IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2024, No. 4, pp. 64–96. DOI:10.46586/tosc.v2024.i4.64-96

Chosen-Prefix Collisions on AES-like Hashing

Shiyao Chen[1], Xiaoyang Dong[2,3,4(✉)], Jian Guo[5] and Tianyu Zhang[5]

[1] Digital Trust Centre, Nanyang Technological University, Singapore, Singapore
shiyao.chen@ntu.edu.sg
[2] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878,
People's Republic of China
xiaoyangdong@tsinghua.edu.cn
[3] Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing,
People's Republic of China
[4] Zhongguancun Laboratory, Beijing, People's Republic of China
[5] Division of Mathematical Sciences, School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore, Singapore
guojian@ntu.edu.sg,tianyu005@e.ntu.edu.sg

Abstract. Chosen-prefix collision (CPC) attack was first presented by Stevens, Lenstra and de Weger on MD5 at Eurocrypt 2007. A CPC attack finds a collision for any two chosen prefixes, which is a stronger variant of collision attack. CPCs are naturally harder to construct but have larger practical impact than (identical-prefix) collisions, as seen from the series of previous works on MD5 by Stevens *et al.* and SHA-1 by Leurent and Peyrin. Despite its significance, the resistance of CPC attacks has not been studied on AES-like hashing.

In this work, we explore CPC attacks on AES-like hashing following the framework practiced on MD5 and SHA-1. Instead of the message modification technique developed for MD-SHA family, we opt for related-key rebound attack to construct collisions for AES-like hashing in view of its effectiveness. We also note that the CPC attack framework can be exploited to convert a specific class of one-block free-start collisions into two-block collisions, which sheds light on the importance of free-start collisions. As a result, we present the first CPC attacks on reduced Whirlpool, Saturnin-hash and AES-MMO/MP in classic and quantum settings, and extend the collision attack on Saturnin-hash from 5 to 6 rounds in the classic setting. As an independent contribution, we improve the memoryless algorithm of solving 3-round inbound phase by Hosoyamada and Sasaki at Eurocrpyt 2020, which leads to improved quantum attacks on Whirlpool. Notably, we find the first 6-round memoryless quantum collision attack on Whirlpool better than generic CNS collision finding algorithm when exponential-size qRAM is not available but exponential-size classic memory is available.

Keywords: Chosen-Prefix Collision · Related-Key Rebound Attack · Quantum Cryptanalysis · Whirlpool · Saturnin-hash · AES-MMO/MP

1 Introduction

A hash function maps an aribtrary-size message to a fixed-size hash value. A secure hash function must satisfy three fundamental security requirements: preimage resistance, second-preimage resistance, and collision resistance. Hash functions play a crucial role in cybersecurity, especially in digital signatures. One popular approach to build a cryptographic hash function is to plug a compression function into the Merkle-Damgård construction [Mer89; Dam], which usually has two categories: MD-SHA family (e.g., MD5 [Riv92], SHA-1 [ST95], and etc.) and AES-like hashing (e.g., AES-MMO [All17; ISO10], Whirlpool [BR00; ISO04],

Received: 2024-09-01 Accepted: 2024-11-01 Published: 2024-12-18

and etc.). However, due to the impressive advances in hash function cryptanalysis in 2004-2005, in particular against the MD-SHA family by Wang et al. [WY05; WLFCY05; WYY05a; WYY05b], the NIST in 2007 decided to call for a new cryptographic hash function to provide alternatives to the MD-SHA family, and eventually announced KECCAK [BDPA13] as the winner to be standardized as the new SHA-3 [ST15] in 2015. In this paper, we focus on the notion of collision resistance on AES-like hashing, *i.e.*, it should be computationally infeasible to find two messages with the same hash value.

Chosen-Prefix Collision. In 2006, a *stronger* variant of collision was brought to concern by Gauravaram *et al.* [GMD06] at AusCERT as well as De Cannière and Rechberger [CR06] at Asiacrypt, namely chosen-prefix collision (CPC). A CPC attack aims at finding a pair of suffixes (or messages) S_0 and S_1 for any two chosen prefixes (or initial values) P_0 and P_1 such that $H(P_0||S_0) = H(P_1||S_1)$. At Eurocrypt 2007, Stevens, Lenstra, and de Weger presented the very first CPC attack. They mounted the attack on MD5 and found colliding X.509 certificates for different identities [SLW07]. The novel method was then improved and used to create a rogue certificate authority (CA) at Crypto 2009 [Ste+09]. The series of work on MD5 by Stevens and others are later summarized in [SLW12]. At Eurocrypt 2019, Leurent and Peryin composed the first CPC attack on SHA-1 [LP19] by extending and systematically improving Steven *et al.*'s idea on MD5. The attack on SHA-1 was further revised with implementation on GPU [LP20]. As a result, a chosen-prefix collision was found on SHA-1 with 900 Nvidia GTX1060 in two months, which directly leads to the shambles of PGP/GnuPG Web of Trust.

