



New results on quantum boomerang attacks

Hongkai Zou^{1,2} · Jian Zou^{1,2}  · Yiyuan Luo³

Received: 1 October 2022 / Accepted: 24 March 2023 / Published online: 8 April 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

At SAC 2021, Frixons et al. proposed quantum boomerang attacks that can effectively recover the keys of block ciphers in the quantum setting. Based on their work, we further consider how to quantize the generic boomerang attacks proposed by Biham et al. at FSE 2002, so as to obtain more generic quantum boomerang attacks. Similar to Frixons et al.'s work, we only consider quantum key recovery attacks in the single-key setting. With the help of some famous quantum algorithms, this paper presents two methods to convert the attacks of Biham et al. into some new quantum key recovery attacks. In order to proof our methods, we apply our new ideas to attack Serpent-256 and ARIA-196. To sum up, for Serpent-256, we give valid 9-round and 10-round quantum key recovery attacks respectively. The quantum time complexity of 9-round and 10-round of Serpent-256 is $2^{115.43}$ and $2^{126.6}$ respectively. Furthermore, we show a valid quantum key attack on 6-round ARIA-196 which has a time complexity of $2^{89.8}$ with negligible memory. The time complexity of the above quantum attacks are better than the corresponding classical attacks and quantum generic key recovery attack via Grover's algorithm.

Keywords Block cipher · Boomerang attack · Key recovery attack · Quantum search · Quantum algorithm

1 Introduction

With the development of quantum technology, research in the field of quantum cryptography has received extensive attention. The research in the field of quantum cryptography mainly includes quantum communication, quantum computation, etc. Among them, quantum communication mainly studies a technology of communication

✉ Jian Zou
fzuzoujian15@163.com

¹ College of Computer and Data Science, Fuzhou University, Fuzhou, China

² Key Lab of Information Security of Network Systems, Fuzhou University, Fuzhou, China

³ School of Information Sciences and Technology, Huizhou University, Huizhou, China

based on the information transmission function of quantum medium, while quantum computation mainly studies quantum computers and quantum algorithms suitable for quantum computers. In the field of quantum communication, quantum key distribution is the most studied direction. However, there are still many problems and challenges. Especially in the implementation process, there are still many works to be done in transmission distance, anti-noise, anti-equipment defects, and other aspects. For the above problems, researchers have done a lot of work in recent years, such as [1–3]. In addition, there are also many results in other research branches of quantum communication. For example, Yin et al. [4] proposed an efficient quantum digital signature protocol and built an integrated quantum security network; Zhou et al. [5] proved experimentally for the first time that quantum technology can provide quantum advantages for machine learning, and designed a quantum version of the blind box game. In this paper, we focus on the quantum security analysis of ciphers.

As we know, the security analysis of symmetric primitives has always been a hot topic in cryptography research. In the classical setting, there are many efficient cryptanalysis techniques, among which differential cryptanalysis [6] is one of the powerful tools to evaluate the security of primitives. Differential cryptanalysis mainly studies the propagation of differences through an encryption process. Adversaries usually use the differential path with high probability to distinguish a cipher from a random permutation.

In 1999, Wanger et al. [7] proposed a variant of differential analysis, which was called the boomerang attack. The boomerang attack uses two high probability short differentials instead of one long differential with low probability. Obviously, it is easier for an adversary to find short differentials with a high probability than a long one with a high enough probability. Later, the boomerang attack was further developed into the amplified boomerang attack [8] and the rectangle attack [9]. As one of the most famous families of attacks, the boomerang attack has been applied to many kinds of ciphers in the single-key and related-key settings [10–12].

In recent years, the research in the field of quantum cryptography has attracted many people's interest. With the help of Grover [13], Simon [14] and other efficient quantum algorithms, many well-known symmetric-key schemes can be broken in the quantum setting [15–17]. On the other hand, some studies have shown that some efficient classical cryptanalysis techniques can also be used in the quantum setting. At Eurocrypt 2020, Hosoyamada and Sasaki [18] considered the application of rebound attack [19, 20] in the quantum setting. By combining Grover's algorithm with quantum collision finding algorithms, they revealed that quantum rebound attacks could use some differentials whose probabilities are too low to be useful in the classical setting. In 2021, Hosoyamada and Sasaki [21] showed how to convert the classical semi-free-start collision attacks on reduced SHA-2 into quantum collision attacks. In addition, there are some earlier efforts about quantizing classical attacks [22–24].

It is well known that the generic quantum key recovery attack against block ciphers is to use Grover's algorithm to search for the key exhaustively. However, much less is known about how to further accelerate the key recovery process of block ciphers in the quantum setting. At SAC 2021, Frixons et al. [25] proposed a valid quantum boomerang key recovery attack, and applied it to SAFER++. Their idea can be applied

to the 5-round key recovery attack against SAFER++, which is faster than the quantum generic key search.

Based on Frixons et al.'s work [25], this paper proposes some new quantum boomerang attack in the single-key setting. In [25], Frixons et al. considered how to construct a quantum key recovery attack by appending some rounds before or after the boomerang distinguisher. However, they didn't consider the case of appending some rounds before and after the distinguisher at the same time. In addition, we find that their technique is not generic, which can't convert some specific classical boomerang attacks into the corresponding quantum boomerang attacks.

Our Contributions. In order to solve the above problems, we present some new techniques to convert Biham et al.'s generic boomerang attacks [10] into valid quantum attacks. Generally, the boomerang attack needs to deal with a large number of possible candidate quartets, which may cause a huge time and memory complexity. At FSE 2002, Biham et al. introduced generic boomerang attacks which could reduce the number of tested possible quartets. Biham et al.'s idea might help us to reduce the complexity of attacks in the classical or quantum settings.

Inspired by Frixons et al.'s idea in [25], we propose two methods to quantize the generic boomerang attacks of [10]. The first method is based on quantum search and quantum collision-finding algorithm, which can verify the correctness of the subkey guess by exhausting the remaining key bits. Another method utilizes the fact that adversaries can obtain two or more right boomerang pairs in a boomerang attack. Since the good key guess will appear more often among all key candidates, we can find these good key by solving the element distinctness problem. Therefore, our second idea combines Ambainis's quantum element distinctness algorithm [26] and quantum search to construct quantum key recovery attacks, which may be faster than the quantum generic key search attack. Different from the first method, the second method can determine some partial good subkeys by solving the element distinctness problem in the first place. Then we can perform a quantum search on the remaining key bits so as to recover the complete key.

Compared with Frixons et al.'s work, our new techniques are more generic, which can convert more classical boomerang attacks into the corresponding quantum boomerang attacks. In addition, our new quantum attacks can also retrieve subkeys on both sides of the boomerang distinguisher, which further improve Frixons et al.'s work. Based on our new methods, we can construct some new quantum key recovery attacks against 9-round and 10-round Serpent-256 and 6-round ARIA-196 respectively. The above attack results are better than the previous quantum generic key recovery attacks and classical attacks. In detail, for 9-round Serpent-256, we construct a quantum key recovery attack with a time complexity of $2^{115.43}$, which is less than the quantum generic key recovery attack by a factor of $2^{12.57}$. When compared with the best classical boomerang attack, the time complexity is reduced by a factor of $2^{8.17}$. For 10-round Serpent-256, we construct a quantum key recovery attack with a time complexity of $2^{126.6}$, which is less than the best classical boomerang attack by a factor of $2^{51.7}$. Compared with the quantum generic key recovery attack, our time complexity is reduced by a factor of $2^{1.4}$. For ARIA-196, we construct a 6-round quantum key recovery attack with a time complexity of $2^{89.8}$, which is less than the quantum generic key recovery attack by a factor of $2^{8.2}$. Compared with the best classical boomerang attack, the time

Table 1 Classical and quantum boomerang attacks on Serpent-256 and ARIA-196 in the single-key setting

Target	Rounds	Setting	Time	Memory	Source
Serpent-256	9	Classic	$2^{123.6}$	$2^{21.5}$	[10]
		Quantum	2^{128}	Negligible	Grover search
		Quantum	$2^{115.43}$	$2^{55.68}$ (QRAQM)	Sect. 5
	10	Classic	$2^{178.3}$	2^{96}	[10]
		Quantum	2^{128}	Negligible	Grover search
		Quantum	$2^{126.6}$	Negligible	Sect. 5
ARIA-196	6	Classic	2^{108}	2^{56}	[27]
		Quantum	2^{98}	Negligible	Grover search
		Quantum	$2^{89.8}$	Negligible	Sect. 6

complexity is reduced by a factor of $2^{18.2}$ without incurring any memory cost. We summarize our new results in Table 1.

The paper is organized as follows. In Sect. 2, we briefly describe the boomerang attack and some quantum tools used in this paper. In Sect. 3, we recall some previous related works. In Sect. 4, we introduce two methods to show how to convert generic boomerang attacks into valid quantum attacks. We apply these techniques to Serpent-256 in Sect. 5. In Sect. 6, we give a valid quantum key recovery attack on 6-round ARIA. We conclude this paper in Sect. 7.

2 Preliminaries

In this section, we give a brief introduction to the boomerang attack and some quantum tools used in this paper.

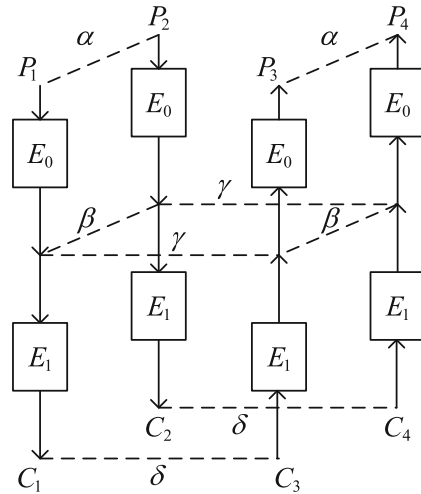
2.1 The boomerang attack

The boomerang attack [7] is a differential cryptanalysis technique proposed by Wagner in 1999. The main idea behind the boomerang attack is to use two short differentials with high probability (as shown in Fig. 1).

Let E be a block cipher that can be described as $E = E_1 \circ E_0$. The block size of E is n bits and the key size is k bits. Assume that E_0 has a differential $\alpha \rightarrow \beta$ with probability p and E_1 has a differential $\gamma \rightarrow \delta$ with probability q . Given oracles of \mathcal{O} and \mathcal{O}^{-1} , where \mathcal{O} is either E or a random permutation, the boomerang attack can be described as follows:

1. Choose a plaintext P_1 at random. Then, compute $P_2 = P_1 \oplus \alpha$ and ask for the ciphertexts $C_1 = \mathcal{O}(P_1)$ and $C_2 = \mathcal{O}(P_2)$.
2. Compute $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$, and ask for the plaintexts $P_3 = \mathcal{O}^{-1}(C_3)$ and $P_4 = \mathcal{O}^{-1}(C_4)$.

Fig. 1 The boomerang attack



3. Check whether $P_3 \oplus P_4 = \alpha$.

If \mathcal{O} is a random permutation, the probability of $P_3 \oplus P_4 = \alpha$ is 2^{-n} . When \mathcal{O} is the target block cipher E , the probability of $P_3 \oplus P_4 = \alpha$ is p^2q^2 . Note that the attack can be constructed for all possible β 's and γ 's simultaneously. In other words, the probability can be improved to $(\hat{p}\hat{q})^2$, where

$$\hat{p} = \sqrt{\sum_{\alpha \xrightarrow{\beta} \beta} \text{Pr}^2[\alpha \rightarrow \beta]}, \quad \hat{q} = \sqrt{\sum_{\gamma \xrightarrow{\delta} \delta} \text{Pr}^2[\gamma \rightarrow \delta]}$$

For detailed descriptions and analysis of the boomerang attack and its variants, please refer to [7–9].

