IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2024, No. 3, pp. 238–265. DOI:10.46586/tosc.v2024.i3.238-265

Improved Quantum Rebound Attacks on Double Block Length Hashing with Round-Reduced AES-256 and ARIA-256

Dongjae Lee[1] and Seokhie Hong[2](✉)

[1] Department of Convergence Security, Kangwon National University, Chuncheon, Republic of Korea
dongjae.lee@kangwon.ac.kr
[2] School of Cybersecurity, Korea University, Seoul, Republic of Korea
shhong@korea.ac.kr

Abstract. At EUROCRYPT 2020, Hosoyamada and Sasaki proposed the first dedicated quantum collision attacks on hash functions. Their proposal presented a quantum adaptation of the rebound attack and revealed that differential trails, which have too low probability for use in classical settings, might be exploitable in quantum settings. After their work, subsequent research has actively delved into analyzing the security of hash functions in the quantum setting.

In this paper, we revisit the quantum rebound attacks on the double block hash function Hirose instantiated with 10-round `AES-256` (`HCF-AES-256`) and 7-round `ARIA-256` (`HCF-ARIA-256`) proposed by Chauhan et al. and Baek et al., respectively. Initially, we identify the flaws in their work and reevaluate the complexity of the attacks. We reveal that the flaws stem from not considering the issue that the S-box differential equation has one solution *on average*. Earlier research addressed this problem by adding auxiliary bits to the search space. If this method is used to correct the flaws, the resulting time complexities are $2^{17.36}$ and $2^{20.94}$ times higher than their proposals. Consequently, in some settings, their attacks become less efficient than generic attacks.

Subsequently, we propose improved quantum rebound attacks using nested quantum amplitude amplification and quantum state preparation. Our improved attack efficiently pre-filters the search space, leading to a reduction in overall time complexity. We first classically reduce the search space and employ quantum state preparation to generate a superposition state over the pre-filtered search space. We then use nested quantum amplitude amplification to further reduce the search space quantumly. As a result, we achieve a reduction in the time complexity of the quantum rebound attacks on `HCF-AES-256` and `HCF-ARIA-256` by factors of $2^{11.2}$ and $2^{19.5}$, respectively, making the attacks more efficient than generic attacks again.

Keywords: Quantum cryptanalysis · Quantum Rebound Attack · Nested Quantum Amplitude Amplification · Quantum State Preparation

1 Introduction

Shor showed that with a sufficiently large quantum computer, the factorization and discrete logarithm problems, upon which the security of widely-used public-key cryptographic schemes such as RSA and ECC depends, could be solved in polynomial time [Sho94]. Consequently, post-quantum security has attracted substantial attention within the cryptography community.

Received: 2024-03-01 Revised: 2024-06-01 Accepted: 2024-08-01 Published: 2024-09-06

In light of this, NIST initiated the standardization of quantum-resistant public-key cryptography in 2016 [Moo16]. This effort led to the selection of the first set of algorithms in July 2022: CRYSTALS-KYBER [ABD+21], CRYSTALS-DILITHIUM [BDK+21], FALCON [FHK+20], and SPHINCS+ [ABB+22]. Building on these selections, NIST recently published drafts of three FIPS: FIPS 203, FIPS 204, and FIPS 205 [Nat23b, Nat23a, Nat23c].

In contrast, post-quantum security for symmetric-key primitives has received comparatively limited attention. For approximately two decades, a quadratic speedup achieved by Grover search [Gro96] was considered the primary advantage in the cryptanalysis of symmetric-key cryptographic primitives on quantum computers. This potential vulnerability can be mitigated by doubling the key length.

In 2010, Kuwakado and Morii showed that classically provably secure 3-round Feistel network could be broken within polynomial time using Simon's algorithm [KM10]. This result sparked significant interest in the symmetric-key community regarding post-quantum security. After this work, numerous works have introduced dedicated quantum attacks beyond Grover search on Even-Mansour construction [KM12], FX-constructions [LM17], block ciphers [BNS19], CBC-like MACs [KLLN16], and authenticated encryption algorithm [Bon17].

For hash functions, several generic quantum collision finding algorithms have been proposed, and among them, the best algorithm depends on how the quantum environment is realized in practice.

Quantum Settings and Generic Collision Attacks The first quantum setting assumes that an exponentially large quantum random access memory (qRAM) is available. In this setting, the best algorithm is the `BHT` algorithm presented by Brassard et al. [BHT98]. Brassard et al. showed that with a qRAM size of $O(2^{n/3})$, the collisions of an n-bit ideal hash function can be found with a time complexity of $O(2^{n/3})$. We call this quantum setting Q-I.

The second quantum setting assumes that qubits for computation and qubits for quantum memory are physically realized in the same way, and therefore, parallelization is taken into account when comparing quantum algorithms. Specifically, the efficiency of quantum algorithms is measured by the tradeoff between time T and space S, where S denotes the maximum size of the quantum memory and classical memory. The best collision finding algorithm in the second quantum setting is the parallel rho method [vW94], which gives the tradeoff $T = 2^{n/2}/S$ [Ber09]. We call this quantum setting Q-II.

The third quantum setting assumes that only a small polynomial-sized quantum computer is available and that quantum memory is more expensive than classical memory. In this setting, the best collision-finding algorithm is the `CNS` algorithm proposed by Chailloux et al. [CNS17]. The `CNS` algorithm finds collisions with a time complexity of $O(2^{2n/5})$, using a quantum memory of size $\tilde{O}(1)$ and classical memory of size $O(2^{n/5})$. We call this quantum setting Q-III.

Dedicated Quantum Collision Attacks The first dedicated quantum collision attack on hash functions was proposed by Hosoyamada and Sasaki in 2020 [HS20]. They introduced a quantum version of the rebound attack [MRST09] on 7-round `AES-MMO` and 6-round `Whirlpool`. Their work also has the implication of demonstrating that differential trails, with probabilities too low for classical use, can be effective in the quantum setting. Since their work, collision attacks on hash functions have been actively studied. Dong et al. presented quantum collision attacks on `AES`-like hashing with low qRAM [DSS+20]. Chauhan et al. introduced quantum collision attacks on double block length hashing Hirose instantiated with round-reduced `AES-256` [CKS21]. Dou et al. and Baek et al. presented quantum collision attacks on `ARIA`-based hash functions [DMLQ21, BK23].

1.1 Contribution of this paper

In this paper, we revisit the previous quantum rebound attacks on Hirose's double block length hashing instantiated with round-reduced AES-256 (HCF-AES-256) [CKS21] and ARIA-256 (HCF-ARIA-256) [BK23]. We begin by identifying flaws in their attacks and subsequently presenting the correct complexities. Additionally, we improve the attacks in terms of time complexity using some quantum algorithms.

The flaws in their work stem from the distinction between the classical and quantum settings regarding the fact that the S-box differential equation has one solution "on average". For the purpose of explanation, we will briefly describe the rebound attack in the following two steps:

1. Solve several S-box differential equations.

2. For all solution, check whether they fulfill specific conditions.

In the classical setting, it is easy to skip Step 2 when there is no solution in Step 1. However, this is not the case in the quantum setting. In the quantum setting, we encapsulate the entire process as a function F, construct the associated quantum gate $\mathcal{U}_F$, and apply Grover search or Quantum Amplitude Amplification (QAA) to it. Therefore, skipping Step 2 is non-trivial in the quantum setting. Additionally, when there are multiple solutions in Step 1, examining all these solutions is also non-trivial in the quantum setting.

These aspects have already been discussed in previous works [HS20, DSS$^+$20]. To resolve the issue, auxiliary bits were added to the search space, aiding the exploration of all solutions in Step 2 and resulting in an increase in time complexity. While [CKS21, BK23] also mentioned these aspects, they did not take them into account when evaluating the complexity of the attack. Based on these considerations, we discuss the flaws in [CKS21, BK23] and present the complexity required for their proposed attacks to function correctly. Consequently, we find that the attacks, in some quantum settings, become less efficient than generic attacks.

Subsequently, we improve the attacks using nested QAA and quantum state preparation. The previous attacks applied quantum search to the entire process, while our approach uses nested QAA to first perform an inner QAA that only checks for the existence of solutions in Step 1. This allows us to achieve a similar effect as skipping Step 2 in the classical setting when no solution is found in Step 1.

Additionally, we propose a method to efficiently pre-filter the search space classically, followed by generating the superposition state over the pre-filtered search space using quantum state preparation. By employing this method in the inner QAA, we reduce the number of iterations required, leading to a decrease in overall time complexity. We find a method to reduce the search space by a factor of 2^8 for the quantum rebound attack on HCF-AES-256.[1]

As a result, our improved quantum rebound attack achieves improvements in terms of time complexity by factors of $2^{8.69} \sim 2^{11.2}$ and $2^{19.5}$ for HCF-AES-256 and HCF-ARIA-256, respectively. A summary of the comparison between previous works and our improved attacks in each quantum setting is provided in Table 1. In Table 1, results that contain flaws are marked as 'Flawed', and results that are less efficient than generic attacks are marked as 'Invalid'. The time complexity is measured in encryption units, using 10-round AES-256 encryption for attacks on HCF-AES-256 and 7-round ARIA-256 encryption for attacks on HCF-ARIA-256.

[1]We also find a method to reduce the search space by a factor of 2^{19} for the quantum rebound attack on HCF-ARIA-256. However, its impact on the overall complexity was negligible, and thus we detail the pre-filtering process for HCF-ARIA-256 in Appendix A.

Table 1: Comparison of previous works with ours.

Comparison in the Q-I setting

Target	Time	qRAM	Reference	Remark
HCF-AES-256	2^{120}	2^{16}	[BHT98]	-
	$2^{85.33}$	$2^{85.33}$	[BHT98]	-
	$2^{85.11}$	2^{16}	[CKS21]	Flawed
	$2^{102.47}$	2^{16}	Section 3.2	-
	$2^{91.19}$	2^{16}	Section 4.2	-

Comparison in the Q-II setting

Target	Time	Space	Reference	Remark
HCF-AES-256	$2^{128}/S$	S	[vW94]	-
	$2^{88.61}/\sqrt{S/2^4}$	$2^4 \le S \le 2^{74.78}$	[CKS21]	Flawed
	$2^{105.97}/\sqrt{S/2^4}$	$2^4 \le S \le 2^{40.06}$	Section 3.2	-
	$2^{94.74}/\sqrt{S/2^4}$	$2^4 \le S < 2^{62.52}$	Section 4.2	-
HCF-ARIA-256	$2^{128}/S$	S	[vW94]	-
	$2^{118.56}/\sqrt{S/2^{2.54}}$	$2^{2.54} \le S \le 2^{16.34}$	[BK23]	Flawed
	$2^{139.50}/\sqrt{S/2^{2.54}}$	-	Section 3.3	Invalid
	$2^{120.00}/\sqrt{S/2^{2.54}}$	$2^{2.54} \le S \le 2^{13.46}$	Section 4.3	-

Comparison in the Q-III setting

Target	Time	C-Memory	Reference	Remark
HCF-AES-256	$2^{102.4}$	$2^{51.2}$	[CNS17]	-
	$2^{86.07}$	0	[CKS21]	Flawed
	$2^{103.43}$	0	Section 3.2	Invalid
	$2^{94.74}$	0	Section 4.2	-

Invalid: Less efficient than generic attack

1.2 Organization of this paper

Section 2 introduces the necessary preliminaries, including a brief description of the target primitives' specifications and an overview of quantum algorithms such as quantum amplitude amplification and quantum state preparation. In Section 3, we analyze previous quantum rebound attacks, discuss flaws in these attacks, and reevaluate the complexities using existing ideas. In Section 4, we propose improved quantum rebound attacks using nested quantum amplitude amplification and quantum state preparation. With our approach, we improve the quantum rebound attacks on HCF-AES-256 and HCF-ARIA-256 in terms of time complexity. We conclude the paper in Section 5.