A chosen-prefix collision attack is harder than a (identical-prefix) collision attack. This can be obtained from a theoretic perspective through reduction (see Section 2.1 for more details). From the aspect of attack complexity, the best collision on MD5 has the time complexity of 2^{16} compared to the best CPC on MD5 with 2^{39} [Ste+09]. The practical impact of a CPC attack is also significantly larger, as seen from its impact on certificates (rogue CA) [SLW07] and protocols (TLS, SSH, IKE) [BL16]. This is partially due to the fact that the adversary has more control over the colliding messages under the CPC setting. Thus, both CPC results on MD5 and SHA-1 directly composed a compelling case on immediately terminating their usage in cryptographic constructions.

Rebound Attack. Introduced by Mendel *et al.* at FSE 2009 [MRST09], rebound attack is among the most powerful cryptanalysis techniques to find collisions on AES-like hashing. A rebound attack consists of an inbound phase and an outbound phase. In the inbound phase, the attacker efficiently finds a conforming pair of a short but dense differential characteristic. The number of conforming pairs can be generated in the inbound phase is denoted as the inbound degree of freedom (DoF). The outbound phase refers to the computation that is not covered by the inbound phase. It propagates the starting point from the inbound phase to fulfill the sufficient constraints for a collision in a probabilistic manner, which we denote as DoF consumption. Thus, the inbound phase must provide enough DoF for the outbound phase. Since the introduction of rebound attacks, many techniques have been proposed to further extend its power, such as mutliple inbound [LMRRS09], Super S-box [GP10; LMRRS09], fast list merge [Nay11], dissection [DDKS12], non-full-active Super S-box [SWWW12], triangulating rebound [DGLP22], bit-based rebound [TSITI24]. Matured and refined over the years, rebound attack has now become a cardinal technique in collision attacks on AES-like hashing. The pioneer work by Mendel *et al.* [MRST09] is highly appreciated by the community and was awarded the FSE Test-of-Time award[1] in March 2024.

[1] https://tosc.iacr.org/index.php/ToSC/ToT_Award

Quantum Implications on Rebound Attacks. The past decade has witnessed notable advances in the quantum cryptanalysis of symmetric-key primitives, such as the attacks on Feistel [KM10], EM construction [KM12; BHNSS19], MACs and AEAD [KLLN16; HI21; BLNS21], FX construction [LM17], generic multi-collision finding [HSX17; LZ19], and Merkle-Damgård construction [AGL22], etc.

Quantum rebound attacks were first explored at Eurocrypt 2020 by Hosoyamada and Sasaki [HS20], where the rebound attack was rephrased as a nested Grover search problem [SS24]. In [HS20], the authors presented a quantum algorithm capable of exploiting the differential of the inbound phase and outbound phase that is not feasible in the classical setting for rebound attacks. As a result, they have succeeded in extending one more attack round on `AES-MMO/MP` and `Whirlpool` compared to the previous best classical attack results. The method has alerted the community that quantum adversaries are able to inspect more weaknesses of a primitive instead of a plain quadratic speedup.

At Asiacrypt 2020, Dong *et al.* [Don+20] found quantum rebound attacks below the CNS collision bound [CNS17] with reduced or no quantum random access memory, and Flórez-Gutiérrez *et al.* [Fló+20] proposed quantum collision attacks on `Gimli`. Later at Crypto 2021, Hosoyamada and Sasaki proposed quantum collision attack on round-reduced `SHA-2` [HS21]. Dong *et al.* proposed quantum free-start collision attacks at Asiacrypt 2021 [Don+21b]. At Crypto 2022, Dong *et al.* [DGLP22] introduced the Super-Inbound technique suited with triangulation algorithm to further extend the attack rounds on `AES-MMO/MP`.

Motivation and Contribution. Notwithstanding the significant impact of CPC attacks on hash functions, the resistance against CPC attacks is yet to be investigated on `AES-like` hashing. Besides, the CPC attack framework is able to construct collisions based on a specific class of one-block free-start collision attacks which finds a balance between the DoF from the key schedule and attack complexity. This technique could be a meaningful alternative to mount two-block collision from one-block free-start collision on `AES-like` hashing. However, previous work drains the key schedule DoF to compensate the probability of the outbound phase or to reach more rounds [Don+21b; DGLP22], thus cannot be used to construct collisions. Thus, having these in mind, we achieve the following results summarized in Table 1.