2.2 Quantum tools

We assume that the quantum algorithms studied in this paper are written in the quantum circuit model. For detailed definitions of qubits, quantum gates, etc., readers can refer to [28].

2.2.1 Quantum complexities

In the standard quantum circuit model, the time complexity of a quantum algorithm is the number of gates or the depth of the circuit. The memory complexity is the number of qubits used in the circuit.

In this paper, we try to construct the key recovery attacks of some block ciphers. For the convenience of analysis, we assume the unit of time to be the time required to run for one encryption in our attack on a block cipher. By using Grover search, given a block cipher with a k -bit key, the time complexity of the generic quantum key

recovery attack is $O(2^{k/2})$. If the time complexity of a quantum key recovery attack is lower than $O(2^{k/2})$, we regard it as a valid quantum attack.

2.2.2 Quantum random access memory

In classical cryptanalysis, many algorithms need to access memory cells dynamically when they are running. This operation is assumed to be performed in time $O(1)$ in the standard random access memory (RAM) model. In the quantum setting, the quantum random access memory (qRAM) model is used in many quantum attacks, which can be regarded as the quantum counterpart of classical random access memory. We use a generic random access gate that only costs $O(1)$ to abstract out memory access:

$$\underbrace{|i\rangle}_{\text{index}} \underbrace{|m_0, \dots, m_{s-1}\rangle}_{\text{data}} \underbrace{|b\rangle}_{\text{result}} \rightarrow |i\rangle |m_0, \dots, m_{s-1}\rangle |b \oplus m_i\rangle$$

According to the terms in [29]: (a) If i and m_j are classical, this is the classical RAM; (b) If i is in superposition and m_j is classical, this is the quantum random access classical memory (QRACM); (c) If i and m_j are in superposition, this is the quantum random access quantum memory (QRAQM).

So far, QRACM and QRAQM are still theoretical quantum memory models and the availability of them is controversial. Although it is not known how to build large qRAMs at present, many research works are still based on the assumption that large qRAMs are available in order to deal with various possibilities in the future. Of course, when we construct quantum algorithms, it is quite meaningful for us to try to avoid the use of qRAM.

2.2.3 Quantum search

It is well known that Grover’s algorithm [13] can speed up exhaustive search procedures by a quadratic factor. Further, Brassard et al. [30] proposed the Amplitude Amplification technique to generalize Grover’s algorithm. In this paper, we use quantum search to refer to the Amplitude Amplification technique. The Amplitude Amplification technique can be explained as follows.

Theorem 1 [[30], Theorem 2 and 4] *Let \mathcal{A} be any quantum algorithm that performs no measurements, and let $\mathcal{B} : X \rightarrow \{0, 1\}$ be a boolean function that classifies the output of \mathcal{A} as "good" or "bad". Let p_a be the success probability of \mathcal{A} . Let $\theta = \arcsin \sqrt{p_a}$. Assume that $T_{\mathcal{A}}$ and $T_{\mathcal{B}}$ are the time complexity of \mathcal{A} and \mathcal{B} respectively, then there exists a quantum algorithm $\text{Amplify}(\mathcal{A}, \mathcal{B})$ that can run in time $\lfloor \pi/4\theta \rfloor (2T_{\mathcal{A}} + T_{\mathcal{B}})$. By measuring the output of $\text{Amplify}(\mathcal{A}, \mathcal{B})$, we obtain a "good" output of \mathcal{A} with probability $\max(1 - p_a, p_a)$. If p_a is known exactly, then we can obtain a good result with probability 1.*

In order to explain our new algorithms, we need to show the relationship between the classical procedure and the quantum procedure. It is pointed out in [31] that there is a

recursive correspondence relationship between a classical search procedure composed of multiple **Repeat** loops and a quantum procedure with nested quantum searches. Therefore, for many quantum attacks, we can describe them classically in the first place.

Generally, the quantum procedure $\text{Amplify}(\mathcal{A}, \mathcal{B})$ corresponds to a classical exhaustive search process with approximately $(\lfloor \pi/4\theta \rfloor)^2$ iterations. The classical exhaustive search process contains a **Repeat** loop that calls \mathcal{A} and a **test** block (i.e., an **If** procedure block) that tests the output of \mathcal{A} . Similar to the work of Frixons et al. [25], in the first place we adopt multiple **Repeat** loops and **If** procedure blocks to describe our algorithms in a classical way. In the second place, by using Theorem 1, we turn our classical algorithm into a quantum procedure $\text{Amplify}(\mathcal{A}, \mathcal{B})$ whose output is either a superposition of possible solutions or none. According to Theorem 1, if a solution exists and the number of iterations of each **Repeat** loop is exact, we can obtain the solution by measuring the output of $\text{Amplify}(\mathcal{A}, \mathcal{B})$ with probability 1.

2.2.4 Quantum collision-finding and element l -distinctness algorithms

This paper involves the following two problems:

Problem 1 (collision-finding) Given a function $F : \{0, 1\}^m \rightarrow \{0, 1\}^n$, where $m \geq n/2$. The goal is to find a pair (x, y) such that $F(x) = F(y)$.

In the quantum setting, we can use Grover’s algorithm to find a collision in time $O(2^{n/2})$, with negligible memory. A more efficient quantum collision-finding algorithm (or called BHT algorithm) is proposed by Brassard, Høyer, and Tapp [32]. In detail, they utilized Grover’s algorithm as a subroutine to construct a quantum collision-finding algorithm for a 2-to-1 function, which only required $O(2^{n/3})$ quantum queries and $O(2^{n/3})$ quantum memory in the QRACM model. In the case that the domain and codomain of F are the same size, i.e. $m = n$, Yuen [33] proved that the BHT algorithm could produce a collision for random functions after $O(2^{n/3})$ quantum queries. Therefore, when $m \geq n$, we can use the BHT algorithm to solve Problem 1. The idea of the BHT algorithm can be described as follows. Firstly, we construct a list L with $2^{n/3}$ elements $(x, F(x))$ by querying F about $O(2^{n/3})$ times. The elements in L are stored in quantum memory, and we can access these elements with the quantum superpositions in the QRACM model. After the above operations, there are $2^{n/3}$ solutions in the search space of size 2^n . Secondly, we use Grover search to find a collision in time $O(\sqrt{\frac{2^n}{2^{n/3}}}) = O(\sqrt{2^{2n/3}}) = O(2^{n/3})$.

Problem 2 (element l -distinctness) Given N elements $x_1, \dots, x_N \in [M]$, check if there are l distinct indices $i_1, \dots, i_l \in [N]$ such that $x_{i_1} = \dots = x_{i_l}$.

We call such l indices i_1, \dots, i_l an l -collision. Under the QRAQM model, Ambainis [26] gave a quantum algorithm to solve Problem 2. Ambainis’s algorithm is based on the quantum walk algorithm, and it requires $O(N^{1/l+1})$ queries and $O(N^{1/l+1})$ quantum memory.

Ambainis’s algorithm can also be used to solve Problem 1, which can work in the worst case $m = n/2$ (where only a single collision exists on expectation). According to the precise analysis in [34], after $(\pi/2)^2 \cdot 2^{n/3}$ queries, we can find a collision with overwhelming probability by using Ambainis’s algorithm. Note that Ambainis’s

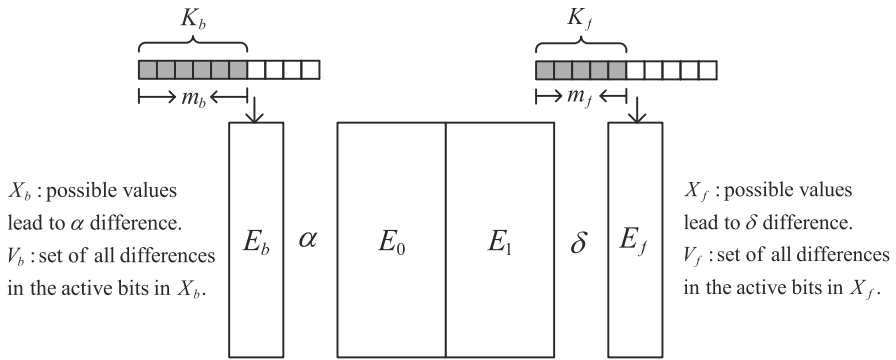


Fig. 2 Outline of boomerang key recovery attack on E

algorithm requires a quantum memory of $O(2^{n/3})$ in the QRAQM model, while the BHT algorithm needs a quantum memory of $O(2^{n/3})$ in the QRACM model.

3 Related works

In this section, we will briefly review some works related to this paper, including two generic boomerang attacks proposed by Biham et al. [10] and the quantum boomerang attack proposed by Frixons et al. [25].

3.1 Biham et al.’s generic boomerang attacks

At FSE 2002, Biham et al. [10] proposed some generic boomerang attacks on the block cipher E , where the target block cipher is described as $E = E_f \circ E_1 \circ E_0 \circ E_b$. Here, $E_1 \circ E_0$ is the boomerang distinguisher, while E_b and E_f are the additional rounds appended before and after the distinguisher respectively. The block cipher size of E is assumed to be n bits, while the key size is k bits. The outline of boomerang key recovery attack on the target block cipher E is shown in Fig. 2.

The notations in Fig. 2 can be explained as follows. Let X_b be the set of all plaintext differences that may cause a difference α after E_b , while V_b be the space spanned by the values in X_b . Assume that $t_b = \log_2|X_b|$ and $r_b = \log_2|V_b|$. Note that $t_b \leq r_b$ and $n - r_b$ bits are inactive for all the values in V_b . Let K_b be the partial key involved in E_b , which can determine the difference of the plaintexts by decrypting pairs with the difference α after E_b . We denote the size of K_b as m_b . Similarly, let X_f be the set of all ciphertexts differences which can lead to a difference δ before E_f , while V_f be the space spanned by the values in X_f . Assume that $t_f = \log_2|X_f|$ and $r_f = \log_2|V_f|$, where $t_f \leq r_f$. Let K_f be the partial key involved in E_f , which can determine the ciphertext difference when encrypting a pair with difference δ . The size of K_f is denoted as m_f .

3.1.1 Generic boomerang attack on $E = E_1 \circ E_0 \circ E_b$

In the first place, we give a brief introduction of Biham et al.’s generic boomerang key recovery attack on $E = E_1 \circ E_0 \circ E_b$ in Algorithm 1. For simplification, we assume that the attack needs y structures.