2 Preliminaries

Both AES and ARIA have a block size of 128 bits. We represent a 128-bit internal state A as follows, where $A[i]$ denotes the i-th byte of A.

$$A = \begin{pmatrix} A[0] & A[4] & A[8] & A[12] \\ A[1] & A[5] & A[9] & A[13] \\ A[2] & A[6] & A[10] & A[14] \\ A[3] & A[7] & A[11] & A[15] \end{pmatrix}.$$

2.1 Description of AES

AES has been the most widely used block cipher since it was selected as an encryption standard at an open competition organized by NIST in 2000 [DR99]. It adopts a substitution-permutation network (SPN) structure, and its round function consists of four operations: SubBytes (SB), ShiftRows (SR), MixColumns (MC), and AddRoundKey (ARK). AES has three variants: AES-128, AES-192, and AES-256. The number of rounds in each variant is 12, 14, and 16, with key sizes of 128-bit, 192-bit, and 256-bit, respectively. The MixColumns operation in the last round is omitted.

A concise overview of the four operations is provided below. For more comprehensive information, please refer to [DR99]. We denote the state before SB, SR, MC, and ARK in round i as X_i, Y_i, Z_i, and W_i, respectively. The round key XORed in round i is denoted as K_i.

- SubBytes: S-box operation is applied on each byte of the state.

- ShiftRows: The state is permuted as follows:

$$\begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{pmatrix} \rightarrow \begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_5 & a_9 & a_{13} & a_1 \\ a_{10} & a_{14} & a_2 & a_6 \\ a_{15} & a_3 & a_7 & a_{11} \end{pmatrix}.$$

- MixColumns: Each column of the state is multiplied by a fixed matrix.

- AddRoundKey: The 128-bit round key is XORed to the state.

2.2 Description of ARIA

ARIA is a block cipher proposed in ICISC 2003 by Kwon et al. and has been the Korean encryption standard since 2004 [KKP+04]. It adopts an SPN structure and its round function consists of three operations: Substitution Layer (SL), Diffusion Layer (DL), and Round Key Addition (RKA). ARIA has three variants: ARIA-128, ARIA-192, and ARIA-256. The number of rounds in each variant is 12, 14, and 16, with key sizes of 128-bit, 192-bit, and 256-bit, respectively. The Diffusion Layer operation of the last round is omitted.

A concise overview of the three operations is provided below. For more comprehensive information, please refer to [KKP+04]. We denote the state before SL, DL, and RKA in round i as X_i, Y_i, and Z_i, respectively. The round key XORed in round i is denoted as K_i.

- Substitution Layer: S-box operation is applied on each byte of the state.

- Diffusion Layer: Multiply the state by a matrix M_{DL}. M_{DL} is defined as follows:

$$M_{\mathsf{DL}} = \begin{pmatrix}
0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0\\
0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1\\
0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1\\
1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0\\
1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1\\
0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1\\
1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0\\
0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0\\
1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1\\
1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0\\
0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1\\
0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0\\
0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0\\
1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0\\
1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1
\end{pmatrix}.$$

- Round Key Addition: The 128-bit round key is XORed to the state.

2.3 Hirose's double block length compression function

Hirose presented the construction of a compression function that outputs $2n$-bit using a component function that produces n-bit output [Hir05, Hir06]. The author proposed two constructions: one based on a random oracle and the other on a block cipher. In this paper, we focus on the construction based on a block cipher.

Let E be a block cipher with a block size of n bits and a key size of $2n$ bits. Then, Hirose's Double Block Length Compression Function (`HCF`) instantiated with E, `HCF`-E, can be defined as Definition 1. The overall structure is shown in Figure 1.

Definition 1. Let `HCF`-E: $\{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^{2n}$ be a compression function such that $(g_i, h_i) = \texttt{HCF-}E(g_{i-1}, h_{i-1}, m_i)$, where $g_i, h_i, m_i \in \{0,1\}^n$. The function `HCF`-$E$ produces the outputs as follows:

$$\begin{cases} g_i = E_{h_{i-1}||m_i}(g_{i-1}) \oplus g_{i-1} \\ h_i = E_{h_{i-1}||m_i}(g_{i-1} \oplus c) \oplus g_{i-1} \oplus c, \end{cases}$$

where $c \in \{0,1\}^n - \{0^n\}$ is a constant.

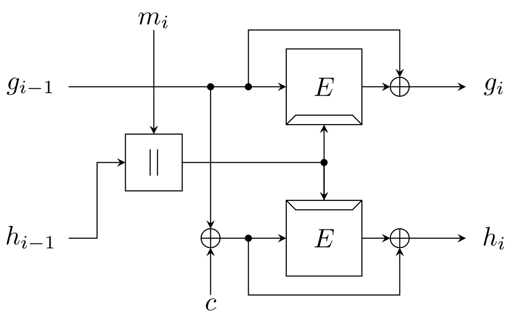

Figure 1: Hirose's double block length compression function.

2.4 Quantum amplitude amplification

Quantum Amplitude Amplification (QAA) [BHMT02] is a generalization of Grover search [Gro96]. For a quantum algorithm $\mathcal{A}$ that generates a superposition state within the search space, QAA amplifies the amplitude of the good state, thereby increasing the probability of measuring the desired outcome. While Grover search utilizes Hadamard gates to generate a uniform superposition of all possible n-bit states, $\mathcal{A}$ does not need to generate a superposition of all states, and it is acceptable even if the amplitudes are not equal. The formal description of QAA is given in Theorem 1.

Theorem 1 (QAA [BHMT02])**.** *Let $\mathcal{A}$ be any measurement-free quantum algorithm such that*

$$\mathcal{A}\,|0\rangle = \sqrt{p}\,|\psi_G\rangle\,|1\rangle + \sqrt{1-p}\,|\psi_B\rangle\,|0\rangle,$$

where $|\psi_G\rangle$ is the superposition of all good states and $|\psi_B\rangle$ is the superposition of all bad states, and $\sqrt{p}$ is the amplitude of $|\psi_G\rangle$. A good state is defined as a state that satisfies a specific condition or property of interest, while a bad state does not satisfy this condition. Let S_0 be a reflection operator around $|0\rangle$:

$$S_0\,|x\rangle = \begin{cases} |x\rangle & \text{if } x = 0, \\ -\,|x\rangle & \text{otherwise.} \end{cases}$$

Let S_{test} be a reflection operator around the good states:

$$S_{test}\,|x\rangle = \begin{cases} -\,|x\rangle & \text{if } x \text{ is good,} \\ |x\rangle & \text{otherwise.} \end{cases}$$

If we apply the quantum operator $-\mathcal{A}S_0\mathcal{A}^{-1}S_{test}$ to the state $\mathcal{A}\,|0\rangle$ for $\lfloor \frac{\pi}{4\,arcsin\sqrt{p}} \rfloor$ times and measure the system, the probability of observing a good state is at least $\max(1-p, p)$.

We denote the cost of a quantum algorithm $\mathcal{A}$ or a quantum gate $\mathcal{U}$ as $\mathcal{C}(\mathcal{A})$ and $\mathcal{C}(\mathcal{U})$, respectively. The complexity of QAA using quantum algorithm $\mathcal{A}$ is denoted as $\mathcal{C}_{\mathrm{QAA}}(\mathcal{A})$. Note that each iteration of QAA involves two calls to $\mathcal{A}$. Therefore, we evaluate $\mathcal{C}_{\mathrm{QAA}}(\mathcal{A})$ as

$$\frac{\pi}{2\sqrt{p}} \cdot \mathcal{C}(\mathcal{A}). \tag{1}$$

Nested QAA Since QAA does not include measurement, it can be part of a quantum algorithm, allowing for nested QAA. Let's consider a quantum algorithm $\mathcal{A}'$ consisting of QAA using quantum algorithm $\mathcal{A}$ followed by a subsequent algorithm $\mathcal{B}$. Specifically, $\mathcal{A}' = \mathcal{B}\ \mathrm{QAA}^{\mathcal{A}}$ such that

$$\mathcal{A}'\,|0\rangle = \mathcal{B}\ \mathrm{QAA}^{\mathcal{A}}\,|0\rangle = \sqrt{p'}\,|\psi_G\rangle\,|1\rangle + \sqrt{1-p'}\,|\psi_B\rangle\,|0\rangle.$$

Then, the complexity of QAA using $\mathcal{A}'$ is as follows:

$$\mathcal{C}_{\mathrm{QAA}}(\mathcal{A}') = \frac{\pi}{2\sqrt{p'}} \cdot \mathcal{C}(\mathcal{A}') = \frac{\pi}{2\sqrt{p'}} \cdot \left(\frac{\pi}{2\sqrt{p}} \cdot \mathcal{C}(\mathcal{A}) + \mathcal{C}(\mathcal{B})\right). \tag{2}$$

A similar approach can be used to derive the complexity for more complex structures.

2.5 Quantum state preparation

A sampling problem involves drawing samples from a specified probability distribution. This challenge can be addressed by generating a quantum system that encodes the entire probability distribution in its amplitudes. According to the fundamental principles of quantum computing, measuring this quantum system will yield results following the desired probability distribution. In this context, quantum state preparation, which refers to methods for generating arbitrary quantum states, has been studied.

In this paper, we are interested in preparing the following quantum state:

$$|\psi\rangle = \sum_{i=0,1,\ldots,|S|-1} \frac{1}{\sqrt{|S|}} |d_i\rangle \tag{3}$$

where $S = \{d_0, d_1, \ldots\}$ is not the complete set of n-bit strings, making it non-trivial to generate this quantum state using a few Hadamard gates. In this subsection, we introduce a method for achieving this. The method employs a gate called Quantum Read-Only Memory (QROM), which can be implemented using Toffoli gates, CNOT gates, and a few ancilla qubits [BGB+18]. Additionally, we introduce another method based on single-qubit rotation gates in Appendix A.

QROM is a quantum gate used to access classical data by taking indices from a quantum register and generating the corresponding data qubit register in a superposition state. Unlike Quantum Random Access Memory (qRAM), which allows for both read and write operations on quantum data, QROM is limited to reading fixed classical data without any modifications. We denote the QROM gate as $\mathcal{U}_{\mathrm{QROM}}$ and it performs the following transformation:

$$\mathcal{U}_{\mathrm{QROM}} \sum a_i |i\rangle |0\rangle = \sum a_i |i\rangle |d_i\rangle$$

where d_i are classical data.

With the QROM, we can generate the desired quantum state as follows. Let $m = \lceil \log_2(|S|) \rceil$. First, we construct QROM with an m-qubit index register and an n-qubit data register, where the elements of S are assigned to the d. Then,

$$\mathcal{U}_{\mathrm{QROM}}(H^{\otimes m} |0\rangle_m) \otimes |0\rangle_n = \sum_{i=0}^{2^m-1} \frac{1}{2^{m/2}} |i\rangle |d_i\rangle.$$

Although the result includes an index register and the number of superposed states is a power of two, differing from Eq. (3), it is good enough for use in our algorithm discussed in Section 4.

Now, we describe how to implement QROM using Toffoli gates, CNOT gates, and a few ancilla qubits, following the method presented in [BGB+18]. The overall strategy is as follows: Given an index idx, for each i from 0 to $2^m - 1$, if idx equals i, then XOR d_i with the data register.

The implementation is divided into two parts: "iterating from 0 to $2^m - 1$" and "XORing the data". We denote the gate that iterates over the m-bit index as IDX_m and the gate that XORs the corresponding data as DAT_d. The DAT_d gate can be easily implemented by placing CNOT gates at the positions where the bits of d_i are 1. As an example, QROM for $m = 1$ and $m = 3$ are shown in Figure 2.

We first describe the method for constructing C-IDX_m(Controlled-IDX_m) gate. This means the IDX_m gate iterates over the indices only when the value of the controlled qubit is 1, and does nothing when the value is 0. We construct C-IDX_m gate by using two C-IDX_{m-1} gates. Note that iterating from 0 to $2^m - 1$ corresponds to iterating from 0 to $2^{m-1} - 1$ twice.