- We extend the CPC attack framework into `AES-like` hashing. Our attacks starts with a birthday phase to ensure a desired difference between the chaining values, then a related-key rebound attack is adopted to construct collision in view of its effectiveness in `AES-like` hashing, instead of the message modification technique that is developed for ARX ciphers and well-practised on `MD-SHA` family.

- As a result, we provide the *first* CPC attacks on reduced `Whirlpool`, `Saturnin-hash`, and `AES-MMO/MP`, both in classic and in quantum. We observed the difference between `MMO` and `MP` modes against CPC attacks in the classical setting when instantiating with `AES-128`. It is noteworthy that `MMO` and `MP` modes are generally reckoned as equivalent in prior works [HS20; Don+20; DGLP22] if only single-key differentials are used. We also improve the classical collision attack on `Saturnin-hash` from the previous best 5 rounds [Don+21b] to 6 rounds using the attack framework. Hence, when assessing the collision resistance of a given target, both the conversion from semi-free-start collision and from free-start collision should be considered, as the corresponding resistance can be different.

- As an independent contribution, we improve the memoryless algorithm in [HS20] of solving 3-round inbound phase. Thereby, all previous quantum attacks based on this algorithm are improved, *e.g.*, the 6-round collision attack in [HS20] and

9-round free-start collision attack in [Don+21b] on `Whirlpool`. Notably, our 6-round quantum collision attack on `Whirlpool` has time complexity of $2^{201.4}$ without classical or quantum memory. Our attack is better than the generic CNS algorithm [CNS17] under the assumption that no exponential-size qRAM is available but large classical memory is available.

Table 1: Summary of our results on `AES-like` hashing

Target	Attack	Rounds	Time	cMem	qRAM	C*	QA*	QB*	QC*	Source
`Whirlpool`										
Hash function	Collision	4	2^{120}	2^{16}	-	✓	-	-	-	[MRST09]
	Collision	5	2^{120}	2^{64}	-	✓	-	-	-	[GP10; LMRRS09]
	Collision	6	2^{248}	2^{248}	-	✓	-	-	-	[Don+21a]
	Collision	6	2^{240}	2^{240}	-	✓	-	-	-	[CGLSZ24]
	Collision	6	2^{228}	-	-	-	✓	-	-	[HS20]
	Collision†	**6**	$\mathbf{2^{201.4}}$	-	-	-	✓	✓	-	**Section 5.2**
	CPC	**6**	$\mathbf{2^{205.4}}$	-	-	-	✓	-	-	**Section 5.1**
Compression function	Semi-free-start	5	2^{120}	2^{16}	-	✓	-	-	-	[MRST09]
	Semi-free-start	7	2^{184}	2^{8}	-	✓	-	-	-	[LMRRS09]
	free-start	8	2^{120}	2^{8}	-	✓	-	-	-	[SWWW12]
	free-start	9	$2^{220.5}$	-	-	-	✓	-	-	[Don+21b]
	free-start†	**9**	$\mathbf{2^{204.53}}$	-	-	-	✓	✓	-	**Section 5.2**
Any	any	any	2^{256}	-	-	✓	✓	-	-	[OW99; Ber09; HS20]
	any	any	$2^{204.8}$	$2^{102.4}$	-	-	-	✓	-	[CNS17]
	any	any	$2^{170.67}$	-	$2^{170.67}$	-	-	-	✓	[BHT98]
`Saturnin-hash`										
Hash function	Collision	5	2^{64}	2^{64}	-	✓	-	-	-	[Don+21b]
	Collision	7	$2^{113.5}$	-	-	-	✓	-	-	[Don+21b]
	CPC	**6**	$\mathbf{2^{112}}$	$\mathbf{2^{64}}$	-	✓	-	-	-	**Section 6.1**
	CPC‡	**7**	$\mathbf{2^{118.70}}$	-	-	-	✓	-	-	**Section 6.2**
Any	any	any	2^{128}	-	-	✓	✓	-	-	[OW99; Ber09; HS20]
	any	any	$2^{102.4}$	$2^{51.2}$	-	-	-	✓	-	[CNS17]
	any	any	$2^{85.33}$	-	$2^{85.33}$	-	-	-	✓	[BHT98]
`AES-128-MMO/MP`										
Hash function	Collision	5	2^{56}	2^{4}	-	✓	-	-	-	[MRST09]
	Collision	6	2^{56}	2^{32}	-	✓	-	-	-	[LMRRS09; GP10]
	Collision	7	$2^{42.5}$	-	2^{48}	-	-	-	✓	[HS20]
	Collision	7	$2^{59.5}$	-	-	-	✓	-	-	[HS20]
	Collision	7	$2^{45.8}$	-	-	-	✓	✓	-	[Don+20]
	Collision	8	$2^{55.53}$	-	-	-	✓	-	-	[DGLP22]
	CPC, `MMO`‡	**5**	$\mathbf{2^{57}}$	$\mathbf{2^{32}}$	-	✓	-	-	-	**Appendix D**
	CPC, `MP`	**5**	$\mathbf{2^{52}}$	$\mathbf{2^{32}}$	-	✓	-	-	-	**Section 7**
	CPC	**6**	$\mathbf{2^{61.5}}$	-	-	-	✓	-	-	**Appendix E**
Any	any	any	2^{64}	-	-	✓	✓	-	-	[OW99; Ber09; HS20]
	any	any	$2^{51.2}$	$2^{25.6}$	-	-	-	✓	-	[CNS17]
	any	any	$2^{42.67}$	-	$2^{42.67}$	-	-	-	✓	[BHT98]