Algorithm 1 Generic Boomerang Attack on $E = E_1 \circ E_0 \circ E_b$ (adapted from [10], Sect. 4)

Input: oracle access to $E = E_1 \circ E_0 \circ E_b$
Output: the key K_b

- 1: Prepare y structures of 2^{r_b} plaintexts P_i . Each structure S takes all the possible values of the r_b active bits with the other $n - r_b$ bits as some constant
- 2: Initialize two empty lists: $T_1 \leftarrow \phi, T_2 \leftarrow \phi$
- 3: For each structure S , perform steps 4-16:
- 4: For all $P_i \in S$, compute $P'_i = E^{-1}(E(P_i) \oplus \delta)$, sort S by values of P'_i
- 5: **for** each (P'_i, P'_j) such that $P'_i \oplus P'_j \in V_b$ **do**
- 6: **if** $P'_i \oplus P'_j \in X_b$ and $P_i \oplus P_j \in X_b$ **then**
- 7: $T_1 \leftarrow T_1 \cup \{(P_i, P_j, P'_i, P'_j)\}$
- 8: **end if**
- 9: **end for**
- 10: **for** each pair $(P_i, P_j, P'_i, P'_j) \in T_1$ **do**
- 11: **for** each possible value of K_b **do**
- 12: **if** $E_{b,K_b}(P_i) \oplus E_{b,K_b}(P_j) = \alpha = E_{b,K_b}(P'_i) \oplus E_{b,K_b}(P'_j)$ **then**
- 13: $T_2 \leftarrow T_2 \cup \{(P_i, P_j, P'_i, P'_j, K_b)\}$
- 14: **end if**
- 15: **end for**
- 16: **end for**
- 17: **return** a guess of K_b which is suggested 4 times in T_2

A brief analysis of Algorithm 1. We restrict our attention to the number of candidate quartets and keys generated in Algorithm 1. For each structure, we do the following analysis. In step 5, we need to find some pairs (P'_i, P'_j) satisfying $P'_i \oplus P'_j \in V_b$. Each structure has about $\frac{(2^{r_b})^2}{2} = 2^{2r_b-1}$ plaintext pairs, which can produce about $\frac{2^{2r_b-1}}{2^{n-r_b}} = 2^{3r_b-n-1}$ collisions of $(n - r_b)$ bits. For two plaintext pairs in a candidate quartet (P_i, P_j, P'_i, P'_j) , the probability that both conditions $P'_i \oplus P'_j \in X_b$ and $P_i \oplus P_j \in X_b$ are satisfied is $(2^{t_b-r_b})^2$. Therefore, for each structure, steps 5-9 generate $2^{3r_b-n-1} \cdot 2^{2t_b-2r_b} = 2^{2t_b+r_b-n-1}$ candidate quartets (P_i, P_j, P'_i, P'_j) . Given a right quartet, both pairs must agree on K_b . Since $|X_b| = 2^{t_b}$ and $|K_b| = 2^{m_b}$, we expect that $2^{m_b-t_b}$ subkeys would transform one of the input differences in X_b to the difference α . It means that each pair suggests $2^{m_b-t_b}$ subkeys and a candidate quartet can suggest $\frac{(2^{m_b-t_b})^2}{2 \cdot 2^{m_b}} = 2^{m_b-2t_b-1}$ subkeys on average. Therefore, we will obtain $2^{2t_b+r_b-n-1} \cdot 2^{m_b-2t_b-1} = 2^{r_b+m_b-n-2}$ candidate subkeys for each structure.

Since Algorithm 1 requires y structures, the data complexity is $y \cdot 2^{r_b+1}$. When it is a valid classical boomerang attack, we have $y \cdot 2^{r_b+1} \leq 2^n$. In addition, the total number of subkey candidates of K_b is expected to be $y \cdot 2^{r_b+m_b-n-2}$, which are distributed over

2^{m_b} subkeys. Thus, the expected number of hits for each subkey is $y \cdot 2^{r_b-n-2} \leq 2^{-3}$. It means that the expected number of hits for a wrong subkey is less than 1/8 while the right subkey is expected to appear 4 times. After obtaining the right subkey K_b , we search for the remaining $(k - m_b)$ -bit subkeys exhaustively to recover the complete key K .

In this paper, the time and memory complexity analysis of the generic boomerang attack is not essential, so we omit it here. Interested readers can refer to [10] for more details.

Remark 1 Note that if we consider the generic boomerang attack on $E = E_f \circ E_1 \circ E_0$, it is only necessary to replace all the sub-scripts b by f in the above equations and consider E^{-1} .

3.1.2 Enhancing the boomerang attack

Biham et al. also proposed a method to use the boomerang distinguisher with both E_b and E_f . Their generic boomerang key recovery attack on $E = E_f \circ E_1 \circ E_0 \circ E_b$ is described as Algorithm 2. Similarly, we assume that the enhancing attack needs y structures.

Algorithm 2 Generic Boomerang Attack on $E = E_f \circ E_1 \circ E_0 \circ E_b$ (adapted from [10], Sect. 5)

Input: oracle access to $E = E_f \circ E_1 \circ E_0 \circ E_b$

Output: the keys K_b and K_f

- 1: Prepare y structures of 2^{r_b} plaintexts P_i . Each structure S takes all the possible values of the r_b active bits with the other $n - r_b$ bits as some constant
 - 2: Initialize two empty lists: $T_1 \leftarrow \phi, T_2 \leftarrow \phi$
 - 3: For each structure S , perform steps 4-17:
 - 4: For all $P_i \in S$ and $\varepsilon \in X_f$, compute $P'_i = E^{-1}(E(P_i) \oplus \varepsilon)$. Define a set G as $G = \{P'_i = E^{-1}(E(P_i) \oplus \varepsilon) | P_i \in S, \varepsilon \in X_f\}$, sort G by values of P'_i
 - 5: **for** each (P'_i, P'_j) such that $P'_i \oplus P'_j \in V_b$ **do**
 - 6: **if** $P'_i \oplus P'_j \in X_b$ and $P_i \oplus P_j \in X_b$ **then**
 - 7: $T_1 \leftarrow T_1 \cup \{(P_i, P_j, P'_i, P'_j)\}$
 - 8: **end if**
 - 9: **end for**
 - 10: **for** each pair $(P_i, P_j, P'_i, P'_j) \in T_1$ **do**
 - 11: Encrypt (P_i, P_j, P'_i, P'_j) to obtain (C_i, C_j, C'_i, C'_j)
 - 12: **for** each possible value of (K_b, K_f) **do**
 - 13: **if** $E_{b,K_b}(P_i) \oplus E_{b,K_b}(P_j) = \alpha = E_{b,K_b}(P'_i) \oplus E_{b,K_b}(P'_j)$ and $E_{f,K_f}^{-1}(C_i) \oplus E_{f,K_f}^{-1}(C'_i) = \delta = E_{f,K_f}^{-1}(C_j) \oplus E_{f,K_f}^{-1}(C'_j)$ **then**
 - 14: $T_2 \leftarrow T_2 \cup \{(P_i, P_j, P'_i, P'_j, K_b, K_f)\}$
 - 15: **end if**
 - 16: **end for**
 - 17: **end for**
 - 18: **return** a guess of (K_b, K_f) which is suggested 4 times in T_2
-

A brief analysis of Algorithm 2. For the convenience of constructing our new algorithms in this paper, we still restrict our attention to the number of candidate quartets and keys generated in Algorithm 2. For each structure S , we perform the following analysis. In step 4 of Algorithm 2, we obtain a set G of size $2^{r_b+t_f}$, where $G = \{P'_i = E^{-1}(E(P_i) \oplus \varepsilon) | P_i \in S, \varepsilon \in X_f\}$. Then, we need to find pairs (P'_i, P'_j) such that $P'_i \oplus P'_j \in V_b$. Each structure G has $\frac{(2^{r_b+t_f})^2}{2} = 2^{2r_b+2t_f-1}$ plaintext pairs, which can produce about $\frac{2^{2r_b+2t_f-1}}{2^{n-r_b}} = 2^{3r_b+2t_f-n-1}$ collisions of $(n - r_b)$ bits. For two pairs in a candidate quartet (P_i, P_j, P'_i, P'_j) , the probability that their differences both belong to X_b is $(2^{t_b-r_b})^2$. Therefore, for each structure, steps 5-9 generate $2^{3r_b+2t_f-n-1} \cdot 2^{2t_b-2r_b} = 2^{2t_b+2t_f+r_b-n-1}$ candidate quartets. For each candidate plaintext quartet, we can not only obtain the corresponding ciphertext quartets (C_i, C_j, C'_i, C'_j) , but also find the corresponding candidate subkeys both in E_b and E_f . As pointed out in Sect.3.1.1, given a right plaintext quartet (P_i, P_j, P'_i, P'_j) , both pairs (P_i, P_j) and (P'_i, P'_j) must agree on K_b . Since $|X_b| = 2^{t_b}$ and $|K_b| = 2^{m_b}$, $2^{m_b-t_b}$ subkeys on average will transform one of the input differences in X_b to difference α . It means that each pair suggests $2^{m_b-t_b}$ subkeys and a plaintext candidate quartet can suggest $\frac{(2^{m_b-t_b})^2}{2 \cdot 2^{m_b}} = 2^{m_b-2t_b-1}$ subkeys on average. Repeating the analysis for E_f , we expect to have $2^{m_f-2t_f-1}$ subkeys suggestions from each ciphertext quartet (C_i, C_j, C'_i, C'_j) . Thus, there are about $2^{m_b+m_f-2t_b-2t_f-2}$ possible candidate subkeys for each quartet in T_1 . After step 17, we can obtain $2^{2t_b+2t_f+r_b-n-1} \cdot 2^{m_b+m_f-2t_b-2t_f-2} = 2^{r_b+m_b+m_f-n-3}$ candidate subkeys for each structure. After analyzing y structures, we can obtain a list T_2 of size $y \cdot 2^{r_b+m_b+m_f-n-3}$, which stores the tuples $(P_i, P_j, P'_i, P'_j, K_b, K_f)$. Thus, we can expect that the good subkey guess (K_b, K_f) is in these tuples. In addition, the probability that each candidate quartet suggests a subkey guess is $2^{m_b+m_f-2t_b-2t_f-2}$.

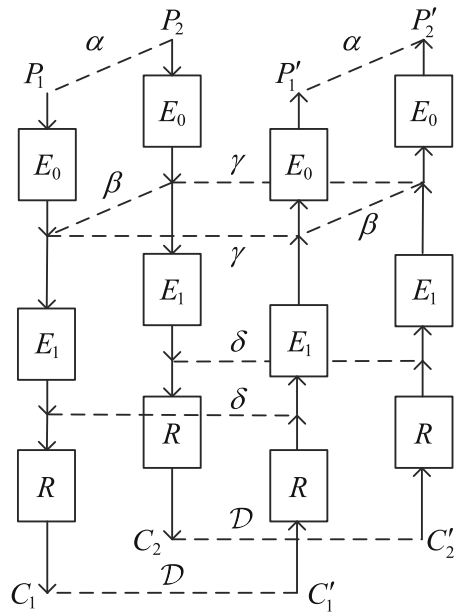
Since the total number of subkey candidates of (K_b, K_f) is about $y \cdot 2^{r_b+m_b+m_f-n-3}$, we can get $\frac{y \cdot 2^{r_b+m_b+m_f-n-3}}{2^{m_b+m_f}} = y \cdot 2^{r_b-n-3}$ subkey hits in total. Recall that the size of each structure G is at most 2^{n+t_f} and the data complexity of Algorithm 2 is $y \cdot 2^{r_b+t_f}$. When it is a valid classical boomerang attack, we have $y \cdot 2^{r_b+t_f} \leq 2^{n+t_f}$. Thus, the expected number of hits for each subkey is $y \cdot 2^{r_b-n-3} \leq 2^{-3}$. It means that the expected number of hits for a wrong subkey is less than 1/8 while the right subkey is expected to appear 4 times (or more). After obtaining the right guesses of K_b and K_f , we search for the remaining $(k - m_b - m_f)$ -bit subkeys exhaustively to recover the complete key K .

We also omit the analysis of the time and memory complexity of Algorithm 2 here. For interested reader, please refer to [10] for more details.

3.2 The quantum boomerang attack proposed by Frixons et al.

At SAC 2021, Frixons et al. [25] proposed the quantum boomerang attack to recover the keys of some block ciphers. In order to construct an effective quantum boomerang attack, they first analyzed a classical key recovery attack, which was based on the boomerang distinguisher.