Let idx be an index input to C-IDX_m. We set the controlled qubit of the first C-IDX_{m-1} gate to 1 if and only if the msb of idx is 0, and set the controlled qubit of the second C-IDX_{m-1} gate to 1 if and only if the msb is 1. More details are shown in Figure 3. IDX_m gate can be implemented by removing the controlled qubit from the C-IDX_m gate. Finally, the QROM gate can be constructed by combining the IDX_m gate and the DAT_d gate.

The C-IDX_m gate is composed of two C-IDX_{m-1} gates, two Toffoli gates, and one CNOT gate. The C-IDX_1 gate consists of two Toffoli gates and one CNOT gate. Consequently, a C-IDX_m gate requires $2^{m+1} - 2$ Toffoli gates and $2^m - 1$ CNOT gates. Therefore, IDX_m gate can be constructed using $2^{m+1} - 4$ Toffoli gates, 2^m CNOT gates, and one X gate. To

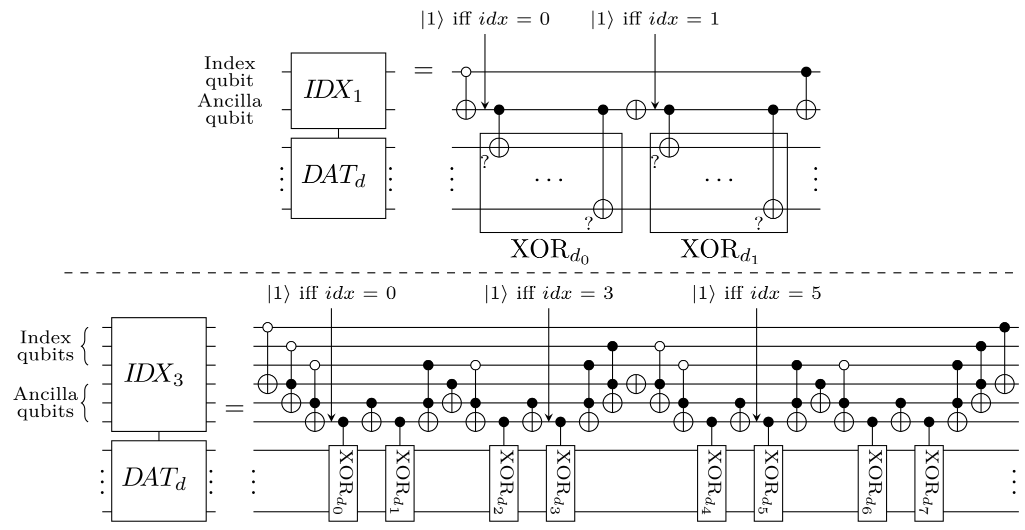

Figure 2: QROM gate implementation for m=1 and m=3.

construct a DAT_d gate, a maximum of $2^m n$ CNOT gates is needed. Therefore, the cost of a QROM with an m-bit index and n-bit data is $2^{m+1} - 4$ Toffoli gates, $2^m(n+1)$ CNOT gates, and one X gate. Note that quantum state preparation using QROM requires $2m$ ancilla qubits (qubits for IDX_m gate).

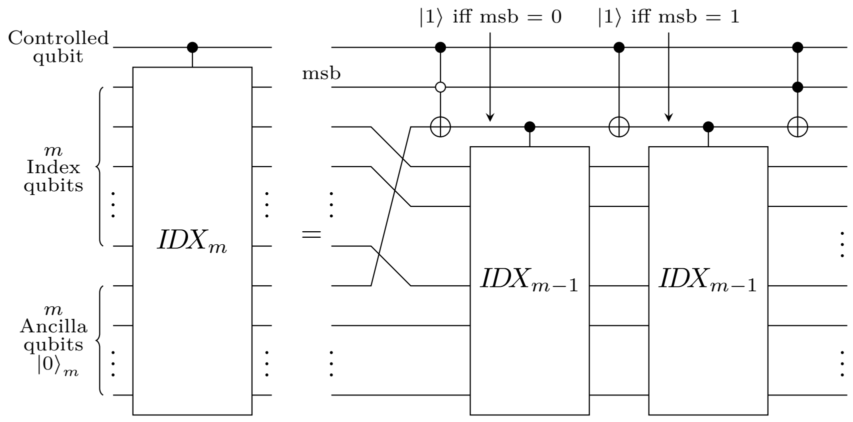

Figure 3: Recursive implementation method for the C-IDX_m gate.

3 Identifying Flaws in Previous Works and Refining Complexity Evaluation

In this section, we first outline the general classical and quantum rebound attacks. Subsequently, we describe the quantum rebound attacks presented in [CKS21, BK23]. Finally, we discuss the flaws in these attacks and propose the correct complexities.

3.1 General Overview of classical and quantum rebound attack

The rebound attack, initially proposed by Mendel et al., is a cryptanalysis technique used for analyzing hash functions [MRST09]. In this study, we focus on the quantum rebound attack on Hirose's Double Block Length Compression Function (HCF). To begin, we introduce a key property of HCF that will be leveraged in the subsequent rebound attack analysis.

Proposition 1. *Let* HCF-E *be an* HCF *instantiated with* E*, and denote* $(g_i, h_i) =$ HCF-E (g_{i-1}, h_{i-1}, m_i) *and* $(g'_i, h'_i) =$ HCF-$E(g'_{i-1}, h_{i-1}, m_i)$. *If* $g_i = g'_i$ *and* $g_{i-1} \oplus g'_{i-1} = c$*, then* $h_i = h'_i$. *Especially,* (g_{i-1}, h_{i-1}, m_i) *and* (g'_{i-1}, h_{i-1}, m_i) *are a collision of* HCF-E.

Proof. Since $g_i = g'_i$,

$$E_{h_{i-1}||m_i}(g_{i-1}) \oplus g_{i-1} = E_{h_{i-1}||m_i}(g'_{i-1}) \oplus g'_{i-1}.$$

From $g'_{i-1} = g_{i-1} \oplus c$,

$$E_{h_{i-1}||m_i}(g_{i-1}) \oplus g_{i-1} = E_{h_{i-1}||m_i}(g_{i-1} \oplus c) \oplus g_{i-1} \oplus c. \tag{4}$$

In Eq. (4), the left-hand side is equal to h'_i, while the right-hand side is equal to h_i, leading to the conclusion that $h'_i = h_i$. □

According to Proposition 1, the task of finding collisions for HCF can be reduced to finding pairs for E where the corresponding plaintext and ciphertext differences are equal to c. This is because $g_i = g'_i$ and $g_{i-1} \oplus g'_{i-1} = c$ is equivalent to $g_{i-1} \oplus g'_{i-1} = c$ and $E_{h_{i-1}||m_i}(g_{i-1}) \oplus E_{h_{i-1}||m_i}(g_{i-1}) = c$.

The classical rebound attack serves as a method to find such pairs. Given a truncated differential trail for E, we partition E into three subparts, denoted as $E = E_{fw} \circ E_{in} \circ E_{bw}$. A rebound attack consists of an inbound phase and an outbound phase, and these phases are repeated until a collision is found. The overall structure of the rebound attack is illustrated in Figure 4, and each phase is briefly described as follows:

- Inbound Phase: Given an input-output difference $(\Delta_{in}, \Delta_{out})$ for E_{in}, find pairs (M, M') that follow the differential trail $\Delta_{in} \to \Delta_{out}$. We refer to such pairs as the starting points. To find the starting points, several S-box differential equations must be solved. If there are no solutions, the Inbound Phase is repeated with a different $(\Delta_{in}, \Delta_{out})$; if solutions exist, we derive all starting points from the solutions and perform the Outbound Phase for each of them.

- Outbound Phase: Propagate (M, M') both forward and backward and check whether the plaintext and ciphertext differences are equal to c.

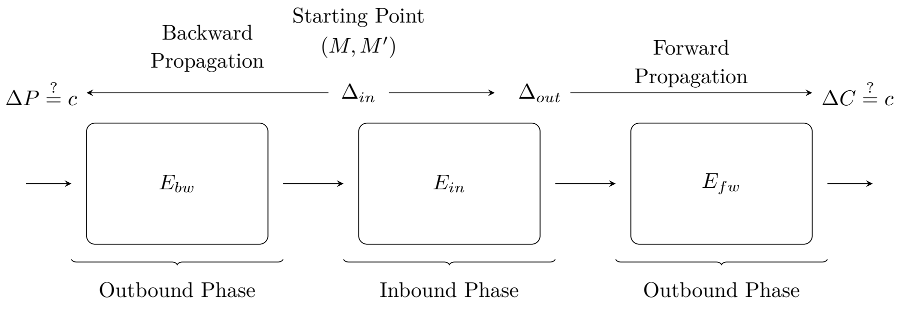

Figure 4: An overall structure of the rebound attack.

A quantum rebound attack is conducted as follows. Let F be a Boolean function such that $F(\Delta_{in}, \Delta_{out}) = 1$ if and only if $(\Delta_{in}, \Delta_{out})$ satisfy the conditions for both the inbound and outbound phases. The problem is now reduced to finding $(\Delta_{in}, \Delta_{out})$ such that $F(\Delta_{in}, \Delta_{out}) = 1$. Therefore, if we can construct a quantum gate $\mathcal{U}_F$ such that

$$\mathcal{U}_F \left|\Delta_{in}, \Delta_{out}\right\rangle \left|y\right\rangle = \left|\Delta_{in}, \Delta_{out}\right\rangle \left|y \oplus F(\Delta_{in}, \Delta_{out})\right\rangle,$$

we can find collisions using Grover's search or QAA.

However, in the Inbound Phase, performing different operations based on the existence of solutions to the S-box differential equations is straightforward in classical computing but not in quantum computing. In [HS20, DSS⁺20], this issue was addressed by adding auxiliary bits to the function F. Specifically, they modified the function to $F(\Delta_{in}, \Delta_{out}, \alpha) = 1$ if and only if the α-th solution corresponding to $(\Delta_{in}, \Delta_{out})$ satisfies all the conditions of the inbound and outbound phases. Each bit of α helps determine which solution to choose for each S-box differential equation when solutions exist. The $\mathcal{U}_F$ gate is also modified to include α as an additional input, such that

$$\mathcal{U}_F \left|\Delta_{in}, \Delta_{out}, \alpha\right\rangle \left|y\right\rangle = \left|\Delta_{in}, \Delta_{out}, \alpha\right\rangle \left|y \oplus F(\Delta_{in}, \Delta_{out}, \alpha)\right\rangle. \tag{5}$$

Let the probability that a starting point satisfies the conditions of the outbound phase be $p = 2^{-n}$. Then, to find a collision, we need to set the size of the search space for $(\Delta_{in}, \Delta_{out})$ as 2^n. Let m be the number of S-box differential equations solved in the inbound phase. Then, the length of the additional input α is m bits. Given these parameters, the complexity of the quantum rebound attack equals $\mathcal{C}_{\mathrm{QAA}}(\mathcal{U}_F)$, and

$$\mathcal{C}_{\mathrm{QAA}}(\mathcal{U}_F) = \frac{\pi}{2} \cdot 2^{(n+m)/2} \cdot \mathcal{C}(\mathcal{U}_F). \tag{6}$$

3.2 Quantum rebound attack on HCF-AES-256

We present a description of the quantum rebound attack on `HCF-AES-256` as proposed in [CKS21]. Subsequently, we identify flaws and provide a corrected complexity evaluation.

3.2.1 Attack procedure presented in [CKS21]

The quantum rebound attack presented in [CKS21] is built upon the 10-round truncated differential trail depicted in Figure 5. This trail is constructed with two inbound phases (inbound phases 1 and 2) and two outbound phases. The initial step of the attack involves finding two pairs following each inbound phase. Subsequently, the attack determines internal values (states, round keys) that make two pairs compatible. Then, the attack proceeds to the outbound phase to check whether the starting point[2] results in a collision.

The probability that a starting point satisfies the conditions of the outbound phase is 2^{-160}. This is because the probabilities of following backward and forward propagation (particularly differential transition through MC in Rounds 2 and 8, as observed in Figure 5) are each 2^{-16}, and the probabilities of satisfying $\Delta X_0 = c$ and $\Delta W_{10} = c$ are each 2^{-64}. Consequently, to find a collision, 2^{160} starting points are required.

