/2}\rangle = (\mathcal{O}_R|\psi\rangle) \otimes |0^{n/2}\rangle$$

holds, where Swap is an operator that swaps last n-qubits: $|x\rangle|y\rangle|z\rangle|w\rangle \mapsto |x\rangle|y\rangle|w\rangle|z\rangle$. Hence we can simulate $\mathcal{O}_R$ given the complete encryption oracle $\mathcal{O}$, using ancilla qubits.

Combining the quantum distinguisher with key recovery attacks. Leander and May proposed a technique to combine Grover's algorithm and Simon's algorithm [LM17]. Particularly, they proposed a quantum algorithm and showed the following proposition:

Proposition 3 (Theorem 2 in [LM17])**.** *Let $\Psi : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function such that $\Psi(k,\cdot) : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a random function for any fixed $k \in \mathbb{F}_2^n$. Let $\Phi : \mathbb{F}_2^m \times \mathbb{F}_2^{3n} \to \mathbb{F}_2^n$ be a function defined by $\Phi(k_0, k_1, k_2, x) = \Psi(k_0, x \oplus k_1) \oplus k_2$. Then, given quantum oracle accesses to $\Phi_{k_0,k_1,k_2}(\cdot)$ and $\Psi(\cdot,\cdot)$, we can recover (k_0, k_1, k_2) with a constant probability and $O((m+n)2^{m/2})$ queries, using $O(m+n^2)$ qubits.*

Leander and May's algorithm firstly defines function $\Phi' : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ by $\Phi'(k,x) = \Phi_{k_0,k_1,k_2}(x) \oplus \Psi(k,x)$. Since $\Phi'(k,x) = \Psi(k_0, x \oplus k_1) \oplus k_2 \oplus \Psi(k,x)$ holds, $\Phi'(k_0,\cdot)$ has a period k_1. Roughly speaking, their algorithm searches for correct key k_0 with the Grover search and checks whether or not $\Phi'(k,\cdot)$ is periodic for the candidate key k by running $O(n)$ many independent Simon's algorithm parallelly. The Grover search for m-bit key k_0 requires $O(m)$ qubits. $O(n)$ parallel Simon's algorithm requires $O(n^2)$ qubits. Thus the above algorithm needs $O(m+n^2)$ qubits.

Note that the above proposition refers only to *query complexity*, but not to *time complexity*. In the above algorithm, it is $O(n^3)$ arithmetic operations after $O(n)$ queries made by Simon's algorithm (see Section 2.2.3) that makes difference between query complexity and time complexity. Considering this $O(n^3)$ operations, the running time of their algorithm is $O((m+n^3)2^{m/2})$.

Now we guess subkeys for the last $(r-3)$-rounds $K_3, \ldots, K_{r-1}$, given the quantum encryption oracle of an r-round Feistel network. If the guess is correct, we can implement a quantum circuit that calculates the first three rounds of the Feistel network. Hence we can detect the correctness of the guess by using the 3-round quantum distinguisher described above. We guess $K_3, \ldots, K_{r-1}$ by using Grover's algorithm, while we use Simon's algorithm for the 3-round distinguisher. Therefore we need the technique by Leander and May that combines Grover's algorithm and Simon's algorithm.

However, Proposition 3 dose not suit our case, and we need the following variant proposition, which can be proven in the similar way as the original proposition.

Proposition 4. *Let $\Psi : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function such that $\Psi(k,\cdot) : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a random function for any fixed $k \in \mathbb{F}_2^n$. Let $\Phi : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function such that $\Phi(k,\cdot) : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a random function for any fixed $k \in \mathbb{F}_2^n \setminus \{k_0\}$, and $\Phi(k_0, x) = \Psi(k_0, x \oplus k_1)$. Then, given quantum oracle accesses to $\Phi(\cdot,\cdot)$ and $\Psi(\cdot,\cdot)$, we can recover (k_0, k_1) with a constant probability and $O((m+n)2^{m/2})$ queries, using $O(m+n^2)$ qubits.*

Again, we search for the correct key k_0 with the Grover search and check whether or not $\Phi'(k,\cdot) = \Phi_{k_0,k_1,k_2}(\cdot) \oplus \Psi(k,\cdot)$ is periodic for the candidate key k by running Simon's algorithm parallelly. Due to the similar reason as for Proposition 3, *time complexity* of this variant algorithm also becomes at most $O((m+n^3)2^{m/2})$.