Fig. 3 The boomerang key recovery attack on $E = R \circ E_1 \circ E_0$



Their classical boomerang key recovery attack can be explained as follows. Adversaries append some additional rounds R to the target cipher. According to the position of R , their key recovery attacks can be divided into two cases: (1) R is appended after the boomerang distinguisher. (2) R is appended before the boomerang distinguisher.

In the following, we consider the first case (see Fig. 3). Assume the target block cipher E can be described as $E = R \circ E_1 \circ E_0$, where $E_1 \circ E_0$ is the boomerang distinguisher satisfying the properties presented in Sect. 2.1. Let \mathcal{D} denote the set of differences that can be obtained from δ after the R -round. For the convenience of analysis, we assume that the size of \mathcal{D} is 2^d . Let p_{out} be the probability that we get δ back from an element in \mathcal{D} by computing the R -round backwards. Fixions et al. also assumed that $1/(\sqrt{p_{out}pq}) < 2^d$, in which case the key recovery attack only needs a single structure. Denote the partial key needed to be guessed in the R -round as K_{out} , and its size is k_{out} . Their key recovery attack is described in Algorithm 3. If the R -round is appended before the distinguisher, we shall consider E^{-1} . Their attack on this case is similar to Algorithm 3.

A brief analysis of Algorithm 3. In steps 4-6, finding such pair (C'_i, C'_j) is an $(n - d)$ -bit collision search problem. Since a structure S is sorted in step 3, these pairs satisfying $C'_i \oplus C'_j \in \mathcal{D}$ can be computed efficiently. According to Algorithm 3, the structure S contains approximately $\frac{1}{p_{out}(pq)^2}$ ciphertext pairs. The probability of each pair satisfying the constraint $C'_i \oplus C'_j \in \mathcal{D}$ is $\frac{|\mathcal{D}|}{2^n} = 2^{d-n}$. Therefore, we can easily obtain a list T_1 of size $\frac{2^{d-n}}{p_{out}(pq)^2}$. In steps 7-13, we need to exhaust all possible key guesses for each of the candidate quartets (C_i, C_j, C'_i, C'_j) in T_1 . By definition, K_{out} has $2^{k_{out}}$ possible values, while p_{out} is the probability of obtaining δ back from an

Algorithm 3 The Classical Key Recovery Attack Considered by Frixons et al.

```

Input: oracle access to  $E = R \circ E_1 \circ E_0$ 
Output: guesses of  $K_{out}$ 
1: Prepare a structure  $S \subseteq \{X \oplus \Delta, \Delta \in \mathcal{D}\}$ , which contains  $1/(\sqrt{p_{out}} pq)$  ciphertexts
2: Initialize two empty lists:  $T_1 \leftarrow \phi, T_2 \leftarrow \phi$ 
3: For all  $C \in S$ , compute  $C' = E(E^{-1}(C) \oplus \alpha)$ , sort  $S$  by values of  $C'$ 
4: for each  $(C'_i, C'_j)$  such that  $C'_i \oplus C'_j \in \mathcal{D}$  do
5:    $T_1 \leftarrow T_1 \cup \{(C_i, C_j, C'_i, C'_j)\}$ 
6: end for
7: for each  $(C_i, C_j, C'_i, C'_j) \in T_1$  do
8:   for each possible value of  $K_{out}$  do
9:     if  $R_{K_{out}}^{-1}(C_i) \oplus R_{K_{out}}^{-1}(C_j) = \delta = R_{K_{out}}^{-1}(C'_i) \oplus R_{K_{out}}^{-1}(C'_j)$  then
10:       $T_2 \leftarrow T_2 \cup \{(C_i, C_j, C'_i, C'_j, K_{out})\}$ 
11:     end if
12:   end for
13: end for
14: return all guesses of  $K_{out}$  in  $T_2$ 

```

element in \mathcal{D} by computing the R rounds backwards. Thus, each of the quartets in T_1 can suggest $2^{k_{out}} \cdot p_{out}^2$ candidate keys K_{out} on average. As a result, the size of T_2 is $\frac{2^{d-n}}{p_{out}(pq)^2} \cdot 2^{k_{out}} \cdot p_{out}^2 = \frac{2^{d+k_{out}-n} \cdot p_{out}}{(pq)^2}$. After the above process, the good key guess of K_{out} is expected to be in T_2 . After recovering K_{out} , we can perform an exhaustive search on the remaining $(k - k_{out})$ bits to obtain the full key K . For a detailed analysis of Algorithm 3, readers can refer to [25].

Based on the above classical key recovery attack, Frixons et al. proposed a quantum boomerang key recovery attack in Algorithm 4.

Algorithm 4 The Quantum Boomerang Attack Shown in [25]

```

Input: superposition oracle access to  $E$  and  $E^{-1}$ 
Output: the full key  $K$ 
1: Repeat  $\frac{2^{d+k_{out}-n} \cdot p_{out}}{(pq)^2}$  times
2:   Perform the following steps to sample a guess of  $K_{out}$ :
3:   Repeat  $\max(1, \frac{1}{2^{k_{out}} \cdot p_{out}^2})$  times
4:     Find a valid pair  $(C_1, C_2, C'_1, C'_2)$ 
     ▷ Use Grover's or quantum collision-finding algorithms
5:     Repeat  $2^{k_{out}}$  times
     ▷ Exhaust  $k_{out}$ -bit key  $K_{out}$ 
6:     If  $R_{K_{out}}^{-1}(C_1) \oplus R_{K_{out}}^{-1}(C_2) = \delta = R_{K_{out}}^{-1}(C'_1) \oplus R_{K_{out}}^{-1}(C'_2)$  then  $K_{out}$  is "good"
7:     EndRepeat
8:     If a valid  $K_{out}$  is obtained then  $(C_1, C_2, C'_1, C'_2)$  is "good"
9:   EndRepeat
10:   For the key  $K_{out}$  obtained above, we do
11:   Repeat  $2^{(k-k_{out})}$  times
12:     Guess the remaining  $(k - k_{out})$  bits of key, check if the full  $K$  matches
13:   EndRepeat
14:   If a valid  $K$  is obtained then  $K_{out}$  is "good"
15: EndRepeat
16: Measure  $K_{out}$ , recompute  $K$  and return it

```

Recall that the classical key recovery attack (Algorithm 3) can be divided into two phases. Firstly, we can test K_{out} by using the property of the boomerang pairs. Secondly, we need to compute the remaining $(k - k_{out})$ bits with brute-force search algorithm. According to the analysis of Algorithm 3, we can check the number of iterations in the loops of Algorithm 4 as follows. As shown in Algorithm 3, we finally obtained a list T_2 of $\frac{2^{d+k_{out}-n} \cdot p_{out}}{(pq)^2}$ tuples $(C_i, C_j, C'_i, C'_j, K_{out})$, in which one of the key guesses is correct. In other words, the probability that the correct subkey K_{out} obtained through the procedure in the loop is $\frac{(pq)^2}{2^{d+k_{out}-n} \cdot p_{out}}$. Therefore, the number of iterations of step 1 is $\frac{2^{d+k_{out}-n} \cdot p_{out}}{(pq)^2}$. The probability that a candidate quartet suggests a key candidate is $2^{k_{out}} \cdot p_{out}^2$, so there are $\max(1, \frac{1}{2^{k_{out}} \cdot p_{out}^2})$ iterations in step 3. Given a valid pair in step 5, we need to repeat step 6 $2^{k_{out}}$ times so as to verify the correctness of a guess of K_{out} . Finally, the remaining $(k - k_{out})$ bits are searched exhaustively in step 12.

Remark 2 Algorithm 4 is described in a classical way. Therefore, when analyzing the quantum complexity of the attack, we need to change the **Repeat** loops into nested quantum searches. We omit the analysis of quantum attack complexity here. For more analysis of Algorithm 4, please refer to [25].

4 Our New Quantum Boomerang Attacks

Inspired by the work of Frixons et al. [25], this section proposes two methods to convert Biham et al.'s generic boomerang attacks into quantum boomerang attacks.

4.1 First method

Similar to Algorithm 4, our first method is a quantized version of Algorithm 1, which combines quantum search or quantum collision-finding algorithm with the boomerang distinguisher (see Algorithm 5).

According to the analysis of Algorithm 1, we shall not only check the number of iterations in the loops of Algorithm 5 but also need to explain some simplified steps in Algorithm 5.

Step 1: the number of iterations depends on the probability of obtaining the good subkey guess K_b through the procedure. Classically, we obtain a list of $y \cdot 2^{m_b+r_b-n-2}$ tuples $(P_1, P_2, P'_1, P'_2, K_b)$ in Algorithm 1. Since the good guess of K_b appears 4 times in these tuples, we shall perform steps 2-18 $y \cdot 2^{m_b+r_b-n-2}$ times to obtain all possible solutions. According to Theorem 1, we shall adopt the quantum procedure $\text{Amplify}(\mathcal{A}, \mathcal{B})$ to implement from the step 1 to step 19 in Algorithm 5, which requires about $\sqrt{y \cdot 2^{m_b+r_b-n-2}}$ iterations. After the above iteration operations, we can obtain a superposition of possible solutions.

Algorithm 5 Our Quantum Boomerang Attack on $E = E_1 \circ E_0 \circ E_b$

Input: superposition oracle access to E and E^{-1}
Output: the full key K

- 1: **Repeat** $y \cdot 2^{m_b+r_b-n-2}$ **times**
- 2: Perform the following steps to sample a guess of K_b :
- 3: **Repeat** $2^{2t_b-m_b+1}$ **times**
- 4: **Repeat** $2^{2r_b-2t_b}$ **times**
- 5: Find a valid quartet (P_1, P_2, P'_1, P'_2)
 - ▷ Use Grover's or quantum collision-finding algorithms
- 6: **If** $P'_1 \oplus P'_2 \in X_b$ and $P_1 \oplus P_2 \in X_b$ **then** (P_1, P_2, P'_1, P'_2) is "good"
- 7: **EndRepeat**
- 8: For the valid boomerang quartet (P_1, P_2, P'_1, P'_2) obtained above:
- 9: **Repeat** 2^{m_b} **times**
 - ▷ Exhaust m_b -bit key K_b to obtain the corresponding key guess
- 10: **If** $E_{b,K_b}(P_1) \oplus E_{b,K_b}(P_2) = \alpha = E_{b,K_b}(P'_1) \oplus E_{b,K_b}(P'_2)$ **then** K_b is "good"
- 11: **EndRepeat**
- 12: **If** a valid K_b is obtained **then** (P_1, P_2, P'_1, P'_2) is "good"
- 13: **EndRepeat**
- 14: For the key guess K_b obtained above, we check its correctness:
- 15: **Repeat** $2^{(k-m_b)}$ **times**
- 16: Guess the remaining $(k - m_b)$ bits of key, check if the full K matches
- 17: **EndRepeat**
- 18: **If** a valid K is obtained **then** K_b is "good"
- 19: **EndRepeat**
- 20: Measure K_b , recompute K and return it

Step 3: we need to find a key candidate for each possible quartet (P_1, P_2, P'_1, P'_2) . According to Algorithm 1, the probability that a candidate quartet yields a key candidate is $2^{m_b-2t_b-1}$. As a result, there are $\max(1, 2^{2t_b-m_b+1})$ iterations in step 3. Generally, we have $2^{2t_b-m_b+1} \geq 1$.