[CKS21] define a Boolean function F that takes $(\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out})$ where $(\Delta^1_{in}, \Delta^1_{out})$ and $(\Delta^2_{in},\ \Delta^2_{out})$ represents the input-output difference for inbound phases 1 and 2, respectively, and outputs 1 iff $(\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out})$ produce a collision; otherwise, it outputs 0. The authors assigned the search space corresponding to $\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out}$ as $\mathbb{F}_2^{32}$, $\mathbb{F}_2^{48}$, $\mathbb{F}_2^{48}$, and $\mathbb{F}_2^{32}$, respectively, to find 2^{160} starting points. The following is the process of computing F on a classical computer as presented in [CKS21]:

Step 1. Compute $(\Delta X_4, \Delta Y_4)$ from $(\Delta Z_3, \Delta W_4) = (\Delta^1_{in}, \Delta^1_{out})$.

[2]Here, the starting point refers to pairs connected by internal values in the inbound phase.

Step 2. Solve 16 differential equations between $(\Delta X_4, \Delta Y_4)$ to find X_4:

For $0 \le i \le 15$,

- Find one $X_4[i]$ such that

$$S(X_4[i]) \oplus S(X_4[i] \oplus \Delta X_4[i]) = \Delta Y_4[i].$$

- Set $X_4[i] \leftarrow min\,\{X_4[i], X_4[i] \oplus \Delta X_4[i]\}$.
- If there are no admissible values for $X_4[i]$, then return to Step 1.

Step 3. Compute $(\Delta X_7, \Delta Y_7)$ from $(\Delta Z_6, \Delta W_7) = (\Delta^2_{in}, \Delta^2_{out})$.

Step 4. Similar to Step 2, solve 16 differential equations between $(\Delta X_7, \Delta Y_7)$ to find X_7. If there are no admissible values for X_7, then return to Step 3.

Step 5. Find the internal values that make X_4 and X_7 compatible.

- Find X_5, X_6, K_4, K_5, and K_6 such that:
  - one-round encryption of X_4 with the round key K_4 equals to X_5,
  - one-round encryption of X_5 with the round key K_5 equals to X_6,
  - one-round encryption of X_6 with the round key K_6 equals to X_7.

Step 6. From K_4, K_5, and K_6, find round keys K_0, K_1, K_2, K_3, K_7, K_8, K_9, and K_{10}.

Step 7. Calculate ΔX_0 from $(X_4, X_4 \oplus \Delta X_4)$ and ΔW_{10} from $(X_7, X_7 \oplus \Delta X_7)$ using the round keys. If $\Delta X_0 = \Delta W_{10} = c$, then outputs 1 and outputs 0, otherwise.

We omit the intricate details of the process connecting X_4 and X_7 in Step 5, as it lies outside the primary focus of this paper. [CKS21] then translated each step into its quantum counterpart to construct the quantum circuit $\mathcal{U}_F$. We have verified that there are no flaws in the process of translating to the quantum counterpart or in deriving the cost of the U_F gate, but since the detailed internal process is not crucial to our paper, we do not include it here. For an in-depth understanding, readers are directed to [CKS21].

[CKS21] suggests that

$$\mathcal{C}(\mathcal{U}_F) = \begin{cases} 2^{5.46} & \text{in Q-I when } 2^{16} \text{ qRAM is available,} \\ 2^{8.96} & \text{and requires } S \ge 2^4 \text{ in Q-II,} \\ 2^{6.42} & \text{in Q-III,} \end{cases}$$

and evaluates the complexity of the attack as $\frac{\pi}{4} \cdot \sqrt{2^{160}} \cdot \mathcal{C}(\mathcal{U}_F)$. The overall complexities they proposed are summarized in Table 1.

3.2.2 Identified flaws and corrected complexity evaluation

The first flaw is in the process of returning to Step 1 and Step 3 when there are no admissible values for $X_4[i]$ in Step 2 and $X_7[i]$ in Step 4, respectively. In Step 1, given $(\Delta^1_{in}, \Delta^1_{out})$, $(\Delta X_4, \Delta Y_4)$ is uniquely determined. Therefore, even if Step 1 is repeated, the result of Step 2 remains unchanged. Consequently, if there are no admissible values for $X_4[i]$ or $X_7[i]$, it is appropriate to directly output 0.

The second and more critical flaw is that the proposed attack can only gather 2^{128} starting points instead of the intended 2^{160}. This discrepancy arises because the probabilities that solutions exist in Step 2 and Step 4 are both 2^{-16}. Therefore, to obtain 2^{160} starting points, consideration must be given to all possible X_4 and X_7 instead of only considering $min(X_4^{(i)}, X_4^{(i)} \oplus \Delta X_4^{(i)})$ and $min(X_7^{(i)}, X_7^{(i)} \oplus \Delta X_7^{(i)})$ for each byte.

In this context, we correct the procedure for computing function F as follows:

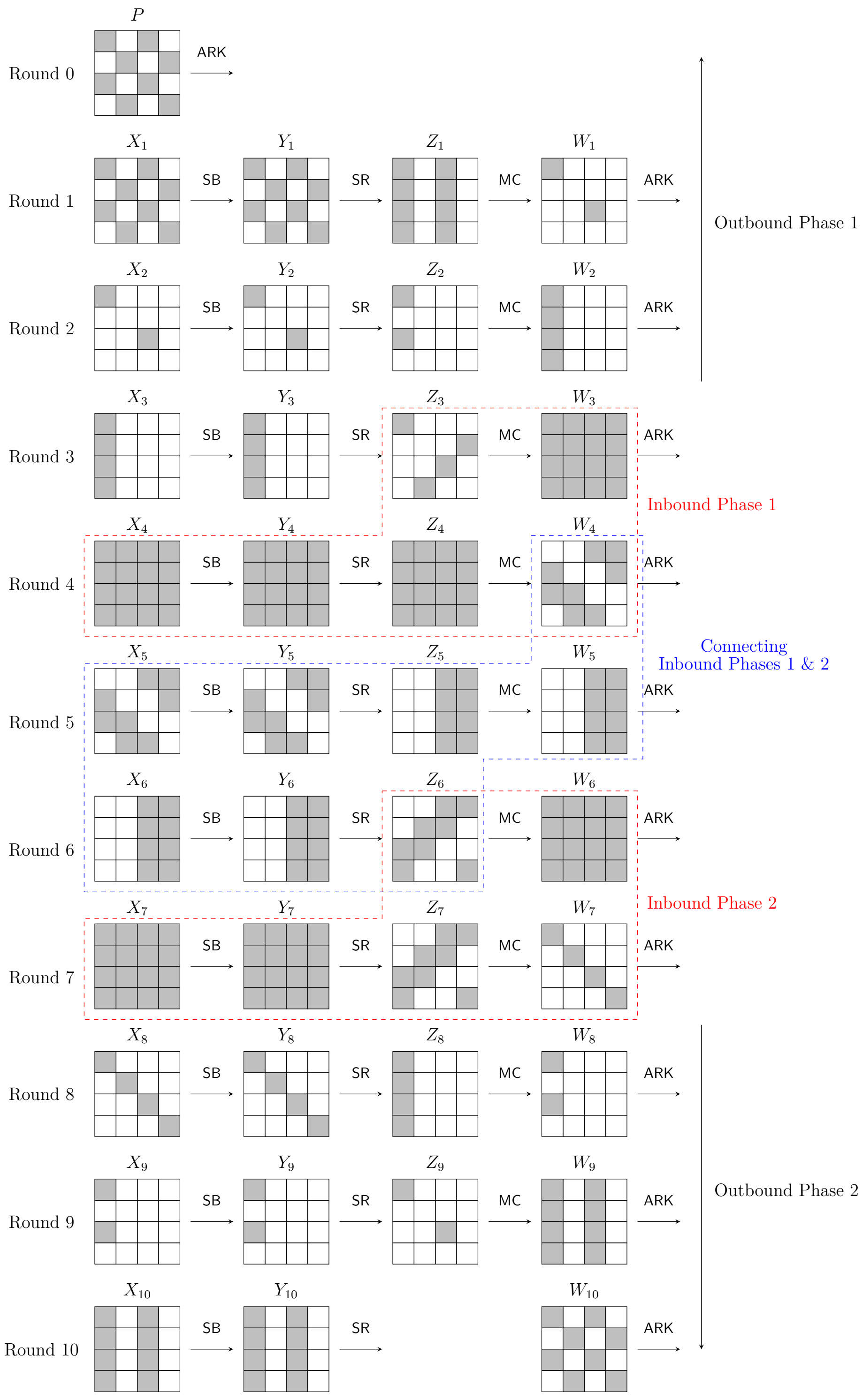

Figure 5: The differential trail for 10-round `AES-256` presented in [CKS21].

Step 1′. Given $(\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out})$, compute $(\Delta X_4, \Delta Y_4)$ and $(\Delta X_7, \Delta Y_7)$ as in Steps 1 and 3.

Step 2′. Check whether all 32 S-box differential equations between ΔX_4 and ΔY_4 and ΔX_7 and ΔY_7 have solutions. If not, output 0. If all equations have solutions, repeat Step 3' approximately 2^{32} times for each pair (X_4, X_7).

Step 3′. Given (X_4, X_7), perform Steps 5, 6, and 7, and output 1 if and only if a collision is found.

In the classical setting, this issue does not impact the complexity, as it is possible to skip Step 3′ when there is an equation that has no solution in Step 2′. However, in the quantum setting, both skipping Step 3′ when there is no solution and repeating Step 3′ only when all 32 equations have solutions are non-trivial.

As discussed in Section 3.1, we can resolve the issue by adding 32 auxiliary bits to the search space. Each of these 32 auxiliary bits helps determine which of the (typically) two solutions to select for each S-box differential equation in Step 2′. Specifically, let us assume that $X_4[0]$ and $X'_4[0]$ are the solutions to the S-box differential equations between $\Delta X_4[0]$ and $\Delta Y_4[0]$. In this case, if the auxiliary bit is 0, we choose $\min\{X_4[0], X'_4[0]\}$, and if the auxiliary bit is 1, we choose $\max\{X_4[0], X'_4[0]\}$.

There are also minor issues to consider. First, the number of solutions for an S-box differential equation is not always two. There is a $\frac{1}{128}$ probability of having four solutions, but the attack considers only two of them. Consequently, the success probability of the attack is $(\frac{127}{128})^{32} \approx 2^{-0.36}$. Second, when calculating the complexity of QAA using $\mathcal{U}_F$, each iteration calls $\mathcal{U}_F$ twice, so the constant factor is $\pi/2$ instead of $\pi/4$. Please refer to Eq. 1 for more details.

In conclusion, for the successful execution of the attack, 32 auxiliary bits are required, and the attack needs to be repeated $2^{0.36}$ times. This results in the correct complexity of the attack being $2^{17.36}$ times larger than the values proposed in [CKS21]. As a result, the attack becomes less efficient than the `BHT` algorithm when large qRAM is available in the Q-I setting and slightly less efficient than the `CNS` algorithm in the Q-III setting, as shown in Table 1.

3.3 Quantum rebound attack on HCF-ARIA-256

We present a description of the quantum rebound attack on `HCF-ARIA-256` as proposed in [BK23]. Subsequently, we identify flaws and provide a corrected complexity evaluation.

3.3.1 Attack Procedure presented in [BK23]

The quantum rebound attack presented in [BK23] is based on the 7-round truncated trail illustrated in Figure 6. This trail comprises two inbound phases (inbound phases 1 and 2) and two outbound phases, and the attack adopts the same strategy as presented in [CKS21].

In backward propagation, to achieve the desired differential transition between the DL in Round 1, all bytes in $\Delta Z_1[3,4,6,8,9,13,14] = \Delta X_2[3,4,6,8,9,13,14]$ must be the same. Given $\Delta Y_2[3,4,6,8,9,13,14]$, the probability of all bytes in $\Delta X_2[3,4,6,8,9,13,14]$ being identical is 2^{-48}. Similarly, the probability of successful forward propagation is also 2^{-48}. The probabilities of $\Delta X_0 = c$ and $\Delta X_8 = c$ are each 2^{-8}; therefore, the overall probability that a starting point satisfies the conditions for the outbound phase is 2^{-112}. In other words, to find a collision, 2^{112} starting points are needed.