Now we explain how to combine the quantum distinguisher with key recovery attacks. Assume that we are given the quantum encryption oracle of an r-round Feistel network $\mathsf{Enc}^r : \{0,1\}^n \to \{0,1\}^n$. For $k = (K'_3, \ldots, K'_{r-1}) \in \{0,1\}^{(r-3)n/2}$, let $D_k : \{0,1\}^n \to \{0,1\}^n$ denote the partial decryption of the last $(r-3)$-rounds with the key candidate $(K'_3, \ldots, K'_{r-1})$. Define $W : \{0,1\}^{(r-3)n/2} \times \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ be the function that outputs the right half of $n/2$-bits of $D_k \circ \mathsf{Enc}^r(x,y)$. We can implement a quantum circuit of W using the quantum oracle of Enc^r and the simulating technique we described above. Note that $W(k_0,x,y) = x \oplus F(K_1 \oplus y \oplus F(K_0 \oplus x))$ holds, where $k_0 = (K_3, \ldots, K_{r-1})$ is the set of partial keys of the Feistel network Enc^r from the 4-th round to the last round.

Now, fix two different $n/2$-bit strings α, β, and define $\Psi, \Phi : \{0,1\}^{(r-3)n/2} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ by

$$\Psi(k,x) = W(k,\alpha,x) \oplus \beta, \Phi(k,x) = W(k,\beta,x) \oplus \alpha.$$

Then $\Psi(k,\cdot)$ is an almost random function for each k, and $\Phi(k,\cdot)$ is also an almost random function for each $k \neq k_0$. In addition, $\Phi(k_0,x) = \Psi(K_0, x \oplus k_1)$ holds, where $k_1 = F(\alpha \oplus K_0) \oplus F(\beta \oplus K_0)$, since

$$\begin{aligned}
\Phi(k_0,x) &= W(k_0,\beta,x) \oplus \alpha \\
&= \beta \oplus F\left(K_1 \oplus x \oplus F(\beta \oplus K_0)\right) \alpha \\
&= \alpha \oplus F\left(K_1 \oplus x \oplus F(\beta \oplus K_0) \oplus F(\alpha \oplus K_0) \oplus F(\alpha \oplus K_0)\right) \oplus \beta \\
&= W(k_0,\alpha,x \oplus k_1) \oplus \beta \\
&= \Psi(k_0, x \oplus k_1).
\end{aligned}$$

Thus, by applying Proposition 4, we can recover the round keys $K_3, \ldots, K_{r-1}$. After we obtain $K_3, \ldots, K_{r-1}$, we can construct a quantum circuit that calculates the first 3 rounds of the Feistel network. Thus, for arbitrary $\alpha, \beta \in \{0,1\}^{n/2}$ such that $\alpha \neq \beta$, we can compute $F(\alpha \oplus K_0) \oplus F(\beta \oplus K_0)$ in polynomial time by using the 3-round distinguisher. Hence we can recover K_0 in time $O(2^{n/4})$ by using the Grover search. Once $K_0, K_3, \ldots, K_{r-1}$ are recovered, we can easily recover K_1, K_2 in time $O(2^{n/4})$ by using the Grover search.

Complexity summary. Consequently, we can recover $K_0, \ldots K_{r-1}$ in time $O(n^3 2^{(r-3)n/4})$, using $O(n^2)$ qubits. In addition, the Grover search to find correct k_0 can be parallelized. If we can afford $O(n^2 2^p)$ qubits, $K_0, \ldots . K_{r-1}$ can be recovered in time $O(n^3 2^{(r-3)n/4-p/2})$.

In particular, for the case $r = 6$, all the complexities are balanced at $\tilde{O}(2^{n/2})$, which is the same as the attack in Section 4:

- The attack requires $O(n2^{n/2})$ quantum queries, $O(n^3 2^{n/2})$ computations, and $O(n^2 2^{n/2})$ qubits.

The time complexity and the number of qubits are almost the same as those in Section 4. However, the trade-off between time complexity and the number of qubits are different. For example, the attack described in this section runs in time $\tilde{O}(2^{3n/4})$ using polynomially many qubits, while the attack in Section 4 requires time $\tilde{O}(2^n)$ using polynomially many qubits (see Fig. 1).

References

[Amb03] Andris Ambainis. Quantum walk algorithm for element distinctness. *CoRR*, quant-ph/0311001, 2003. See FOCS 2004 SIAM J. Comput. 37(1): 210–239 (2007).

[BB17] Gustavo Banegas and Daniel J. Bernstein. Low-communication parallel quantum multi-target preimage search. Cryptology ePrint Archive, Report 2017/789, 2017. To appear at SAC2017.

[BBG+13] Robert Beals, Stephen Brierley, Oliver Gray, Aram W. Harrow, Samuel Kutin, Noah Linden, Dan Shepherd, and Mark Stather. Efficient distributed quantum computing. In *Proceedings of the Royal Society A*, volume 469, page 20120686. The Royal Society, 2013.

[BBHT98] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortsch. Phys.*, 46(4-5):493–505, June 1998. https://arxiv.org/abs/quant-ph/9605034.

[Ber09] Daniel J. Bernstein. Cost analysis of hash collisions: will quantum computers make SHARCS obsolete? In *SHARCS 2009*, 2009.