Steps 4, 5 and 6: in order to find a candidate value of the subkey K_b , we first need to find a valid pair (P_1, P_2) in step 5. The pair satisfies the conditions $P_1 \oplus P_2 \in V_b$ and $P'_1 \oplus P'_2 \in V_b$, where $P'_1 = E^{-1}(E(P_1) \oplus \delta)$ and $P'_2 = E^{-1}(E(P_2) \oplus \delta)$. Obviously, finding such a valid pair (P_1, P_2) is a collision search problem. In the quantum setting, we can use the Grover's algorithm or quantum collision-finding algorithm to solve the problem. In order to reduce the number of candidate quartets, we need to check whether the differences of two pairs both belong to X_b in step 6. This is the main difference between our algorithm and Frixons et al.'s algorithm. According to the analysis in Algorithm 1, the probability that both conditions $P'_1 \oplus P'_2 \in X_b$ and $P_1 \oplus P_2 \in X_b$ are satisfied is $(2^{t_b-r_b})^2$. Therefore, we need to perform steps 5-6 $2^{2r_b-2t_b}$ times to find a valid quartet (P_1, P_2, P'_1, P'_2) . That is, the number of iterations of step 4 is $2^{2r_b-2t_b}$.

Step 9: we need to find the key candidate (in the quantum setting, actually the superposition of all possible candidates) for a given quartet. A simple method is to enumerate all possible values of K_b .

Step 15: we search for the remaining $(k - m_b)$ bits exhaustively. After running exact Amplitude Amplification with about $2^{(k-m_b)/2}$ iterations, we can find the solution or prove that there is no solution with probability 1.

Note that the number of iterations in the loops of Algorithm 5 is written in a classical way. By changing the **Repeat** loops into nested quantum searches, we obtain

an approximate quantum time complexity of Algorithm 5 as follows:

$$\sqrt{y \cdot 2^{m_b+r_b-n-2}} \cdot \left(\sqrt{2^{2t_b-m_b+1}} \cdot \left(\sqrt{2^{2r_b-2t_b}} \cdot \underbrace{2^{(n-r_b)/3}}_{\text{Step 5}} + 2^{m_b/2} \right) + 2^{(k-m_b)/2} \right). \tag{1}$$

The above complexity corresponds to the case that we adopt the quantum collision-finding algorithm in step 5. (If there are many solutions in step 5, we use the BHT algorithm; Otherwise, we use Ambainis’s algorithm.) Therefore, it also requires about $2^{(n-r_b)/3}$ quantum memory to find a valid quartet in the qRAM model. According to Theorem 1, assuming that the above numbers of iterations are exact, we can obtain a more exact complexity of Algorithm 5 by considering the additional factors of quantum search (see Eq. 2).

$$2 \left\lfloor \frac{\pi}{4} \sqrt{y \cdot 2^{m_b+r_b-n-2}} \right\rfloor \left(2 \left\lfloor \frac{\pi}{4} \sqrt{2^{2t_b-m_b+1}} \right\rfloor \cdot \left(2 \left\lfloor \frac{\pi}{4} \cdot \sqrt{2^{2r_b-2t_b}} \right\rfloor \cdot 4 \cdot 2^{(n-r_b)/3} + 2 \left\lfloor \frac{\pi}{4} 2^{m_b/2} \right\rfloor \right) + 4 \left\lfloor \frac{\pi}{4} 2^{(k-m_b)/2} \right\rfloor \right) \tag{2}$$

If we use Grover search instead of quantum collision-finding algorithm in step 5, the factor $2^{(n-r_b)/3}$ in Eqs. 1 and 2 will be replaced by $2^{(n-r_b)/2}$. Meanwhile, the attack only uses a negligible quantum memory. Therefore, when large qRAMs are not available, we shall adopt Grover’s algorithm instead of quantum collision-finding algorithm to find a valid quartet in step 5.

Similarly, we can rewrite Algorithm 2 into the quantized version Algorithm 6, which is composed of multiple **Repeat** loops and **If** blocks in the same way. Since the analysis of Algorithm 6 is similar to Algorithm 5, we omit it here. By changing the **Repeat** loops of Algorithm 6 into nested quantum searches, we obtain an approximate quantum time complexity of Algorithm 6 as follows:

$$\sqrt{y \cdot 2^{m_b+r_b+m_f-n-3}} \cdot \left(\sqrt{2^{2t_b+2t_f-m_b-m_f+2}} \cdot \left(\sqrt{2^{2r_b-2t_b}} \cdot 2^{(n-r_b)/2} + 2^{(m_b+m_f)/2} \right) + 2^{(k-m_b-m_f)/2} \right). \tag{3}$$

According to Theorem 1, assuming that the numbers of iterations in Algorithm 6 are exact, we can obtain a more exact quantum time complexity of Algorithm 6 (see Eq. 4).

$$2 \left\lfloor \frac{\pi}{4} \sqrt{y \cdot 2^{m_b+r_b+m_f-n-3}} \right\rfloor \left(2 \left\lfloor \frac{\pi}{4} \sqrt{2^{2t_b+2t_f-m_b-m_f+2}} \right\rfloor \cdot \left(2 \left\lfloor \frac{\pi}{4} \sqrt{2^{2r_b-2t_b}} \right\rfloor \cdot 4 \cdot 2^{(n-r_b)/2} + 2 \left\lfloor \frac{\pi}{4} 2^{(m_b+m_f)/2} \right\rfloor \right) + 4 \left\lfloor \frac{\pi}{4} 2^{(k-m_b-m_f)/2} \right\rfloor \right) \tag{4}$$

Algorithm 6 Our Quantum Boomerang Attack on $E = E_f \circ E_1 \circ E_0 \circ E_b$

Input: superposition oracle access to E and E^{-1}
Output: the full key K

- 1: **Repeat** $y \cdot 2^{m_b+r_b+m_f-n-3}$ **times**
- 2: Use the following steps to sample a guess of (K_b, K_f) :
- 3: **Repeat** $2^{2t_b+2t_f-m_b-m_f+2}$ **times**
- 4: **Repeat** $2^{2r_b-2t_b}$ **times**
- 5: Find a valid quartet (P_1, P_2, P'_1, P'_2) ▷ Use Grover's algorithm generally
- 6: **If** $P'_1 \oplus P'_2 \in X_b$ and $P_1 \oplus P_2 \in X_b$ **then** (P_1, P_2, P'_1, P'_2) is "good"
- 7: **EndRepeat**
- 8: For the valid boomerang quartet (P_1, P_2, P'_1, P'_2) obtained above:
- 9: Ask for the corresponding ciphertexts (C_1, C_2, C'_1, C'_2)
- 10: **Repeat** $2^{m_b+m_f}$ **times** ▷ Exhaust key (K_b, K_f) to obtain the corresponding key guess
- 11: **If** $E_{b,K_b}(P_1) \oplus E_{b,K_b}(P_2) = \alpha = E_{b,K_b}(P'_1) \oplus E_{b,K_b}(P'_2)$ and $E_{f,K_f}^{-1}(C_1) \oplus E_{f,K_f}^{-1}(C'_1) = \delta = E_{f,K_f}^{-1}(C_2) \oplus E_{f,K_f}^{-1}(C'_2)$ **then** (K_b, K_f) is "good"
- 12: **EndRepeat**
- 13: **If** a valid (K_b, K_f) is obtained **then** (P_1, P_2, P'_1, P'_2) is "good"
- 14: **EndRepeat**
- 15: For the key guess (K_b, K_f) obtained above, we check its correctness:
- 16: **Repeat** $2^{(k-m_b-m_f)}$ **times**
- 17: Guess the remaining $(k - m_b - m_f)$ bits of key, check if the full K matches
- 18: **EndRepeat**
- 19: **If** a valid K is obtained **then** (K_b, K_f) is "good"
- 20: **EndRepeat**
- 21: Measure (K_b, K_f) , recompute K and return it

Remark 3 For the boomerang attack on $E = E_f \circ E_1 \circ E_0 \circ E_b$, there are usually many remaining key bits in the classical analysis. This means that the factor $2^{(k-m_b-m_f)}$ will dominate the calculation of time complexity. Therefore, we can directly adopt Grover search to find a valid quartet in Algorithm 6 without affecting the overall attack complexity, which is different from Algorithm 5. The advantage is that the attack does not require the use of qRAMs. If the remaining key bits that need to be guessed are few, we adopt the quantum collision-finding algorithm at step 5 of Algorithm 6.

4.2 Second method

Our second method is based on quantum search and Ambainis's algorithm [26]. Recall that a good subkey guess is suggested 4 times in Biham et al.'s generic boomerang attacks. Thus, similar to the quantum related-key boomerang attack on AES-256 in [25], we can adopt Ambainis's element 4-distinctness algorithm to find these target, which requires $O(2^{4n/5})$ query complexity and $O(2^{4n/5})$ memory complexity in the QRAQM model. In the following, we show how to quantize Algorithm 1 in Algorithm 7.

Analysis of Algorithm 7. From Sect. 3.1.1, we know that the good key guess will be suggested 4 times among $y \cdot 2^{r_b+m_b-n-2}$ key candidates. In order to find the good

Algorithm 7 Another Quantum Boomerang Attack on $E = E_1 \circ E_0 \circ E_b$ in the QRAQM model

```

Input: superposition oracle access to  $E$  and  $E^{-1}$ 
Output: a guess of  $K_b$ 
1: function  $FuncA(i)$  ▷  $i$  is a  $\log_2(y \cdot 2^{m_b+r_b-n-2})$ -bit value
2:   Depending on  $i$ , choose a set of plaintext  $P_1$  with  $2^{2t_b-m_b+1}$  subsets, and each subset has  $2^{(n-r_b)/3}$ 
   plaintexts
3:   Repeat  $2^{2t_b-m_b+1}$  times
4:     Choose one of the  $2^{2t_b-m_b+1}$  subsets for  $P_1$ 
5:     Repeat  $2^{2r_b-2t_b}$  times
6:       /* Use the idea of BHT algorithm to find a valid quartet  $(P_1, P_2, P'_1, P'_2)$  */
7:       Create the list of  $2^{(n-r_b)/3}$  plaintexts  $P_1$ 
8:       Ask for the plaintext set  $S : \{E^{-1}(E(P_1) \oplus \delta)\}$ 
9:       Repeat  $2^{2(n-r_b)/3}$  times
10:        Choose plaintext  $P_2$  so that  $P_1 \oplus P_2 \in V_b$ 
11:        Compute  $P'_2 = E^{-1}(E(P_2) \oplus \delta)$ 
12:        Check if there is  $P'_1 \in S$  such that  $P'_1 \oplus P'_2 \in V_b$ 
13:        If there is, return  $(P_1, P_2, P'_1, P'_2)$ 
14:      EndRepeat
15:      If  $P'_1 \oplus P'_2 \in X_b$  and  $P_1 \oplus P_2 \in X_b$ 
16:      then return  $(P_1, P_2, P'_1, P'_2)$ 
17:    EndRepeat ▷ A valid boomerang quartet  $(P_1, P_2, P'_1, P'_2)$  is obtained
18:    /* Exhaust  $m_b$ -bit key  $K_b$  to obtain the corresponding key guess */
19:    Repeat  $2^{m_b}$  times
20:      If  $E_{b,K_b}(P_1) \oplus E_{b,K_b}(P_2) = \alpha = E_{b,K_b}(P'_1) \oplus E_{b,K_b}(P'_2)$  then return  $K_b$ 
21:    EndRepeat
22:  EndRepeat
23:  At this point, we have a (single) quartet and a key guess
24: end function
25: Apply Ambainis's 4-distinctness algorithm to  $FuncA$ 
26: Measure and return the single 4-collision among the key outputs of  $FuncA$ 

```

guess of key, we define a function $FuncA$ that can deterministically produce a valid quartet and an m_b -bit key guess K_b in Algorithm 7. Roughly speaking, $FuncA$ starts from a set of plaintext values. By using the idea of BHT algorithm, we perform a search of valid quartets on the space of this plaintext set. Next, we use the property of boomerang pairs to sieve them to obtain a single expected quartet. Then, we check whether it can yield a key guess, which can produce $y \cdot 2^{r_b+m_b-n-2}$ key guesses of K_b . Finally, we use Ambainis's element 4-distinctness algorithm to find the good guess of key.