[BK23] defines a Boolean function F that takes $(\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out})$ where $(\Delta^1_{in}, \Delta^1_{out})$ and $(\Delta^2_{in}, \Delta^2_{out})$ represents the input-output difference for inbound phases 1 and 2, respectively, and outputs 1 iff $(\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out})$ produce a collision; otherwise, it outputs 0. The authors assigned the search space corresponding to $\Delta^1_{in}, \Delta^2_{in}, \Delta^1_{out}, \Delta^2_{out}$ as $\mathbb{F}_2^{28}$, $\mathbb{F}_2^{28}$, $\mathbb{F}_2^{28}$, and $\mathbb{F}_2^{28}$, respectively, to find 2^{160} starting points. The following is the process of computing F on a classical computer as presented in [BK23]:

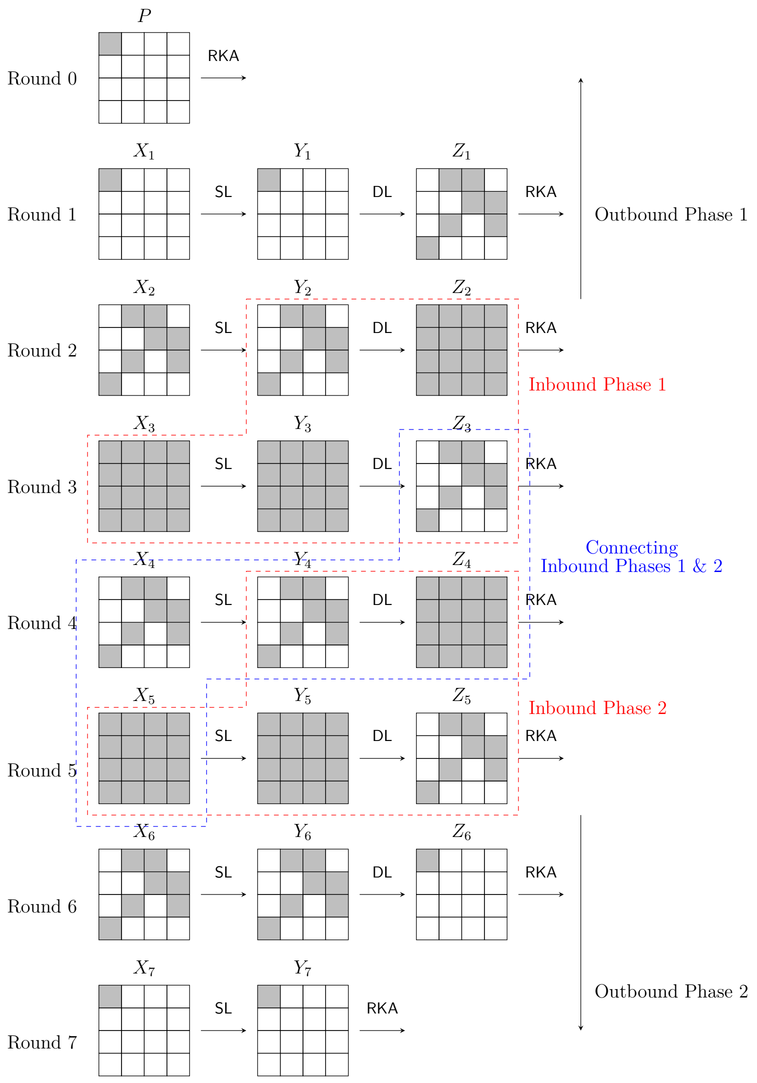

Figure 6: The differential trail for 7-round `ARIA-256` presented in [BK23].

Step 1. Choose a 9-byte random value to use later.

Step 2. Compute $(\Delta X_3, \Delta Y_3)$ from $(\Delta Y_2, \Delta Z_3) = (\Delta^1_{in}, \Delta^1_{out})$.

Step 3. Solve 16 differential equations between $(\Delta X_3, \Delta Y_3)$ to find X_3:

For $0 \le i \le 15$,

- Find one $X_3[i]$ such that

$$S(X_3[i]) \oplus S(X_3[i] \oplus \Delta X_3[i]) = \Delta Y_3[i].$$

- Set $X_3[i] \leftarrow min\{X_3[i], X_3[i] \oplus \Delta X_3[i]\}$.
- If there are no admissible values for $X_3[i]$, then return to Step 2.

Step 4. Compute $(\Delta X_5, \Delta Y_5)$ from $(\Delta Y_4, \Delta Z_5) = (\Delta^2_{in}, \Delta^2_{out})$.

Step 5. Similar to Step 2, solve 16 differential equations between $(\Delta X_5, \Delta Y_5)$ to find X_5. If there are no admissible values for X_5, then return to Step 4.

Step 6. Compute ΔX_4 from $\Delta Z_3 = \Delta^1_{out}$.

Step 7. Solve 7 S-box differential equations between (ΔX_4[3, 4, 6, 8, 9, 13, 14], ΔY_4[3, 4, 6, 8, 9, 13, 14]) to find X_4[3, 4, 6, 8, 9, 13, 14]. Assign values for X_4[0, 1, 2, 5, 7, 10, 11, 12, 15] with nine bytes selected in Step 1 to determine the full X_4.

Step 8. Find internal values that make X_3, X_4, and X_5 compatible.

- Find K_3 and K_4 from X_3, X_4, and X_5.
- Find the master key compatible with K_3 and K_4.

Step 9. From the master key, find the round keys K_0, K_1, K_2, K_5, K_6, and K_7.

Step 10. Calculate ΔX_0 from $(X_3, X_3 \oplus \Delta X_3)$, and ΔX_8 from $(X_5, X_5 \oplus \Delta X_5)$ using round keys. If $\Delta X_0 = \Delta X_8 = c$, then outputs 1 and outputs 0, otherwise.

We omit the detailed description of the process for finding the master key in Step 8, as it lies outside the primary focus of this paper. Also, we do not include how to construct the $\mathcal{U}_F$ gate and evaluate its cost here for the same reason as described in Section 3.2.1.

[BK23] suggests that $\mathcal{C}(\mathcal{U}_F) = 2^{62.91}$ and requires $S \geq 2^{2.54}$ in the Q-II setting, and evaluates the complexity of the attack as $\frac{\pi}{4} \cdot \sqrt{2^{112}} \cdot \mathcal{C}(\mathcal{U}_F)$. The overall complexities they proposed are summarized in Table 1.

3.3.2 Identified flaws and corrected complexity evaluation

The quantum rebound attack on `HCF-ARIA-256` presented in [BK23] also includes all the flaws discussed in Section 3.2.2, and the methods to resolve them are identical. Since the attack process involves solving 39 S-box differential equations (16 in Step 3, 16 in Step 5, and 7 in Step 7), we can correct the main flaw by adding 39 auxiliary bits to the search space. Additionally, we should consider that the success probability of the attack is $\left(\frac{127}{128}\right)^{39} \approx 2^{-0.44}$ and that the constant factor must be adjusted from $\pi/4$ to $\pi/2$.

In conclusion, for the successful execution of the attack, 39 auxiliary bits are required, and the attack needs to be repeated $2^{0.44}$ times. This results in the correct complexity of the attack being $2^{20.94}$ times larger than the values proposed in [BK23]. Consequently, the attack becomes less efficient than the parallel rho algorithm in the Q-II setting, as shown in Table 1.

4 Improving quantum rebound attacks by nested QAA and quantum state preparation

In this section, we introduce improved quantum rebound attacks using nested QAA and quantum state preparation. First, we describe the general structure of our improved attack. Then we apply the method to previous quantum rebound attacks on `HCF-AES-256` [CKS21] and HCF-ARIA-256 [BK23] to achieve a reduction in time complexity.

4.1 Overall Structure of Improved Quantum Rebound Attacks

Recall the parameters defined in Section 3.1. The probability that a starting point satisfies the conditions of the outbound phase is $p = 2^{-n}$, and the number of S-box differential equations solved in the inbound phase is m, which makes α an m-bit value. F is a Boolean function that takes $(\Delta_{in}, \Delta_{out})$ and auxiliary bits α and outputs 1 if and only if a collision is derived from the α-th solution corresponding to $(\Delta_{in}, \Delta_{out})$. $\mathcal{U}_F$ is a quantum gate that represents the function F.

4.1.1 Quantum gates for our improved attacks

We introduce two quantum gates for our improved quantum rebound attack.

Search space pre-filtering gate Suppose that we can pre-filter the search space for $(\Delta_{in}, \Delta_{out})$ efficiently by a factor of 2^k, and S is the reduced search space. Note that $|S| = 2^{n-k}$. We construct a quantum gate $\mathcal{U}_{\mathrm{PF}}$ that generates the superposition state over S, based on the quantum state preparation method presented in Section 2.5. Specifically,

$$\mathcal{U}_{\mathrm{PF}} \left|0\right\rangle_n = \frac{1}{\sqrt{|S|}} \sum_{i \in S} \left|i\right\rangle.$$

Solution existence-checking gate We construct a gate $\mathcal{U}_{\mathrm{SC}}$ to check for the existence of solutions to the m S-box differential equations in the inbound phase. Specifically,

$$\mathcal{U}_{\mathrm{SC}} \left|\Delta_{in}, \Delta_{out}\right\rangle \left|0\right\rangle = \left|\Delta_{in}, \Delta_{out}\right\rangle \left|y\right\rangle,$$

where $y = 1$ if and only if all m S-box differential equations in the inbound phase have solutions.

4.1.2 Attack procedure

Our improved quantum rebound attacks start with the QAA with the quantum algorithm $\mathcal{A}$ such that

$$\mathcal{A} = \mathcal{U}_{\mathrm{SC}}\, \mathcal{U}_{\mathrm{PF}},$$
$$\mathcal{A} \left|0\right\rangle_n \left|0\right\rangle = \sum_{\substack{\text{solution}\\ \text{exist}}} \left|\Delta_{in}, \Delta_{out}\right\rangle \left|1\right\rangle + \sum_{\substack{\text{solution}\\ \text{not exist}}} \left|\Delta_{in}, \Delta_{out}\right\rangle \left|0\right\rangle.$$

Since $|S| = 2^{n-k}$ and S contains 2^{n-m} $(\Delta_{in}, \Delta_{out})$ that make all m S-box differential equations have solutions in the inbound phase, the amplitude of the good state is $\sqrt{2^{k-m}}$. Then,

$$\mathcal{C}_{\mathrm{QAA}}(\mathcal{A}) = \frac{\pi}{2} \cdot 2^{(m-k)/2} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{\mathrm{SC}})),$$

and the result quantum state is the superposition of the 2^{n-m} state, that is

$$\sum_{\substack{\text{solution}\\ \text{exist}}} \left|\Delta_{in}, \Delta_{out}\right\rangle.$$

Now, let us consider a quantum algorithm $\mathcal{A}'$. This algorithm concatenates a uniform superposition of m-bit for α to the result of QAA using $\mathcal{A}$, and then applies the $\mathcal{U}_F$ gate (as defined in Eq. (5)). Specifically,

$$\begin{aligned}\mathcal{A}' \left|0\right\rangle_n \left|0\right\rangle_m \left|0\right\rangle &= \mathcal{U}_F(\mathrm{QAA}^{\mathcal{A}} \left|0\right\rangle_n \otimes H^{\otimes m} \left|0\right\rangle_m) \left|0\right\rangle \\ &= \sum_{\substack{\text{solution}\\ \text{exist}}} \sum_{\alpha=0}^{2^m-1} \left|\Delta_{in}, \Delta_{out}, \alpha\right\rangle \left|F(\Delta_{in}, \Delta_{out}, \alpha)\right\rangle.\end{aligned}$$

Since the probability that a collision is derived from a starting point (which corresponds to *one* solution) is $p = 2^{-n}$, the amplitude of the good state is $\sqrt{2^{-n}}$. Therefore, from Eq. (2), the complexity of the attack is

$$\mathcal{C}_{\mathrm{QAA}}(\mathcal{A}') = \frac{\pi}{2} \cdot 2^{n/2} \cdot \left(\frac{\pi}{2} \cdot 2^{(m-k)/2} \cdot \left(\mathcal{C}\left(\mathcal{U}_{\mathrm{PF}}\right) + \mathcal{C}\left(\mathcal{U}_{\mathrm{SC}}\right)\right) + \mathcal{C}(\mathcal{U}_F)\right). \quad (7)$$

4.1.3 Complexity Analysis

If $\frac{\pi}{2} \cdot 2^{(m-k)/2} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{\mathrm{SC}}))$ is a dominant factor (it is the case of the attack on HCF-AES-256, Section 4.2), the complexity of the attack becomes

$$\frac{\pi^2}{4} \cdot 2^{(n+m-k)/2} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{\mathrm{SC}})).$$

Compared to Eq (6), the constant factor is reduced by a factor of about $2^{k/2}$, and $\mathcal{C}(\mathcal{U}_{\mathrm{SC}})$ is smaller than $\mathcal{C}(\mathcal{U}_F)$.[3] Therefore, if we can implement $\mathcal{U}_{\mathrm{PF}}$ efficiently, we can reduce the complexity of the attack.