[BHMT02] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.

[BHT97] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum algorithm for the collision problem. *CoRR*, quant-ph/9705002, 1997. Quantum Cryptanalysis of Hash and Claw-Free Functions. LATIN 1998: 163-169.

[Bon17] Xavier Bonnetain. Quantum key-recovery on full AEZ. Cryptology ePrint Archive, Report 2017/767, 2017. To appear at SAC 2017.

[DDKS15] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. New attacks on feistel structures with improved memory complexities. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 433–454. Springer, 2015.

[DFJ13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved key recovery attacks on reduced-round AES in the single-key setting. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 371–387. Springer, 2013.

[DP15] Patrick Derbez and Léo Perrin. Meet-in-the-middle attacks and structural analysis of round-reduced PRINCE. In Gregor Leander, editor, *FSE 2015*, volume 9054 of *LNCS*, pages 190–216. Springer, 2015. Cryptology ePrint Archive, Report 2015/239.

[DS08] Hüseyin Demirci and Ali Aydin Selçuk. A meet-in-the-middle attack on 8-round AES. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 116–126. Springer, 2008.

[GJNS14] Jian Guo, Jérémy Jean, Ivica Nikolic, and Yu Sasaki. Meet-in-the-middle attacks on generic feistel constructions. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 458–477. Springer, 2014.

[GJNS16] Jian Guo, Jérémy Jean, Ivica Nikolic, and Yu Sasaki. Meet-in-the-middle attacks on classes of contracting and expanding feistel constructions. *IACR Trans. Symmetric Cryptol.*, 2016(2):307–337, 2016.

[GR03] Lov Grover and Terry Rudolph. How significant are the known collision and element distinctness quantum algorithms? *CoRR*, quant-ph/0309123, 2003. See GR04.

[GR04] Lov Grover and Terry Rudolph. How significant are the known collision and element distinctness quantum algorithms. *Quantum Info. Comput.*, 4(3):201–206, May 2004.

[Gro96] Lov. K Grover. A fast quantum mechanical algorithm for database search. In *STOC 1996*, pages 212–219, 1996. https://arxiv.org/abs/quant-ph/9605043.

[HS17] Akinori Hosoyamada and Yu Sasaki. Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. Cryptology ePrint Archive, Report 2017/977, 2017. http://eprint.iacr.org/2017/977.

[IS12] Takanori Isobe and Kyoji Shibutani. All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography, 2012*, volume 7707 of *LNCS*, pages 202–221. Springer, 2012.

[IS13] Takanori Isobe and Kyoji Shibutani. Generic key recovery attack on Feistel scheme. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 464–485. Springer, 2013.

[Kap14] Marc Kaplan. Quantum attacks against iterated block ciphers. *arXiv preprint arXiv:1410.1434*, 2014.

[KLLN16a] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 207–237. Springer, 2016.

[KLLN16b] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Quantum differential and linear cryptanalysis. *IACR Trans. Symmetric Cryptol.*, 2016(1):71–94, 2016.

[KM10] Hidenori Kuwakado and Masakatu Morii. Quantum distinguisher between the 3-round Feistel cipher and the random permutation. In *ISIT 2010*, pages 2682–2685. IEEE, 2010.

[KM12] Hidenori Kuwakado and Masakatu Morii. Security on the quantum-type Even-Mansour cipher. In *ISITA 2012*, pages 312–316. IEEE, 2012.

[Knu02] Lars R. Knudsen. The Security of Feistel Ciphers with Six Rounds or Less. *J. Cryptology*, 15(3):207–222, 2002.

[LM17] Gregor Leander and Alexander May. Grover meets Simon - quantumly attacking the FX-construction. Cryptology ePrint Archive, Report 2017/427, 2017. To appear at Asiacrypt 2017.

[MBTM17] Kerry A. McKay, Larry Bassham, Meltem SÃűnmez Turan, and Nicky Mouha. NISTIR 8114 Report on Lightweight Cryptography. Technical report, U.S. Department of Commerce, National Institute of Standards and Technology, 2017. https://doi.org/10.6028/NIST.IR.8114.

[MS17] Bart Mennink and Alan Szepieniec. XOR of PRPs in a quantum world. In Tanja Lange and Tsuyoshi Takagi, editors, *PQCrypto 2017*, volume 10346 of *LNCS*, pages 367–383. Springer, 2017.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[Sim97] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[Tan09] Seiichiro Tani. Claw finding algorithms using quantum walk. *Theoretical Computer Science*, 410(50):5285 – 5297, 2009. Mathematical Foundations of Computer Science (MFCS 2007).

[Zha05] Shengyu Zhang. Promised and distributed quantum search. In Lusheng Wang, editor, *Computing and Combinatorics: 11th Annual International Conference, COCOON 2005 Kunming, China, August 16–19, 2005 Proceedings*, pages 430–439, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.