The time complexity of Algorithm 7 can be computed as follows. Firstly, the probability that a quartet yields a key candidate is $2^{m_b-2t_b-1}$. Thus, there are $\max(1, 2^{-(m_b-2t_b-1)})$ iterations in step 3. Secondly, we use the idea of BHT to find a candidate quartet satisfying $(n - r_b)$ -bit collision in steps 7-14. We check whether the candidate quartet generates a key guess by repeating 2^{m_b} times in step 19. Let $a = \log_2(y \cdot 2^{m_b+r_b-n-2})$. According to Ambainis's element 4-distinctness algorithm [26], we need to perform $2^{4a/5}$ calls to the function $FuncA$ to obtain the good key guess. In addition, the **Repeat** loops inside the function can be regarded as nested

quantum searches, so the quantum time complexity of Algorithm 7 is about:

$$2^{4a/5} \left(2^{(2t_b - m_b + 1)/2} \left(2^{r_b - t_b} \left(2 \times 2^{(n - r_b)/3} + 2^{(n - r_b)/3} \right) + 2^{m_b/2} \right) \right). \tag{5}$$

Since the memory complexity of Ambainis’s 4-distinctness algorithm is the same as its query complexity, the attack requires $2^{4a/5}$ quantum memory in the QRAQM model. On the other hand, we also require $2^{(n - r_b)/3}$ quantum memory to find a candidate quartet. As a result, the attack works in the qRAM model and requires $\max(2^{4a/5}, 2^{(n - r_b)/3})$ quantum memory. According to Theorem 1, assuming that the numbers of iterations in *FuncA* are exact, we can obtain a more exact complexity (see Eq. (6)). Since the function *FuncA* can be seen as a subroutine (steps 3-13 of Algorithm 5) in Algorithm 5, we can also use Grover search instead of BHT to find a pair (P'_1, P'_2) such that $P'_1 \oplus P'_2 \in V_b$.

$$2 \cdot 2^{4a/5} \cdot \left(2 \left\lfloor \frac{\pi}{4} \cdot 2^{(2t_b - m_b + 1)/2} \right\rfloor \cdot \left(2 \left\lfloor \frac{\pi}{4} \cdot 2^{r_b - t_b} \right\rfloor \cdot \left(2 \cdot 2^{(n - r_b)/3} + 4 \left\lfloor \frac{\pi}{4} 2^{(n - r_b)/3} \right\rfloor \right) + 2 \left\lfloor \frac{\pi}{4} 2^{m_b/2} \right\rfloor \right) \right) \tag{6}$$

Similarly, we can construct a new quantum version for Algorithm 2, as shown in Algorithm 8. The main idea is to construct a function *FuncB* that can deterministically produce a valid quartet and an $(m_b + m_f)$ -bit key guess (K_b, K_f) . The function *FuncB* is equivalent to the steps 3-14 of Algorithm 6.

Since the analysis of Algorithm 8 is similar to Algorithm 7, we omit it here. Let $c = \log_2(y \cdot 2^{m_b + r_b + m_f - n - 3})$. By changing the loops in *FuncB* into nested quantum searches, the approximate quantum time complexity of Algorithm 8 is computed as follows:

$$2^{4c/5} \left(2^{(2t_b + 2t_f - m_b - m_f + 2)/2} \left(2^{r_b - t_b} \left(2^{(n - r_b)/2} + 2 \cdot 2^{t_f} \right) + 2^{(m_b + m_f)/2} \right) \right). \tag{7}$$

A more exact quantum time complexity is shown in Eq. (8). The attack also works in the QRAQM model and requires about $2^{4c/5}$ quantum memory.

$$2 \cdot 2^{4c/5} \cdot \left(2 \left\lfloor \frac{\pi}{4} \cdot 2^{(2t_b + 2t_f - m_b - m_f + 2)/2} \right\rfloor \cdot \left(2 \left\lfloor \frac{\pi}{4} \cdot 2^{r_b - t_b} \right\rfloor \cdot \left(2 \cdot 2^{(n - r_b)/2} + 2 \cdot 2^{t_f} \right) + 2 \left\lfloor \frac{\pi}{4} 2^{(m_b + m_f)/2} \right\rfloor \right) \right) \tag{8}$$

Remark 4 It is worth noting that another common situation in classical boomerang attacks is when $l(l > 4)$ boomerang pairs occur within the trials. In this case, we use Ambainis’s l -distinctness algorithm instead of 4-distinctness algorithm in Algorithms 7 and 8.

4.3 Comparison between the two methods

In this section, we propose two methods to convert the classical boomerang attacks of Biham et al. into some valid quantum attacks. We follow the parameters and symbol

Algorithm 8 Another Quantum Boomerang Attack on $E = E_f \circ E_1 \circ E_0 \circ E_b$ in the QRAQM model

Input: superposition oracle access to E and E^{-1}
Output: a guess of (K_b, K_f)

- 1: **function** $FuncB(i)$ $\triangleright i$ is a $\log_2(y \cdot 2^{m_b+r_b+m_f-n-3})$ -bit value
- 2: Depending on i , choose a set of plaintext P'_2 with $2^{2t_b+2t_f-m_b-m_f+2}$ subsets, and each subset has $2^{(n-r_b)}$ plaintexts
- 3: **Repeat** $2^{2t_b+2t_f-m_b-m_f+2}$ **times**
- 4: Choose one of the $2^{2t_b+2t_f-m_b-m_f+2}$ subsets for P'_2
- 5: **Repeat** $2^{2r_b-2t_b}$ **times**
- 6: */* Use the idea of Grover's algorithm to find a valid quartet (P_1, P_2, P'_1, P'_2) */*
- 7: Choose a plaintext P'_1 at random
- 8: In the search space of $2^{(n-r_b)}$, use Grover search to find a pair (P'_1, P'_2) such that $P'_1 \oplus P'_2 \in V_b$
- 9: **If** $P'_1 \oplus P'_2 \in X_b$, **then** compute $P_1 := E^{-1}(E(P'_1) \oplus \varepsilon)$ and $P_2 := E^{-1}(E(P'_2) \oplus \varepsilon)$
- 10: Check if there is (P_1, P_2) such that $P_1 \oplus P_2 \in X_b$
- 11: **If** there is, **then return** (P_1, P_2, P'_1, P'_2)
- 12: **EndRepeat** \triangleright A valid boomerang quartet (P_1, P_2, P'_1, P'_2) is obtained
- 13: Ask for the corresponding ciphertexts (C_1, C_2, C'_1, C'_2)
- 14: */* Exhaust $(m_b + m_f)$ -bit key (K_b, K_f) to obtain the corresponding key guess */*
- 15: **Repeat** $2^{m_b+m_f}$ **times**
- 16: **If** $E_{b,K_b}(P_1) \oplus E_{b,K_b}(P_2) = \alpha = E_{b,K_b}(P'_1) \oplus E_{b,K_b}(P'_2)$ and $E_{f,K_f}^{-1}(C_1) \oplus E_{f,K_f}^{-1}(C'_1) = \delta = E_{f,K_f}^{-1}(C_2) \oplus E_{f,K_f}^{-1}(C'_2)$ **then return** (K_b, K_f)
- 17: **EndRepeat**
- 18: **EndRepeat**
- 19: At this point, we have a (single) quartet and a key guess
- 20: **end function**
- 21: Apply Ambainis's 4-distinctness algorithm to $FuncB$
- 22: Measure and return the single 4-collision among the key outputs of $FuncB$

assumptions set by Biham et al., and give a detailed complexity analysis in Sect. 4.1 and 4.2. The complexity of each new algorithm we proposed is actually related to these parameters (i.e. $m_b, r_b, m_f, r_f, t_b, t_f, n, k$), which are given according to the differential paths proposed by the original author. The specific parameters and differential paths can refer to [10] and [27]. Therefore, we do not know which algorithm is better in our quantum boomerang attacks at the beginning. We need to substitute the parameters of the corresponding classical boomerang attack into the complexity formula for calculation and comparison to determine which algorithm is better.

Moreover, we will explain the intrinsic difference between the two methods here. In short, the difference between the two methods is that the idea of verifying the correctness of the subkey guess is different. The classical generic boomerang attack is divided into two steps: first, we recover partial subkey bits with the help of the right boomerang quartets; then, we exhaust the remaining key bits to recover the complete key. Our first method is to combine the above two steps of the classical attack directly, which is also an idea of Frixons et al [25]. In the quantum setting, it is a good strategy to discriminate the right key guess by checking exhaustively the remaining bits of key, as we did in step 15 of Algorithm 5 and step 16 of Algorithm 6. Therefore, in the

first method, we use the method of exhausting the remaining key bits to determine the correctness of the subkey guess, thus completing the key recovery attack.

Unlike the first method, our second method only corresponds to the first step of the classical boomerang attack, which can only recover the partial key first. In the classical generic boomerang attack, there are usually multiple valid boomerang quartets, and then multiple key guesses are generated. Here, we need to look for a key guess that appears at least 4 times among all the key guesses generated by the quartets. This is an element 4-distinctness problem. Classically, the time complexity of solving this problem is N . In the quantum setting, we can use Ambainis’s algorithm to accelerate the solution of this problem, and the solution obtained corresponds to the correct subkey guess that appears 4 times. This is the idea of Algorithm 7 and Algorithm 8. Therefore, the second method is to obtain the correct subkey by solving the element distinctness problem, while the first method is to obtain the correct subkey by exhausting the remaining key bits, which is the difference between the two methods. In addition, Algorithm 7 and Algorithm 8 can only recover the partial key. In order to recover the complete key, we need to exhaust the remaining key bits.

5 Application to Serpent-256

5.1 Brief description of Serpent

Serpent [35] is one of the candidates for the Advanced Encryption Standard. The block size of Serpent is 128 bits, and the key length can be chosen between 1 to 256 bits. In this paper, we focus on Serpent with an initial key of 256 bits.

Serpent has a 32-round SP-network operation on four 32-bit words (X_0, X_1, X_2, X_3) . Each round consists of three steps: key mixing, S boxes and linear transformation. Following the notations used in [35], we denote each intermediate value of the round i as \hat{B}_i and the round function as R_i , where $i = 0, 1, \dots, 31$. Each \hat{B}_i consists of four 32-bit words (X_0, X_1, X_2, X_3) . In addition, Serpent has 8 different 4×4 S boxes, marked as $S_i (i = 0, \dots, 7)$.