If $\mathcal{C}(\mathcal{U}_F)$ is a dominant factor (it is the case of the attack on HCF-ARIA-256, Section 4.3), the complexity of the attack becomes

$$\frac{\pi}{2} \cdot 2^{n/2} \cdot \mathcal{C}(\mathcal{U}_F),$$

which is reduced by a factor of $2^{m/2}$ compared to Eq (6).

4.2 Improved quantum rebound attack on HCF-AES-256

In line with previous works [HS20, DSS+20, CKS21], we analyze the complexity based on the number of S-box computations. Specifically, we approximate the complexity of a 10-round AES computation as equivalent to 200 S-box computations.[4] Additionally, following recent research [LXX+23], we approximate the cost of one AES S-box as equivalent to 50 Toffoli gates.

4.2.1 Constructing $\mathcal{U}_{\mathrm{PF}}$ and $\mathcal{U}_{\mathrm{SC}}$

Recall that the quantum rebound attack on `HCF-AES-256` required 2^{160} starting points. Since one starting point is obtained on average given $(\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}^1, \Delta_{out}^2)$, we need to set the size of the search space to 160 bits. [CKS21] set the search space for Δ_{in}^1, Δ_{in}^2, Δ_{out}^1, and Δ_{out}^2 as $\mathbb{F}_2^{32}$, $\mathbb{F}_2^{48}$, $\mathbb{F}_2^{48}$, and $\mathbb{F}_2^{32}$, respectively. However, there is no specific mention of how the search space was constructed for Δ_{out}^1 and Δ_{in}^2, although each has 8 active bytes.

We observe that by fixing the values of certain 2 bytes among the 8 active bytes in Δ_{out}^1 and Δ_{in}^2, we can pre-filter the impossible values for Δ_{in}^1 and Δ_{out}^2. Specifically, for Δ_{out}^1, fixing the values of $\Delta_{out}^1[12, 13]$ allows us to pre-filter the impossible values for each byte in Δ_{in}^1 as follows:

1. Fixing $\Delta_{out}^1[12, 13]$ also fixes $\Delta Z_4[12-15]$ and $\Delta Y_4[1, 6, 11, 12]$.
2. With $\Delta Y_4[1, 6, 11, 12]$ fixed, each byte of $\Delta X_4[1, 6, 11, 12](= \Delta W_3[1, 6, 11, 12])$ can only assume 127 values out of 256.
3. As $\Delta W_3[1, 6, 11, 12]$ is limited to 127 values, $\Delta Z_3[0, 7, 10, 13]$ is restricted to 127 values.

In Inbound Phase 2, using a similar approach, fixing two bytes $\Delta_{in}^2[2, 3]$ allows us to pre-filter the impossible values for each byte in Δ_{out}^2. Consequently, we can reduce the overall search space by a factor of 2^8.

Now, we can construct the pre-filtering gate $\mathcal{U}_{\mathrm{PF}}$ as illustrated in Figure 8. In the Q-I setting, state preparation can be implemented using qRAM. By utilizing a qRAM of size 2^{16}, state preparation for 2 bytes can be achieved with a single query to qRAM.[5] Therefore, we can construct $\mathcal{U}_{\mathrm{PF}}$ using 4 qRAM queries. In the Q-I setting, since S-box

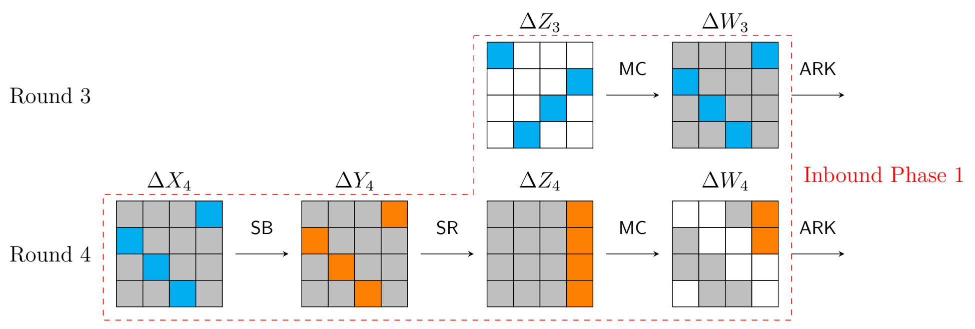

Figure 7: Pre-filtering Δ_{in}^{1} for quantum rebound attack on `HCF-AES-256`: If we fix the value for $\Delta_{out}^{1}[12, 13]$, each byte of Δ_{in}^{1} is limited to 127 possible values. The orange cell indicates the cell whose value is fixed, and the blue cell indicates the cell that can have limited values.

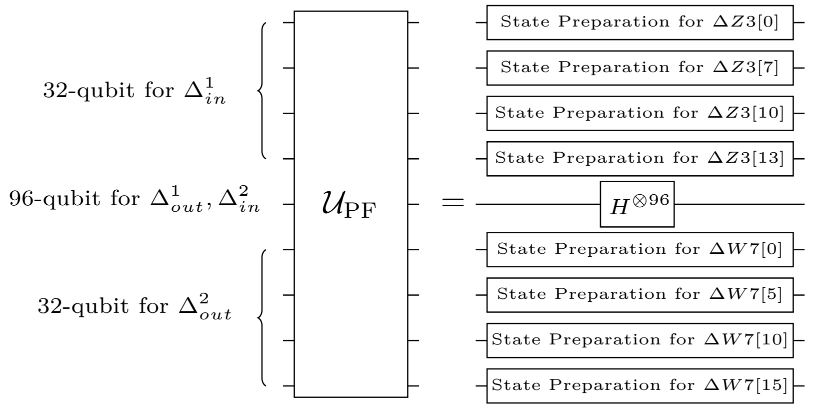
32-qubit for Δ_{in}^{1}
96-qubit for $\Delta_{out}^{1}, \Delta_{in}^{2}$
32-qubit for Δ_{out}^{2}
$\mathcal{U}_{\text{PF}}$
=
State Preparation for $\Delta Z3[0]$
State Preparation for $\Delta Z3[7]$
State Preparation for $\Delta Z3[10]$
State Preparation for $\Delta Z3[13]$
$H^{\otimes 96}$
State Preparation for $\Delta W7[0]$
State Preparation for $\Delta W7[5]$
State Preparation for $\Delta W7[10]$
State Preparation for $\Delta W7[15]$

Figure 8: Pre-filtering gate for quantum rebound attack on `HCF-AES-256`.

computation can also be implemented with one query to qRAM, the complexity of $\mathcal{U}_{\text{PF}}$ is equivalent to 4 S-box computations.

In the Q-II and Q-III settings, we can construct $\mathcal{U}_{\text{PF}}$ using the quantum state preparation described in Section 2.5. The QROM-based state preparation for an 8-qubit system requires 252 Toffoli gates, which we approximate to 5 S-box computations. Since $\mathcal{U}_{\text{PF}}$ includes 8 state preparations, as illustrated in Figure 8, the complexity of $\mathcal{U}_{\text{PF}}$ is equivalent to 40 S-box computations. To summarize,

$$\mathcal{C}(\mathcal{U}_{\text{PF}}) = \begin{cases} 4/200 \approx 2^{-5.64} & \text{in Q-I,} \\ 40/200 \approx 2^{-2.32} & \text{in Q-II and Q-III.} \end{cases}$$

For $\mathcal{U}_{\text{SC}}$, we can check the existence of solutions of an S-box differential equation with one query to qRAM in the Q-I setting. In the Q-II and Q-III settings, we use the dedicated quantum circuit for S-box proposed by Bonnetain et al. [BNS19]. Their dedicated circuit costs 506 Toffoli gates, comprising 422 Toffoli gates for solving the quadratic equation and an additional 84 Toffoli gates for the two multiplications in $\mathbb{F}_2^8$, as recent research [LLYL23] demonstrated that each multiplication in $\mathbb{F}_2^8$ can be implemented using 42 Toffoli gates. We approximate this cost to be equivalent to 12 S-box computations.

[3] F includes the process of checking the existence of solutions.
[4] AES requires 16 and 4 S-box computations for the round function and key schedule for each round.
[5] 2^{16} qRAM is required to solve the S-box differential equations.

Since there are 32 S-box differential equations involved in the inbound phase,

$$\mathcal{C}(\mathcal{U}_{\mathrm{SC}}) = \begin{cases} 32/200 \approx 2^{-2.64} & \text{in Q-I,} \\ 32 \cdot 12/200 \approx 2^{0.94} & \text{in Q-II and Q-III.} \end{cases}$$

4.2.2 Complexity Analysis

From Eq. (7), the complexity of the attack is

$$\begin{aligned} &2^{0.36} \cdot \frac{\pi}{2} \cdot 2^{80} \cdot \left(\frac{\pi}{2} \cdot 2^{(32-8)/2} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{SC})) + \mathcal{C}(\mathcal{U}_F) \right) \\ &= 2^{81.01} \cdot (2^{12.65} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{SC})) + \mathcal{C}(\mathcal{U}_F)), \end{aligned}$$

where $2^{0.36}$ is due to success probability. As we described in 3.2.1, the range of $\mathcal{C}(\mathcal{U}_F)$ is $2^{5.46}$ to $2^{8.96}$ according to the quantum setting, which is negligible in this case. Therefore, the complexity of the attack is

$$\begin{cases} 2^{81.01} \cdot (2^{12.65} \cdot (2^{-5.64} + 2^{-2.64})) = 2^{91.19} & \text{in Q-I,} \\ 2^{81.01} \cdot (2^{12.65} \cdot (2^{-2.32} + 2^{0.94})) = 2^{94.74} & \text{in Q-II and Q-III.} \end{cases}$$

4.3 Improved quantum rebound attack on HCF-ARIA-256

For the quantum rebound attack on HCF-ARIA-256 presented in [BK23], we can pre-filter search space by a factor of 2^{19} as described in Appendix B. And we can construct $\mathcal{U}_{\mathrm{PF}}$ and $\mathcal{U}_{\mathrm{SC}}$ gates in a similar way to Section 4.2. However, this does not affect the complexity of the attack in this case, as the improvement achieved by a pre-filtering method is negligible. From Eq. (7), the complexity of the attack is

$$2^{0.44} \cdot \frac{\pi}{2} \cdot 2^{56} \cdot \left(\frac{\pi}{2} \cdot 2^{(40-19)/2} \cdot (\mathcal{C}(\mathcal{U}_{\mathrm{PF}}) + \mathcal{C}(\mathcal{U}_{SC})) + \mathcal{C}(\mathcal{U}_F) \right),$$

where $2^{0.44}$ is due to the success probability. As we described in Section 3.2.1, the attack only beats the generic attack in the Q-II setting, and $\mathcal{C}(\mathcal{U}_F) = 2^{62.91}$ in this setting. Therefore, the complexity of the attack is $2^{120.00}$ in the Q-II setting.