⋆ The attack is valid in the classical setting.

* The attack is valid in the quantum setting under assumptions below respectively:
QA: neither exponential-size qRAM nor classic memory is available.
QB: exponential-size qRAM is not available while exponential-size classical memory is available.
QC: exponential-sized qRAM is available.

† The result is obtained by improving the algorithm of solving 3-round inbound phase in [HS20].

‡ The CPC attack is comparable to previous best collision result at corresponding round.

Outline. In Section 2, we briefly introduce collision and variants, `AES-like` hashing, quantum computing and rebound attacks. Our CPC attack framework on `AES-like` hashing is presented in Section 3. We then provide our improved algorithm on solving 3-round inbound in Section 4. In Section 5, 6 and 7, we present our attacks on `Whirlpool`, `Saturnin-hash` and `AES-MP`. Finally, we concludes our paper in Section 8.

2 Preliminaries

2.1 Collision and Variants

Given a hash function H, we have the following notions of collisions ranked from the strongest to the weakest:

- A collision attack finds a pair of messages M_0, M_1 for a chosen initial value IV such that $H(IV||M_0) = H(IV||M_1)$;
- A semi-free start collision attack finds a pair of messages M_0, M_1 and any initial value IV' such that $H(IV'||M_0) = H(IV'||M_1)$;
- A free-start collision attack finds finds a pair of messages M_0, M_1 and a pair of initial values IV'_0, IV'_1, such that $H(IV'_0||M_0) = H(IV'_1||M_1)$.

A chosen-prefix collision (CPC) attack is defined

- A CPC attack finds a pair of suffixes S_0 and S_1 for any two arbitrarily chosen prefixes P_0 and P_1, such that $H(P_0||S_0) = H(P_1||S_1)$.

A CPC problem is a harder variant of collision, which can be intuitively observed from a reduction by setting $P_0 = P_1 = IV$. In the classical setting, the generic attack complexity to find collisions (and variants) for a hash function with an n-bit output is $O(2^{n/2})$ due to the birthday paradox. As such, any dedicated attack that finds a collision (or variant) with less than $O(2^{n/2})$ complexity constitutes an attack. In the quantum setting, the generic bound depends on the assumptions on quantum and classic memory, which will be introduced in Section 2.3.

2.2 AES-like Hashing

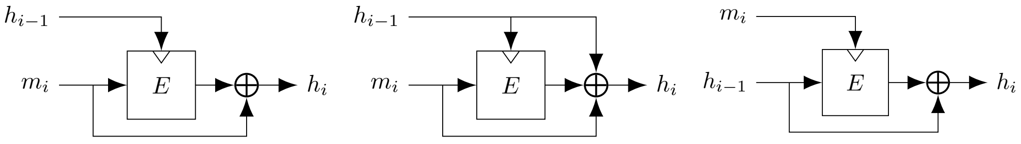


Figure 1: A selection of PGV modes

It is a common and practical approach to convert a secure block cipher through PGV modes [PGV93] into the compression function used in a hash function. A selection of PGV modes is illustrated in Figure 1 and expressed in formulas:

- MMO mode: $CF(h_{i-1}, m_i) = E_{h_{i-1}}(m_i) \oplus m_i$.
- MP mode: $CF(h_{i-1}, m_i) = E_{h_{i-1}}(m_i) \oplus m_i \oplus h_{i-1}$.
- DM mode: $CF(h_{i-1}, m_i) = E_{m_i}(h_{i-1}) \oplus h_{i-1}$.