Let \hat{S}_i be the application of the S-box $S_{i \bmod 8}$ 32 times in parallel, and L be the linear transformation. The encryption algorithm of Serpent is described as follows:

$$\begin{aligned} \hat{B}_0 &:= P \\ \hat{B}_{i+1} &:= R_i(\hat{B}_i) \\ C &:= \hat{B}_{32} \end{aligned}$$

where

$$\begin{aligned} R_i(X) &= L(\hat{S}_i(X \oplus \hat{K}_i)) \quad i = 0, \dots, 30 \\ R_i(X) &= \hat{S}_i(X \oplus \hat{K}_i) \oplus \hat{K}_{32} \quad i = 31 \end{aligned}$$

The linear transformation L outputs X_0, X_1, X_2, X_3 :

$$\begin{aligned} X_0 &:= X_0 \lll 13, & X_2 &:= X_2 \lll 3, \\ X_1 &:= X_1 \oplus X_0 \oplus X_2, & X_3 &:= X_3 \oplus X_2 \oplus (X_0 \ll 3) \\ X_1 &:= X_1 \lll 1, & X_3 &:= X_3 \lll 7, \\ X_0 &:= X_0 \oplus X_1 \oplus X_3, & X_2 &:= X_2 \oplus X_3 \oplus (X_1 \ll 7) \\ X_0 &:= X_0 \lll 5, & X_2 &:= X_2 \lll 22 \end{aligned}$$

where \lll denotes bit rotation to the left, and \ll denotes bit shift to the left.

5.2 Quantum boomerang attack on 9-round Serpent-256

In [9], Biham et al. gave an 8-round rectangle distinguisher against Serpent-256. Assuming that Serpent-256 starts from the round 0, the distinguisher part ($E_1 \circ E_0$) covers from round 1 to round 8 of Serpent, where E_0 contains rounds 1-4 and E_1 covers rounds 5-8. The differential characteristics used by E_0 and E_1 are shown in [9], and their probabilities are represented as follows:

$$\sum_{\alpha \xrightarrow{\beta} \beta} \Pr^2(\alpha \rightarrow \beta) = 2^{-50.8}, \quad \sum_{\gamma \xrightarrow{\delta} \delta} \Pr^2(\gamma \rightarrow \delta) = 2^{-69.8}.$$

That is, we have $\hat{p} = 2^{-25.4}, \hat{q} = 2^{-34.9}$. Since the boomerang and rectangle distinguishers share the same α and δ , we can also use the same characteristics to obtain an 8-round boomerang distinguisher. The 8-round boomerang distinguisher requires about $4(\hat{p}\hat{q})^{-2} = 4 \cdot (2^{-25.4} \cdot 2^{-34.9})^{-2} = 2^{122.6}$ adaptive chosen plaintexts and ciphertexts. As shown in [10], the best time complexity of the classical boomerang attack on 9-round Serpent-256 is $2^{123.6}$ in the single-key setting.

In this paper, we use Algorithm 5 to construct a 9-round quantum key recovery attack on Serpent-256, which can be decomposed as follows: E_b is round 0, E_0 contains rounds 1-4 and E_1 covers rounds 5-8. According to [9, 10], we have $m_b = r_b = 76, t_b = 48.85, n = 128, k = 256$ in this case. To obtain 4 right quartets, we need $y = \frac{4 \cdot 2^{122.6}}{2^{76+1}} = 2^{47.6}$ structures.

According to Algorithm 5 and Eq. 2, the approximate quantum time complexity of our quantum boomerang attack on 9-round Serpent-256 is shown in Eq. 9. From Eq. 9, the time complexity of the 9-round quantum attack constructed by Algorithm 5 is $2^{124.8}$, which is lower than that of the generic quantum search but higher than that of the classical attack $2^{123.6}$. Therefore, using Algorithm 5 to attack 9-round Serpent-256 is not efficient. In order to solve this problem, we adopt Algorithm 7 to attack 9-round Serpent-256. According to Algorithm 7 and Eq. 5, we can obtain a new quantum time complexity, which is shown in Eq. 10. At this point, compared with the classical attack and quantum generic key search, our quantum attack has lower time complexity.

$$\begin{aligned}
 & 2^{\frac{(47.6+76+76-128-2)}{2}} \left(2^{\frac{(2 \times 48.85-76+1)}{2}} \left(2^{(76-48.85)} \cdot 2^{\frac{(128-76)}{3}} + 2^{\frac{76}{2}} \right) + 2^{\frac{(256-76)}{2}} \right) \tag{9} \\
 & \approx 2^{34.8} \left(2^{11.35} \left(2^{27.15} \cdot 2^{17.3} + 2^{38} \right) + 2^{90} \right) \approx 2^{124.8} < 2^{128} \\
 & 2^{\frac{(47.6+76+76-128-2) \times 4}{5}} \left(2^{\frac{(2 \times 48.85-76+1)}{2}} \left(2^{(76-48.85)} \left(2 \cdot 2^{\frac{(128-76)}{3}} + 2^{\frac{(128-76)}{3}} \right) + 2^{\frac{76}{2}} \right) \right) \tag{10} \\
 & \approx 2^{55.68} \left(2^{11.35} \left(2^{27.15} \left(2^{18.3} + 2^{17.3} \right) + 2^{38} \right) \right) \approx 2^{112.48}
 \end{aligned}$$

According to Eq. 6, when the numbers of iterations in Algorithm 7 are exact, we can further obtain a more accurate quantum time complexity (see Eq. 11 for details). At this time, the quantum time complexity of recovering the 76-bit subkey is $2^{115.43}$. After obtaining the 76-bit subkey, the remaining $(256 - 76)$ -bit key can be recovered in time 2^{90} by using Grover’s algorithm. Thus, the total quantum time complexity is $2^{115.43} + 2^{90} \approx 2^{115.43}$, which is still better than the generic quantum search algorithm (2^{128}) and classical attack ($2^{123.6}$). Therefore, we can construct an effective quantum key recovery attack on 9-round Serpent-256 by using Algorithm 7. Note that Algorithm 7 uses Ambainis’s 4-distinctness quantum algorithm which requires quantum memory with the same query complexity. Since we need to call the function $2^{55.68}$ times to attack 9-round Serpent-256 by using Algorithm 7, we also require $2^{55.68}$ quantum memory in the QRAQM model.

$$\begin{aligned}
 & 2 \cdot 2^{55.68} \cdot \left(2 \left\lfloor \frac{\pi}{4} 2^{11.35} \right\rfloor \cdot \left(2 \left\lfloor \frac{\pi}{4} 2^{27.15} \right\rfloor \cdot \left(2 \cdot 2^{17.3} + 4 \left\lfloor \frac{\pi}{4} 2^{17.3} \right\rfloor \right) + 2 \left\lfloor \frac{\pi}{4} 2^{38} \right\rfloor \right) \right) \tag{11} \\
 & \approx 2^{56.68} \left(2^{12} \left(2^{27.8} \left(2^{18.3} + 2^{18.95} \right) + 2^{38.65} \right) \right) \approx 2^{115.43}
 \end{aligned}$$

5.3 Quantum boomerang attack on 10-round Serpent-256

The 10-round boomerang attack on Serpent-256 uses the same 8-round distinguisher in Sect. 5.2. At this time, E_b is round 0, $E_1 \circ E_0$ covers rounds 1-8, and E_f is round 9. According to [9, 10], we have $m_b = r_b = 76, m_f = r_f = 20, t_b = 48.85, t_f = 13.6, n = 128, k = 256$. Since the 8-round boomerang distinguisher used is the same as the 9-round key recovery attack, the 10-round key recovery attack also requires $y = 2^{47.6}$ structures. As shown in [10], the best time complexity of the classical boomerang attack on 10-round Serpent-256 is $2^{173.8}$ in the single-key setting.

We can construct a quantum key recovery attack on 10-round Serpent-256 by using Algorithm 6. As shown in Eq. 12, the approximate time complexity of our 10-round quantum attack is $2^{124.3}$. According to Eq. 4, when the numbers of iterations in Algorithm 6 are exact, we obtain a more accurate time complexity by putting the additional factors of quantum search (see Eq. 13 for details). The time complexity of our attack obtained by Algorithm 6 is $2^{126.6}$, which is better than the generic quantum key search (2^{128}) and classical attack ($2^{173.8}$). Therefore, we can obtain a valid quantum key recovery attack on 10-round Serpent-256 by using Algorithm 6. It is noted that Grover’s algorithm is used to find valid quartets in Algorithm 6 instead of quantum collision-finding algorithm, so the quantum memory required for our quantum attack is negligible.

$$2^{\frac{(47.6+76+76+20-128-3)}{2}} \left(2^{\frac{(2 \times 48.85+2 \times 13.6-76-20+2)}{2}} \left(2^{(76-48.85)} \cdot 2^{\frac{52}{2}} + 2^{\frac{96}{2}} \right) + 2^{\frac{(256-96)}{2}} \right) \tag{12}$$

$$\approx 2^{44.3} (2^{15.45} (2^{27.15} \cdot 2^{26} + 2^{48}) + 2^{80}) \approx 2^{124.3} < 2^{128}$$

$$2 \lfloor \frac{\pi}{4} 2^{44.3} \rfloor \cdot (2 \lfloor \frac{\pi}{4} 2^{15.45} \rfloor \cdot (2 \lfloor \frac{\pi}{4} 2^{27.15} \rfloor \cdot 2 \lfloor \frac{\pi}{4} 2^{26} \rfloor + 2 \lfloor \frac{\pi}{4} 2^{48} \rfloor) + 4 \lfloor \frac{\pi}{4} 2^{80} \rfloor) \tag{13}$$

$$\approx 2^{44.95} (2^{16.1} (2^{54.45} + 2^{48.65}) + 2^{81.65}) \approx 2^{126.6} < 2^{128}$$

After our attempt, we find that using the second method in this paper can not convert the classical 10-round boomerang attack into a valid quantum boomerang attack. Therefore, we do not describe it in detail here.

6 Application to ARIA-196

Kwon et al. [36] proposed an AES-like block cipher named ARIA at ICISC’03. ARIA has a 128-bit block size and a key length of 128, 192 or 256 bits. The overall structure of ARIA is a substitution and permutation network. The round function of ARIA is composed of the following three operations: (1) Round Key Addition: the 128-bit round key generated by the key schedule is XORed to the intermediate state. (2) Substitution Layer: ARIA uses two different S-boxes to construct two types of substitution layers for odd and even rounds. (3) Diffusion Layer: it is an involutory linear transformation $P : GF(2^8)^{16} \rightarrow GF(2^8)^{16}$. For more details about ARIA cipher, please refer to [36].

In this section, we apply our methods to the version of ARIA with a 196-bit key. Fleischmann et al. [27] constructed a classical key recovery attack on 6-round ARIA-196 based on a 5-round boomerang distinguisher. According to [27], the probability of the 5-round distinguisher is 2^{-105} and adversaries need $y = 2^{53}$ structures to mount a 6-round attack. In addition, we have $m_b = r_b = 0, m_f = r_f = 56, t_f = 38.08, n = 128, k = 196$ in this case. As shown in [27], the best time complexity of the classical boomerang attack on 6-round ARIA-196 is 2^{108} in the single-key setting.