5 Conclusion

In this study, we revisited the previous quantum rebound attacks on `HCF-AES-256` and `HCF-ARIA-256`. We identified flaws in these attacks and corrected them. As a result, some of these attacks became less efficient than generic attacks and therefore invalid. Furthermore, we proposed improved quantum rebound attacks using nested quantum amplitude amplification and quantum state preparation. Our improved attacks reduce the time complexity of the previous attacks, making them more efficient than generic attacks again. We believe that our approach can be applied to other quantum cryptanalysis techniques that involve solving S-box differential equations.

Acknowledgments

We thank the anonymous reviewers of ToSC for their constructive comments. Their comments have greatly contributed to the improvement of this work. This work was supported in part by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2022-II221196, Regional strategic Industry convergence security core talent training business), and in part by the BK21 FOUR (Fostering Outstanding Universities for Research).

References

[ABB$^+$22] Jean-Philippe Aumasson, Daniel J. Bernstein, Ward Beullens, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, and Bas Westerbaan. SPHINCS+: Submission to the NIST post-quantum project (v.3.1). https://sphincs.org/data/sphincs+-r3.1-specification.pdf, June 2022.

[ABD$^+$21] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER: Algorithm Specifications and Supporting Documentation (v3.02). https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf, August 2021.

[BDK$^+$21] Shi Bai, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation (v3.1). https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf, February 2021.

[Ber09] Daniel J Bernstein. Cost analysis of hash collisions: Will quantum computers make sharcs obsolete. *SHARCS*, 9:105, 2009.

[BGB$^+$18] Ryan Babbush, Craig Gidney, Dominic W Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven. Encoding electronic spectra in quantum circuits with linear T complexity. *Physical Review X*, 8(4):041015, 2018.

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

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum cryptanalysis of hash and claw-free functions. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN 1998*, volume 1380 of *LNCS*, pages 163–169, April 1998.

[BK23] Seungjun Baek and Jongsung Kim. Quantum rebound attacks on reduced-round aria-based hash functions, 2023.

[BNS19] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. Quantum security analysis of AES. *IACR Trans. Symmetric Cryptol.*, 2019(2):55–93, 2019.

[Bon17] Xavier Bonnetain. Quantum key-recovery on full AEZ. In Carlisle Adams and Jan Camenisch, editors, *SAC 2017*, volume 10719 of *LNCS*, pages 394–406, August 2017.

[CKS21] Amit Kumar Chauhan, Abhishek Kumar, and Somitra Kumar Sanadhya. Quantum free-start collision attacks on double block length hashing with round-reduced AES-256. *IACR Trans. Symm. Cryptol.*, 2021(1):316–336, 2021.

[CNS17] André Chailloux, María Naya-Plasencia, and André Schrottenloher. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 211–240, December 2017.

[DMLQ21] Shanque Dou, Ming Mao, Yanjun Li, and Dongying Qiu. Quantum rebound attack to DM structure based on ARIA algorithm. In *Journal of Physics: Conference Series*, volume 2078, page 012003. IOP Publishing, 2021.

[DR99] Joan Daemen and Vincent Rijmen. AES proposal: Rijndael. 1999.

[DSS+20] Xiaoyang Dong, Siwei Sun, Danping Shi, Fei Gao, Xiaoyun Wang, and Lei Hu. Quantum collision attacks on AES-like hashing with low quantum random access memories. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 727–757, December 2020.

[FHK+20] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON: Fast-Fourier Lattice-based Compact Signatures over NTRU (v1.2). https://falcon-sign.info/falcon.pdf, October 2020.

[Goo] Google QuantumAI - One qubit gates. https://quantumai.google/cirq/google/devices#one_qubit_gates. Accessed: 2024-05-31.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996.

[Hir05] Shoichi Hirose. Provably secure double-block-length hash functions in a black-box model. In Choonsik Park and Seongtaek Chee, editors, *ICISC 04*, volume 3506 of *LNCS*, pages 330–342, December 2005.

[Hir06] Shoichi Hirose. Some plausible constructions of double-block-length hash functions. In Matthew J. B. Robshaw, editor, *FSE 2006*, volume 4047 of *LNCS*, pages 210–225. Springer, Heidelberg, March 2006.

[HS20] Akinori Hosoyamada and Yu Sasaki. Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 249–279, May 2020.

[IBM] IBM quantum platform documentation - native gates and operations. https://docs.quantum.ibm.com/run/native-gates#native-gates-and-operations. Accessed: 2024-05-31.

[JATK+24] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D. Nation, Lev S. Bishop, Andrew W. Cross, Blake R. Johnson, and Jay M. Gambetta. Quantum computing with Qiskit, 2024.

[Kit97] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191, 1997.

[KKP+04] Daesung Kwon, Jaesung Kim, Sangwoo Park, Soo Hak Sung, Yaekwon Sohn, Jung Hwan Song, Yongjin Yeom, E-Joong Yoon, Sangjin Lee, Jaewon Lee, Seongtaek Chee, Daewan Han, and Jin Hong. New block cipher: ARIA. In Jong In Lim and Dong Hoon Lee, editors, *ICISC 03*, volume 2971 of *LNCS*, pages 432–445, November 2004.

[KLLN16] 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, August 2016.

[KM10] Hidenori Kuwakado and Masakatu Morii. Quantum distinguisher between the 3-round feistel cipher and the random permutation. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 2682–2685. IEEE, 2010.

[KM12] Hidenori Kuwakado and Masakatu Morii. Security on the quantum-type even-mansour cipher. In *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*, pages 312–316. IEEE, 2012.

[LLYL23] Qing-bin Luo, Xiao-yu Li, Guo-wu Yang, and Qiang Li. Quantum reversible circuits for $GF(2^8)$ multiplication based on composite field arithmetic operations. *Quantum Information Processing*, 22(1):58, 2023.

[LM17] Gregor Leander and Alexander May. Grover meets simon - quantumly attacking the FX-construction. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 161–178, December 2017.

[LXX$^+$23] Da Lin, Zejun Xiang, Runqing Xu, Shasha Zhang, and Xiangyong Zeng. Optimized quantum implementation of aes, 2023.

[Moo16] Dustin Moody. Post-quantum cryptography standardization: Announcement and outline of nist's call for submissions. In *International Conference on Post-Quantum Cryptography-PQCrypto*, 2016.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In Orr Dunkelman, editor, *FSE 2009*, volume 5665 of *LNCS*, pages 260–276. Springer, Heidelberg, February 2009.

[MVBS05] Mikko Möttönen, Juha J. Vartiainen, Ville Bergholm, and Martti M. Salomaa. Transformation of quantum states using uniformly controlled rotations. *Quantum Inf. Comput.*, 5(6):467–473, 2005.

[Nat23a] National Institute of Standards and Technology. Module-lattice-based digital signature standard. (U.S. Department of Commerce, Washington, DC), Draft Federal Information Processing Standards Publication (FIPS) 204, https://doi.org/10.6028/NIST.FIPS.204.ipd, August 2023.

[Nat23b] National Institute of Standards and Technology. Module-lattice-based key-encapsulation mechanism standard. (U.S. Department of Commerce, Washington, DC), Draft Federal Information Processing Standards Publication (FIPS) 203, https://doi.org/10.6028/NIST.FIPS.203.ipd, August 2023.

[Nat23c] National Institute of Standards and Technology. Stateless hash-based digital signature standard. (U.S. Department of Commerce, Washington, DC), Draft Federal Information Processing Standards Publication (FIPS) 205, https://doi.org/10.6028/NIST.FIPS.205.ipd, August 2023.

[NC10] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information.* Cambridge university press, 2010.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134. IEEE Computer Society, 1994.

[vW94] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with application to hash functions and discrete logarithms. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, and Ravi S. Sandhu, editors, *ACM CCS 94*, pages 210–218. ACM Press, November 1994.

A Single-qubit rotation gate based quantum state preparation

We introduce a method based on the single-qubit rotation gates presented in [MVBS05], which can serve as an alternative to the QROM-based method discussed in Section 2.5. We explain the process of transforming an arbitrary quantum state to zero state for better understanding. First, we introduce the uniformly controlled rotation gate $\mathcal{R}_a^k(\alpha)$, defined by k control qubits, one target qubit, a rotation axis a, and rotation angles $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_{2^k}\}$. It comprises 2^k controlled rotation gates, each corresponding to all possible k-bit states, as illustrated in Figure 9.

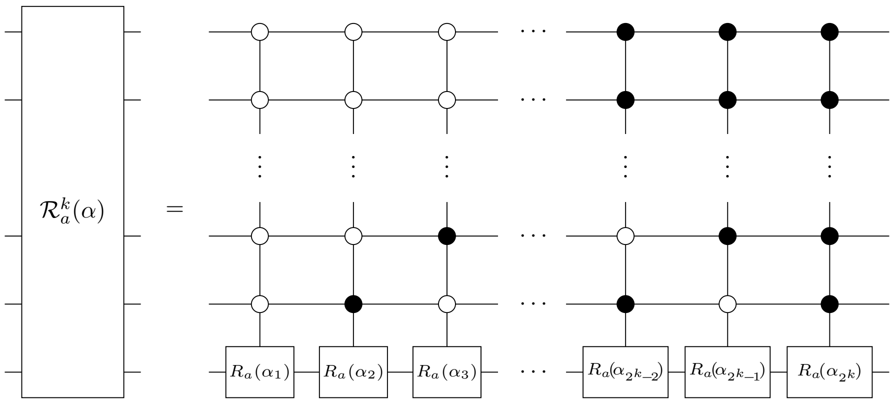

Figure 9: Uniformly controlled rotation gate $\mathcal{R}_a^k(\alpha)$.

Our objective is to transform an arbitrary n-qubit quantum state

$$|\psi\rangle = \sum a_i \, |i\rangle_n$$

into a zero state $|0\rangle_n$ using uniformly controlled rotation gates about y-axis. The y-axis refers to one of the axes in the Bloch sphere representation of a qubit's state.[6] As an example, the overall strategy for $n = 8$ is illustrated in Figure 10.

$$|\psi\rangle_8 \xrightarrow{\mathcal{R}_y^7(\alpha^1)} |\psi_1\rangle_7 \otimes |0\rangle_1 \xrightarrow{\mathcal{R}_y^6(\alpha^2)} |\psi_2\rangle_6 \otimes |0\rangle_2 \xrightarrow{\mathcal{R}_y^5(\alpha^3)} \cdots |\psi_7\rangle_1 \otimes |0\rangle_7 \xrightarrow{\mathcal{R}_y^0(\alpha^7)} |0\rangle_8$$

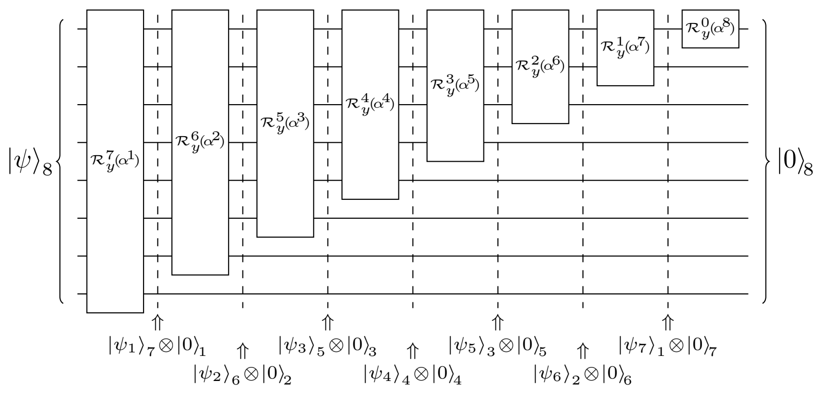
$|\psi\rangle_8$
$|0\rangle_8$
$\mathcal{R}_y^7(\alpha^1)$
$\mathcal{R}_y^6(\alpha^2)$
$\mathcal{R}_y^5(\alpha^3)$
$\mathcal{R}_y^4(\alpha^4)$
$\mathcal{R}_y^3(\alpha^5)$
$\mathcal{R}_y^2(\alpha^6)$
$\mathcal{R}_y^1(\alpha^7)$
$\mathcal{R}_y^0(\alpha^8)$
$|\psi_1\rangle_7 \otimes |0\rangle_1$
$|\psi_2\rangle_6 \otimes |0\rangle_2$
$|\psi_3\rangle_5 \otimes |0\rangle_3$
$|\psi_4\rangle_4 \otimes |0\rangle_4$
$|\psi_5\rangle_3 \otimes |0\rangle_5$
$|\psi_6\rangle_2 \otimes |0\rangle_6$
$|\psi_7\rangle_1 \otimes |0\rangle_7$

Figure 10: Structure of quantum circuit transforming $|\psi\rangle$ to $|0\rangle$.