AES [DR02] is a selection of specifications from the Rijndael block cipher family with b-bit block and k-bit key, where $b = 128$ and $k \in \{128, 192, 256\}$. We later refers different AES specifications as AES-k, *i.e.*, AES-128, AES-192, AES-256. The round function of AES is depicted in Figure 2. Since its standardisation by NIST in 2001, AES has been widely adopted in many applications owing to its outstanding record of withstanding attacks. It has also inspired many designs in homage to or build with AES operations. Consequently, hash functions with a compression function based on or similar to the AES round function is refered to as AES-like hash functions or AES-like hashing, to list a few: Whirlpool [BR00], Grøstl [Gau+09], Saturnin-hash [Can+20], etc. AES-like hash functions are widely used in applications: AES-128 in MMO mode is used in the standards of the Zigbee protocol suite [All17] and ISO/IEC [ISO10], Whirlpool is adopted as the ISO/IEC standard in [ISO04], etc.

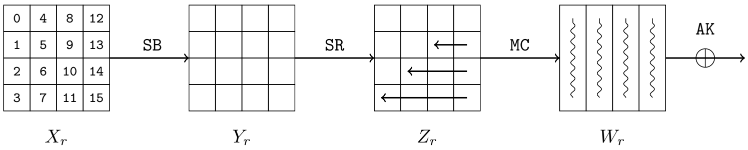


Figure 2: The AES round function

AES-like ciphers usually follow the Substitution-Permutation Network (SPN) design and organize the internal state as a two-dimensional matrix. An AES-like round function is usually composed of SubBytes (SB), ShiftRows (SR), MixColumn (MC), and AddRoundKey (AK), or some related variants of those:

- SubByte employs the S-box for each cell.
- ShiftRow rotates each row cyclically by some constant.
- MixColumn left multiplies each column by a constant matrix.
- AddRoundKey mixes the round key with bit-wise exclusive-OR (XOR).

2.3 Quantum Computing

The state space of an n-qubit quantum system is the set of all unit vectors in $\mathbb{C}^{2^n}$ under the orthonormal basis $\{|0\cdots 00\rangle, |0\cdots 01\rangle, \cdots, |1\cdots 11\rangle\}$, alternatively written as $\{|i\rangle : 0 \le i < 2^n\}$. Quantum computing is achieved by manipulating the state of an n-qubit system by a sequence of unitary transformations and measurements.

Superposition Oracles of Boolean Functions. The superposition oracle of a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is a unitary transformation $\mathcal{U}_f$ on an $(n+1)$-qubit system with the following functionality:

$$\mathcal{U}_f\left(\sum_{x\in\mathbb{F}_2^n} a_i |x\rangle |y\rangle\right) = \sum_{x\in\mathbb{F}_2^n} a_i |x\rangle |y \oplus f(x)\rangle .$$

Grover's Algorithm. Given a quantum black-box access to a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ with $0 < f^{-1}(1) \ll 2^n$, Grover's algorithm finds an element $x \in \mathbb{F}_2^n$ such that $f(x) = 1$ with a high probability and $O(\sqrt{2^n/|f^{-1}(1)|})$ superposition oracle calls: the algorithm iteratively applies the unitary transformation $(2|\psi\rangle\langle\psi| - I)\mathcal{U}_f$ upon the uniform superposition $|\psi\rangle = \frac{1}{\sqrt{2^n}}\sum_{x\in\mathbb{F}_2^n}|x\rangle$. This process amplifies the amplitudes of those values x with $f(x) = 1$. Finally, a measurement gives a value x of interest with an overwhelming probability [Gro96].

Quantum Generic Bounds. As discussed in [HS20], the quantum generic bounds depend on different assumptions regarding the existence of quantum random-access memory (qRAM):

- Assumption QA: neither exponential-size qRAM nor classic memory is available.

  The quantum version of parallel rho's algorithm achieves a time-space trade off of time $2^{n/2}/S$ with S as the maximum size of quantum computers and classical memory [OW99; Ber09; HS20].

- Assumption QB: exponential-size qRAM is not available while exponential-size classical memory is available.

  Chailloux, Naya-Plasencia, and Schrottenloher introduced the CNS algorithm to find a collision in time $O(2^{2n/5})$ with a quantum computer of size $O(1)$ and $O(2^{n/5})$ classical memory [CNS17].

- Assumption QC: exponential-size qRAM is available.

  Brassard, Høyer, and Tapp introduced the generic quantum collision attack (BHT algorithm) with $2^{n/3}$ quantum time complexity and $2^{n/3}$ qRAM [BHT98].