We can construct a valid quantum boomerang attack on 6-round ARIA by using Algorithm 5. The quantum time complexity is about $2^{87.5}$, as shown in Eq. 14. We can see that the time complexity is below the complexity of the Grover search (2^{98}) and the complexity of the classical attack (2^{108}). Assuming that the numbers of iterations are exact in Algorithm 5, we obtain a more accurate quantum time complexity (see Eq. 15 for details).

$$2^{\frac{(53+56+56-128-2)}{2}} \left(2^{\frac{(2 \times 38.08-56+1)}{2}} \left(2^{(56-38.08)} \cdot 2^{\frac{(128-56)}{2}} + 2^{\frac{56}{2}} \right) + 2^{\frac{(196-56)}{2}} \right) \tag{14}$$

$$\approx 2^{17.5} (2^{10.58} (2^{17.92} \cdot 2^{36} + 2^{28}) + 2^{70}) \approx 2^{87.5} < 2^{98}$$

$$2 \lfloor \frac{\pi}{4} 2^{17.5} \rfloor \cdot (2 \lfloor \frac{\pi}{4} 2^{10.58} \rfloor \cdot (2 \lfloor \frac{\pi}{4} 2^{17.92} \rfloor \cdot 4 \cdot 2^{36} + 2 \lfloor \frac{\pi}{4} 2^{28} \rfloor) + 4 \lfloor \frac{\pi}{4} 2^{70} \rfloor) \tag{15}$$

$$\approx 2^{18.15} (2^{11.23} (2^{18.57} \cdot 2^{38} + 2^{28.65}) + 2^{71.65}) \approx 2^{89.8}$$

The final time complexity is still better than the classical attack and quantum generic key recovery attack. Therefore, we can use Algorithm 5 to obtain a valid quantum key recovery attack on 6-round ARIA-196. Note that we use Grover search instead of

quantum collision-finding algorithm to find a valid pair in step 5 of Algorithm 5 here, so the quantum memory required for our 6-round quantum attack is negligible. In addition, we find that using Algorithm 7 to attack 6-round ARIA-196 is not more effective than using Algorithm 5, so we omit it here.

7 Conclusion

Based on the works of Frixons et al. and Biham et al., this paper proposes two more generic quantum boomerang attacks and applies them to Serpent-256 and ARIA-196. For Serpent-256, we construct valid 9 and 10 rounds of quantum key recovery attacks respectively. For ARIA-196, we construct a valid 6-round quantum key recovery attack. Compared with the quantum generic key recovery attack, our attack can reduce the time complexity by a factor of $2^{12.57}$ at most. Therefore, the attacks proposed in this paper are meaningful. Our results show that some classical cryptanalysis methods still can offer the valuable reference in the quantum setting. In addition, our attacks are based on the differential paths of the existing boomerang attacks, which seems to limit the improvement effect of our attacks. Therefore, this is also a problem that needs to be solved in the future work, that is, whether we can find new and more suitable differential paths for the target block ciphers to further improve the time complexity of our attacks, and even increase the number of rounds that can be attacked. Moreover, whether there are other classical cryptanalysis techniques that can be converted into valid quantum attacks is an interesting open question.

On the other hand, in this paper we only consider quantum boomerang attacks in the single-key setting. But in fact, many classical boomerang or rectangle attacks work in the related-key setting. Whether we can construct effective quantum boomerang or rectangle attacks in the related-key setting deserves further study.

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

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Liu, W.-B., Li, C.-L., Xie, Y.-M., Weng, C.-X., Jie, G., Cao, X.-Y., Yu-Shuo, L., Li, B.-H., Yin, H.-L., Chen, Z.-B.: Homodyne detection quadrature phase shift keying continuous-variable quantum key distribution with high excess noise tolerance. *PRX Quant.* **2**, 040334 (2021)
2. Xie, Y.-M., Yu-Shuo, L., Weng, C.-X., Cao, X.-Y., Zhao-Ying Jia, Yu., Bao, Y.W., Yao, F., Yin, H.-L., Chen, Z.-B.: Breaking the rate-loss bound of quantum key distribution with asynchronous two-photon interference. *PRX Quant.* **3**, 020315 (2022)
3. Gu, J., Cao, X.Y., Fu, Y., He, Z.W., Yin, Z.J., Yin, H.L., Chen, Z.B.: Experimental measurement-device-independent type quantum key distribution with flawed and correlated sources. *Sci. Bull.* **67**(21), 2167–2175 (2022)

4. Yin, H.-L., Yao, F., Li, C.-L., Weng, C.-X., Li, B.-H., Jie, G., Yu-Shuo, L., Huang, S., Chen, Z.-B.: Experimental quantum secure network with digital signatures and encryption. *Natl. Sci. Rev.* **10**, 228 (2022)
5. Zhou, M.-G., Cao, X.-Y., Yu-Shuo, L., Yang Wang, Yu., Bao, Z.-Y.J., Yao, F., Yin, H.-L., Chen, Z.-B.: Experimental quantum advantage with quantum coupon collector. *Research* **2022**, 1–11 (2022)
6. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. *J. Cryptol.* **4**(1), 3–72 (1991)
7. Wagner, D.: The boomerang attack. In: Knudsen, L.R. (ed.) *Fast Software Encryption*, 6th International Workshop, FSE '99, Rome, Italy, March 24–26, 1999, Proceedings. *Lecture Notes in Computer Science*, vol. 1636, pp. 156–170. Springer (1999)
8. Kelsey, J., Kohno, T., Schneier, B.: Amplified boomerang attacks against reduced-round MARS and serpent. In: Schneier, B. (ed.) *Fast Software Encryption*, 7th International Workshop, FSE 2000, New York, NY, USA, April 10–12, 2000, Proceedings. *Lecture Notes in Computer Science*, vol. 1978, pp. 75–93. Springer (2000)
9. Biham, E., Dunkelman, O., Keller, N.: The rectangle attack - rectangling the serpent. In: Pfitzmann, B. (ed.) *Advances in Cryptology - EUROCRYPT 2001*, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6–10, 2001, Proceeding. *Lecture Notes in Computer Science*, vol. 2045, pp 340–357. Springer (2001)
10. Biham, E., Dunkelman, O., Keller, N.: New results on boomerang and rectangle attacks. In: Daemen, J., Rijmen, V., (eds.) *Fast Software Encryption*, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4–6, 2002, Revised Papers. *Lecture Notes in Computer Science*, vol. 2365, pp. 1–16. Springer (2002)
11. Zhao, B., Dong, X., Jia, K.: New related-tweakey boomerang and rectangle attacks on Deoxys-BC including BDT effect. *IACR Trans. Symmetric Cryptol.* **2019**(3), 121–151 (2019)
12. Biryukov, A., Khovratovich, D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: Matsui, M. (ed.) *Advances in Cryptology - ASIACRYPT 2009*, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6–10, 2009. Proceedings. *Lecture Notes in Computer Science*, vol. 5912, pp. 1–18. Springer (2009)
13. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Miller, G.L. (ed.) *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, Philadelphia, Pennsylvania, USA, May 22–24, 1996, pp. 212–219. ACM (1996)
14. Simon, D.R.: On the power of quantum computation. *SIAM J. Comput.* **26**(5), 1474–1483 (1997)
15. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Breaking symmetric cryptosystems using quantum period finding. In: Robshaw, M., Katz, J., (ed.) *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference*, Santa Barbara, CA, USA, August 14–18, 2016, Proceedings, Part II. *Lecture Notes in Computer Science*, vol. 9815, pp. 207–237. Springer (2016)
16. 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 - The Cryptographers' Track at the RSA Conference 2019*, San Francisco, CA, USA, March 4–8, 2019, Proceedings. *Lecture Notes in Computer Science*, vol. 11405, pp. 391–411. Springer (2019)
17. Bonnetain, X., Leurent, G., Naya-Plasencia, M., Schrottenloher, A.: Quantum linearization attacks. In: Tibouchi, M., Wang, H., (eds.) *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security*, Singapore, December 6–10, 2021, Proceedings, Part I. *Lecture Notes in Computer Science*, vol. 13090, pp. 422–452. Springer (2021)
18. Hosoyamada, A., Sasaki, Y.: Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. In: Canteaut, A., Ishai, Y. (eds.) *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II. *Lecture Notes in Computer Science*, vol. 12106, pp. 249–279. Springer (2020)
19. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The rebound attack: cryptanalysis of reduced whirlpool and grøstl. In: Dunkelman, O. (ed) *Fast Software Encryption*, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22–25, 2009, Revised Selected Papers. *Lecture Notes in Computer Science*, vol. 5665, pp. 260–276. Springer (2009)
20. Lamberger, M., Mendel, F., Schläffer, M., Rechberger, C., Rijmen, V.: The rebound attack and subspace distinguishers: Application to whirlpool. *J. Cryptol.* **28**(2), 257–296 (2015)

21. Hosoyamada, A., Sasaki, Y.: Quantum collision attacks on reduced SHA-256 and SHA-512. In: Malkin T., Peikert, C. (eds.) *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part I. Lecture Notes in Computer Science*, vol. 12825, pp. 616–646. Springer (2021)
22. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Quantum differential and linear cryptanalysis. *IACR Trans. Symmetric Cryptol.* **2016**(1), 71–94 (2016)
23. Hosoyamada, A., Sasaki, Y.: Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In: Smart, N.P. (ed.) *Topics in Cryptology - CT-RSA 2018 - The Cryptographers' Track at the RSA Conference 2018, San Francisco, CA, USA, April 16–20, 2018, Proceedings. Lecture Notes in Computer Science*, vol. 10808, pp. 198–218. Springer (2018)
24. Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: On quantum slide attacks. In: Paterson, K.G., Stebila, D., (eds.) *Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12–16, 2019, Revised Selected Papers. Lecture Notes in Computer Science*, vol. 11959, pp. 492–519. Springer (2019)
25. Frixons, P., Naya-Plasencia, M., Schrottenloher, A.: Quantum boomerang attacks and some applications. In: AlTawy, R., Hülsing, A. (eds.) *Selected Areas in Cryptography - 28th International Conference, SAC 2021, Virtual Event, September 29 - October 1, 2021, Revised Selected Papers. Lecture Notes in Computer Science*, vol. 13203, pp. 332–352. Springer (2021)
26. Ambainis, A.: Quantum walk algorithm for element distinctness. *SIAM J. Comput.* **37**(1), 210–239 (2007)
27. Fleischmann, E., Forler, C., Gorski, M., Lucks, S.: New boomerang attacks on ARIA. In: Gong, G., Gupta, K.C. (eds.) *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12–15, 2010. Proceedings. Lecture Notes in Computer Science*, vol. 6498, pp. 163–175. Springer (2010)
28. Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information. Mathematical Structures in Computer Science* (2002)
29. Kuperberg, G.: Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. In: Severini, S., Brandão, F.G.S.L. (eds.) *8th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2013, May 21–23, 2013, Guelph, Canada. LIPIcs*, vol. 22, pp. 20–34. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2013)
30. Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. *AMS Contemp. Math. Series* **305**, 53–74 (2000)
31. Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: Quantum security analysis of AES. *IACR Trans. Symmetric Cryptol.* **2019**(2), 55–93 (2019)
32. Brassard, G., Høyer, P., Tapp, A.: Quantum cryptanalysis of hash and claw-free functions. In: Lucchesi, C.L., Moura, A.V. (eds.) *LATIN '98: Theoretical Informatics, Third Latin American Symposium, Campinas, Brazil, April, 20–24, 1998, Proceedings. Lecture Notes in Computer Science*, vol. 1380, pp. 163–169. Springer (1998)
33. Yuen, H.: A quantum lower bound for distinguishing random functions from random permutations. *Quantum Inf. Comput.* **14**(13–14), 1089–1097 (2014)
34. Childs, A.M., Eisenberg, J.M.: Quantum algorithms for subset finding. *Quantum Inf. Comput.* **5**(7), 593–604 (2005)
35. Anderson, R., Biham, E., Knudsen, L.: Serpent: A proposal for the advanced encryption standard. Technical report, NIST AES Proposal (1998)
36. Kwon, D., Kim, J., Park, S., Sung, S.H., Sohn, Y., Song, J.H., Yeom, Y., Yoon, E.J., Lee, S., Lee, J., Chee, S., Han, D., Hong, J.: New block cipher: ARIA. In: Lim, J.I., Lee, D.H. (ed.) *Information Security and Cryptology - ICISC 2003, 6th International Conference, Seoul, Korea, November 27–28, 2003, Revised Papers. Lecture Notes in Computer Science*, vol. 2971, pp. 432–445. Springer (2003)

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.