[6]For more details on the Bloch sphere, see [NC10].

Now, we will describe how to find appropriate angles α^1, α^2, …, α^n. As a first step, we find $\alpha^1 = \{\alpha_1^1, \alpha_2^2 \dots \alpha_{2^{n-1}}^1\}$ such that $\mathcal{R}_y^{n-1}(\alpha^1)$ makes the last qubit zero, i.e., $\mathcal{R}_y^{n-1}(\alpha^1)\,|\psi\rangle = |\psi_1\rangle_{n-1} \otimes |0\rangle$. Specifically, for $(n-1)$-bit string $b = 0, 1, \dots, 2^{n-1}-1$,

$$\alpha_{b+1}^1 = 2\arcsin\left(|a_{b||1}| \Big/ \sqrt{|a_{b||0}|^2 + |a_{b||1}|^2}\right).$$

Then, $R_y(\alpha_{b+1}^1)(a_{b||0}\,|b\rangle\,|0\rangle + a_{b||1}\,|b\rangle\,|1\rangle) = a'_{b||0}\,|b\rangle\,|0\rangle$.

Similarly, by subsequently applying $R_y^{n-2}(\alpha^2)$, $R_y^{n-3}(\alpha^3)$, …, $R_y^0(\alpha^n)$, we can transform $|\psi\rangle$ to $|0\rangle$. The j-th angle for α^k can be computed as follows:

$$\alpha_j^k = 2\arcsin\left(\sqrt{\left(\sum_{l=1}^{2^k-1} |a_{(2j-1)2^{k-1}+l}|^2\right) \Big/ \left(\sum_{l=1}^{2^k} |a_{(j-1)2^k+l}|^2\right)}\right),$$

where $k = 1, 2, \dots n$ and $j = 1, 2, \dots, 2^{n-k}$.

Finally, we introduce how to decompose uniformly controlled rotation gates into single-qubit rotation gates and CNOT gates. We only provide the method here. For a validation of this method, please refer to the original work [MVBS05]. $\mathcal{R}_a^n(\alpha)$ gate can be decomposed into a sequence of alternating single-qubit rotation gate and CNOT gate repeated 2^n times. The illustration for the case when $n = 3$ is depicted in Figure 11.

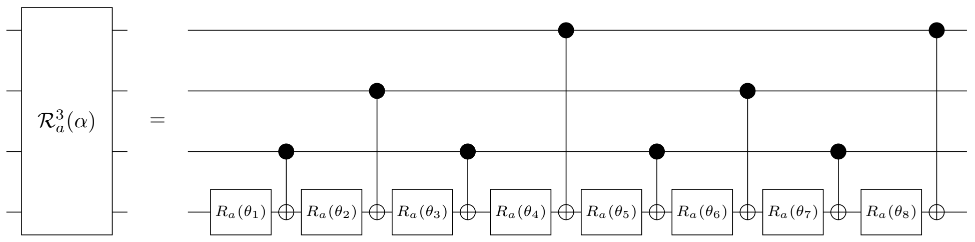

Figure 11: Decomposition example of uniformly controlled rotation gate $R_a^3(\alpha)$.

Defnie $\theta = \{\theta_1, \theta_2, \dots, \theta_{2^n}\}$ where θ_i represents the angle for i-th single-qubit rotation gates, and $q = \{q_1, q_2, \dots, q_{2^n}\}$ where q_i represents the control qubit of i-th CNOT gate. For example, in Figure 11, $q = \{3, 2, 3, 1, 3, 2, 3, 1\}$. Note that the qubit on which all single-qubit rotation gates act, and the target qubit of all CNOT gates, is the same as the target qubit of the $\mathcal{R}_a^n(\alpha)$ gate. We can compute θ and q as follows:

1. M is a $2^n \times 2^n$ matrix with $M[i][j] = 2^{-n}(-1)^{b_{j-1} \cdot g_{i-1}}$ where b_x and g_x represents the binary code and Gray code representation of integer x.
2. $\theta = M \cdot \alpha$.
3. $q_i = n - j$ where $i = i' \cdot 2^j$ and i' is odd.

We have implemented the single-qubit-based method using the Qiskit library [JATK+24], and the code is available at[7]

A.1 Comparison with the QROM-Based Method

In this comparison of two methods for quantum state preparation, we will ignore errors that occur in a quantum computing environment. Although the gate count for both methods is

[7]The content is being removed because the URL contains the author's name. It will be made public at a later time.

similar, the actual cost of single-qubit rotation gates and Toffoli gates can vary significantly depending on the quantum computing environment. The efficiency of each method depends on which operations are provided as *native gates* by the computing platform. Native gates are the fundamental operations that a quantum processor can perform directly, as opposed to logic gates, which may need to be constructed from multiple native gates. If we assume a native gate set based on Clifford+T gates, as is common in evaluating the complexity of quantum algorithms in the symmetric key cryptography field, then the cost of Toffoli gates is relatively lower. This is because approximating a single-qubit rotation gate with arbitrary angles using Clifford+T gates can be costly.[8]

On the other hand, there are environments where single-qubit rotation gates, especially R_y gate, are more efficient. For instance, the IBM Quantum Platform includes R_z and X gates in the native gate set for Heron and Eagle processors [IBM].[9] Google's documentation states that Google devices support arbitrary one-qubit gates of any rotation [Goo]. In such cases, single-qubit rotation gates can become more cost-effective than Toffoli gates, making the single-qubit based method more efficient. Additionally, single-qubit rotation gate based method has the advantage of not requiring ancilla qubits.

B Pre-filtering for quantum rebound attack on HCF-ARIA-256

Recall that the quantum rebound attack on `HCF-ARIA-256` required 2^{112} starting points. [CKS21] set the each search space for Δ^1_{in}, Δ^2_{in}, Δ^1_{out}, and Δ^2_{out} as $\mathbb{F}_2^{28}$. However, there is also no specific mention of how the search space was constructed while each has 7 active bytes.

We set a size of search space for both inbound phases 1 and 2 as 2^{56} and explore all possible combinations of selecting 7 bytes to be fixed among 14 active bytes in each inbound phase. As a result, we find that the case depicted in Figure 12 maximizes the reduction in search space. Specifically, for inbound phase 1:

1. Fix the value for $\Delta^1_{out}[4,6,8,9,13]$ and $\Delta^1_{in}[4,9]$.
2. Considering DL in Round 3, $\Delta Y_3[1,2,6,8,12,15]$ is fixed.
3. Considering SL in Round 3, each byte of $\Delta X_3[1,2,6,8,12,15]$ is limited to 127 values.
4. Considering DL in Round 2, where $\Delta Y_2[0,1,2,5,7,10,11,12,15] = 0$, the following relationships hold:

$$\begin{aligned}\Delta Z_2[1] &= \Delta^1_{in}[8] \oplus \Delta^1_{in}[9],\\ \Delta Z_2[2] &= \Delta^1_{in}[4] \oplus \Delta^1_{in}[6],\\ \Delta Z_2[6] &= \Delta^1_{in}[9] \oplus \Delta^1_{in}[13],\\ \Delta Z_2[8] &= \Delta^1_{in}[4] \oplus \Delta^1_{in}[13],\\ \Delta Z_2[12] &= \Delta^1_{in}[6] \oplus \Delta^1_{in}[9],\\ \Delta Z_2[15] &= \Delta^1_{in}[4] \oplus \Delta^1_{in}[8].\end{aligned}$$

   Since each byte of $\Delta Z_2[1,2,6,8,12,15]$ is limited to 127 values and $\Delta^1_{in}[4,9]$ is fixed, each byte of $\Delta^1_{in}[6,8,13]$ can have approximately $64(= 256/4)$ values.

[8]The Solovay-Kitaev theorem shows that such approximations require a polylogarithmic number of gates [Kit97].

[9]$R_y(\theta) = XR_z(\theta)X$.

Similarly, by fixing $\Delta^2_{in}[4,6,8,9,13]$ and $\Delta^2_{out}[4,9]$, we can pre-filter the impossible values for $\Delta^2_{out}[6,8,13]$. Furthermore, we can appropriately select the values of $\Delta^1_{out}[4,6,8,9,13]$ and $\Delta^2_{in}[4,6,8,9,13]$ to ensure that there are always solutions for the 5 S-box differential equations between ΔX_4 and ΔY_4. As a result, we can remove 5 auxiliary bits from the search space. Additionally, for each of the pairs $(\Delta^1_{out}[3], \Delta^2_{in}[3])$ and $(\Delta^1_{out}[14], \Delta^2_{in}[14])$, we can pre-filter half of the values where solutions for the corresponding S-box differential equations between ΔX_4 and ΔY_4 do not exist. Consequently, we can reduce the size of overall search space by a factor of 2^{19}.

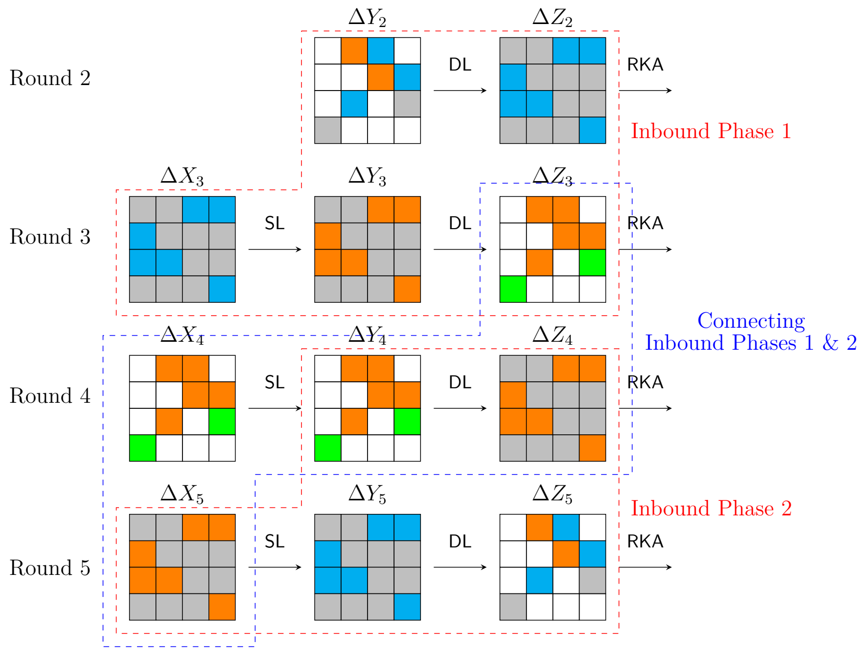

Figure 12: Pre-filtering search space for quantum rebound attack on `HCF-ARIA-256`: If we fix a value for $\Delta^1_{in}[4,9]$, $\Delta^1_{out}[4,6,8,9,13]$, $\Delta^2_{in}[4,6,8,9,13]$, and $\Delta^2_{out}[4,9]$, each byte of $\Delta^1_{in}[6,8,13]$ and $\Delta^2_{out}[6,8,13]$ can take limited values. $(\Delta^1_{out}[3], \Delta^2_{in}[3])$ and $(\Delta^1_{out}[14], \Delta^2_{in}[14])$ can take limited values because of the `SL` of Round 4. The orange cell indicates the cell whose value is fixed. The blue cell indicates the cell that can have limited values. The green cell indicates a cell whose value, when concatenated with another green cell, can have limited values.