2.4 Rebound Attack

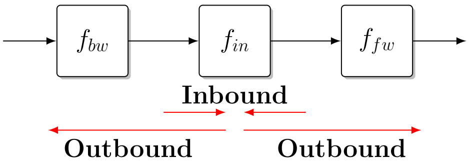


Figure 3: The Rebound Attack

Rebound attack was first proposed by Mendel *et al.* at FSE 2009 [MRST09]. Abstractly, the technique is used to generate a pair of inputs fulfilling a desired differential $\delta \to \Delta$ for a cipher. A high-level overview of the rebound attack is depicted in Figure 3. A rebound attack constitutes two phases:

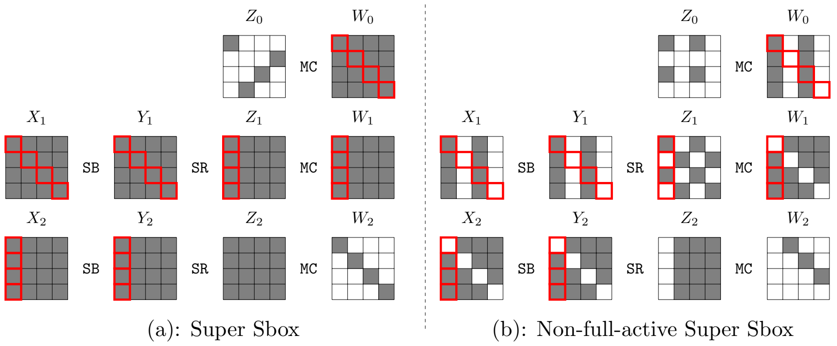
- Inbound Phase: A sufficient amount of pairs of input and output, or starting points, is generated to conform the inbound differential.

- Outbound Phase: The starting points are propagated forward and backward to fulfill the collision condition probabilistically.

The inbound phase must provide enough DoF for the outbound phase. We denote the probability of a starting point fulfill the collision conditions in the outbound phase as p. Thus, at least $1/p$ starting points should be generated in the inbound phase. Alternatively, the DoF of the inbound phase should be at least $1/p$.

Super S-box Technique. In 2009, the super S-box technique was proposed by Gilbert *et al.* [GP10] and Lamberger *et al.* [LMRRS09]. The technique extends Mendel *et al.*'s construction of the inbound phase by identifying four independent permutations across two consecutive `AES` rounds as four super S-boxes, which is as shown in Figure 4 (a). In [SLWSO10], Sasaki *et al.* further reduced the memory complexity by considering non-full-active super S-boxes as shown in Figure 4 (b).

In the following, we consider a general scenario where the internal state of the cipher is a $d \times d$ matrix of c-bit cells and discuss the two techniques in detail:



(a): Super Sbox

(b): Non-full-active Super Sbox

Figure 4: A truncated differential with full or non-full-active super S-box

Full-active Super S-box. As shown in Figure 4 (a), with the input difference $\Delta X_1^{(i)}$ of the i-th super S-box $\mathtt{SSB}_i$ (marked by red lines), we can compute to $\Delta Y_2^{(i)} = \mathtt{SSB}_i(x \oplus \Delta X_1^{(i)}) \oplus \mathtt{SSB}_i(x)$ for $x \in \mathbb{F}_2^{dc}$ and store the pair $(x, x \oplus \Delta X_1^{(i)})$ in a table $\mathbb{L}^{(i)}[\Delta Y_2^{(i)}]$.

In the inbound phase, the full-active super S-box technique works as follows. Given the input difference $\Delta_{in} = \Delta Z_0$, we compute $\Delta X_1^{(i)}$ for $0 \leq i \leq d-1$, then build d tables $\mathbb{L}^{(0)}, \ldots, \mathbb{L}^{(d-1)}$ as mentioned above. For each $\Delta_{out} = \Delta W_2 \in \mathbb{F}_2^{dc}$, compute $\Delta Y_2^{(i)}$ (with $0 \leq i \leq d-1$) to access the table $\mathbb{L}^{(i)}[\Delta Y_2^{(i)}]$ to obtain a pair conforming to the truncated differential of the inbound part. Hence, for given Δ_{in}, we need $d \times 2^{dc}$ memory to store d tables, and will obtain $|\Delta_{out}| = 2^{dc}$ pairs on average satisfying the inbound part.

At Eurocrypt 2020, Hosoyamada and Sasaki [HS20] converted the classical super S-box technique into a quantum one. They introduced two quantum ways. The first one is to use the qRAM to replace the classical memory to store the super S-box, which needs an exponential size of qRAM. The second one is to apply Grover's algorithm to search a conforming pair for a given input-output difference $(\Delta X_1^{(i)}, \Delta Y_2^{(i)})$ of $\mathtt{SSB}_i$. This method needs about $2^{dc/2}$ super S-box computations to find the right pair.

Non-full-active Super S-box. As shown in Figure 4 (b), the non-full-active super S-box technique mainly exploits the following property of MDS:

Property 1. $\mathtt{MC} \cdot (Z[1], Z[2], \cdots, Z[d])^T = (W[1], W[2], \cdots W[d])^T$ can be used to fully determine the remaining unknowns if any d cells of Z, W are known.

Suppose there are s ($s < d$) non-active cells (with zero difference) and $2d - s$ active cells among ΔZ_1 and ΔW_1, by guessing the differences of $(d - s)$ active cells, differences of the rest d active cells can be obtained according to Property 1. Take the first column of Z_1 and W_1 in Figure 4 (b) as an example, with three non-active cells of $\Delta Z_1[1, 3]$ and $\Delta W_1[0]$, one cell $\Delta Z_1[0]$ can be guessed to obtain differences of $\Delta Z_1[2]$ and $\Delta W_1[1, 2, 3]$.

Then, given a fixed input-output difference of $\mathtt{SSB}_i$, for each guessing of the differences of $(d - s)$ active cells, all the input-output differences for the $(2d - s)$ active cells of two S-box layers can be obtained (marked by red cells in X_1, Y_1, X_2, Y_2). Thus, the input and output values of these $(2d - s)$ active S-boxes can be deduced by checking the differential distribution table ($\mathtt{DDT}$). Based on these values, for the equation $W_1 = \mathtt{MC}(Z_1)$, still according to Property 1, we have $(2d - s)$ known cells among W_1 and Z_1, which forms a $(d - s)$-cell filter with probability $2^{-(2d-s-d)c} = 2^{(s-d)c}$. Hence, for a fixed $(\Delta X_1^{(i)}, \Delta Y_2^{(i)})$, one can find $2^{(d-s)c} \cdot 2^{(s-d)c} = 1$ conforming pair on average for the inbound phase at a time complexity of $2^{(d-s)c}$ and a memory cost of 2^{2c} to store the $\mathtt{DDT}$. To give an example, we provide the pseudo-code of solving process of Figure 4 (b) in Appendix A.

In the quantum setting, Dong *et al.* [Don+20] converted the non-full-active super S-box technique into quantum by searching the $2^{(d-s)c}$ differences with Grover's algorithm, which gains a quadratic speedup. For both quantum and classical settings, the time complexity is determined by the parameter s.

3 CPC Attack Framework on `AES-like` Hashing

A CPC attack is the hardest among collision and its variants, as its setting forces the adversary to start with a random difference stemming from the chaining values (in `MMO/MP` modes, a random difference in the key). As illustrated in Figure 5, two random prefixes $(P, P)'$ result in random chaining variables $(H_1,\ H_1')$. Hence a random difference ΔH_1 is fed into the key schedule of the next block.

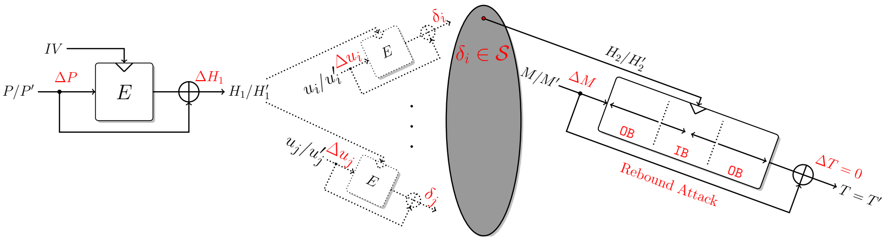


Figure 5: Related-key rebound-based collision framework on `AES-like` hashing

At Eurocrypt 2007, Stevens, Lenstra, and de Weger [SLW07] introduced the CPC attack framework on `MD5`. The attack starts with a birthday attack phase: it finds a pair of message blocks such that the resulting chaining value difference δ possess certain desirable properties. In said paper, the attack first find a $\delta = (\delta_a, \delta_b, \delta_c, \delta_d)$ with $\delta_a = 0$, and $\delta_b = \delta_c = \delta_d$). Then, the attacker constructs collision exploiting the property of δ with a second block or a succession of near-collision blocks using the message modification technique. The framework is later extended to allow more complex properties of δ and applied to find CPC attacks on `MD5` [Ste+09; SLW12] and on `SHA-1` [Ste13; LP19; LP20].

However, the attack framework is yet to be explored for `AES-like` hashing. This is mainly due to the strong diffusion of the wide-trail strategy in `AES-like` structures, which makes the message modification technique practiced on the `MD-SHA` family less effective. In this section, we propose the CPC attack framework with related-key rebound technique on `AES-like` hashing, as shown in Figure 5. We denote the compression function as CF, and our attack framework consists of three steps:

- Step 1: Find a class of related-key rebound attacks and a set $\mathcal{S}$, such that for any given input chaining value difference $\delta \in \mathcal{S}$, we are able to construct a free-start collision.

- Step 2: Starting from chaining values (H_1, H_1'), which is already fixed by the chosen prefixes (P, P'), find message blocks (u_i, u_i'), such that the output chaining value difference $\delta_i = CF(H_1, u_i) \oplus CF(H_1', u_i') \in \mathcal{S}$.

- Step 3: For this specific input chaining difference δ_i, perform the corresponding related-key rebound attack and finds a collision.

In Step 2, the time complexity of the birthday attack to find proper (u_i, u_i') is $\sqrt{2^n/|\mathcal{S}|}$ in both classical and quantum settings. In Step 3, we consider related-key rebound attacks in classical and quantum settings by assuming the probability of the outbound phase as p, and the DoF of the inbound phase should be larger than $1/p$:

- In the classical setting, we apply the birthday attack in step 2 to obtain a desired chaining value difference with time complexity $\sqrt{2^n/|\mathcal{S}|}$. For the inbound phase, assuming the time complexity to find one starting point is $\mathcal{T}^c_{\mathtt{IB}}$, the time complexity of the rebound attack is about $\mathcal{T}^c_{\mathtt{IB}}/p$. The overall classical attack will be $\sqrt{2^n/|\mathcal{S}|} + \mathcal{T}^c_{\mathtt{IB}}/p$.

- In the quantum setting, we use the birthday attack in step 2, which is valid under all QA, QB, QC assumptions and costs time $\sqrt{2^n/|\mathcal{S}|}$. we apply the 2-round super S-box or 3-round for the inbound phase. Assuming the quantum time to produce one starting point is $\mathcal{T}^q_{\mathtt{IB}}$, the final quantum attack will be $\sqrt{2^n/|\mathcal{S}|} + \mathcal{T}^q_{\mathtt{IB}}/\sqrt{p}$.

Implications on collision attacks on AES-like hashing. CPC attacks are backward compatible to collision attacks by simply setting $IV_0 = IV_1 = IV$. In the context of AES-like hashing, we discuss the implications of our CPC attack framework on collision attacks. On the one hand, previous collision and semi-free-start collision attacks on AES-like hashing can hardly leverage DoF of the difference of the chaining variables (or key of the block cipher E in MMO and MP modes). Taking Whirlpool as an example, the best differential based collision attacks are Lamberger *et al.*'s 5-round classical attack [LMRRS09] and 6-round quantum attack by Hosoyamada and Sasaki [HS20]. In both attacks, all DoF originates from the message blocks but not the key. At Asiacrypt 2020 [Don+20] and Crypto 2022 [DGLP22], Dong *et al.* converted the 7-round and 8-round semi-free-start collision attacks on AES-MMO/MP into quantum collisions. Though they succeeded in leveraging DoF from the values of the key, there is no difference in the key and the truncated differential used in the rebound attack is still single-key, *i.e.*, the DoF of the key differences is not utilized in both collision attacks. On the other hand, though the CPC framework provides a conversion from free-start collision to collision, the attack in previous work can hardly be utilized, as the DoF in key are often drained to compensate the probability of the outbound phase or to reach more rounds as seen in [Don+20; DGLP22]. The specific class of free-start collisions that balances the key DoF as well as the attack capability thus can be converted by the framework is yet to be explored.

Remark. We have developed an automatic search model for our CPC attack framework on AES-like hashing based on the related-key rebound models proposed by Dong *et al.* [Don+21b] at ASIACRYPT 2021. On top of that, we add more constraints to accurately evaluate the complexity of the birthday phase based on the DoF cost in key states, guaranteeing the obtained trail with enough DoF for the outbound, which improves Dong *et al.*'s model and removes many invalid trails. These will lead to a more complicated objective function aiming at balancing Step 2 and Step 3 mentioned above. Due to the limit space, we provide the model part in Appendix C and our codes at https://anonymous.4open.science/r/CPC_Rebound-515B.

4 Improved Quantum Algorithm to Solve 3-round Inbound

4.1 Inbound Phase with Three Full-active Rounds

Follow the notation that the state of the target cipher is of $d \times d$ c-bit cells. Take Whirlpool as an example, as shown in Figure 6, given fixed differences ΔZ_0 and ΔW_3, Jean *et al.* [JNP12] introduced an algorithm to find conforming pairs for this 3-round differential. At Eurocrypt 2020, Hosoyamada and Sasaki introduced a memoryless variant [HS20, Section A] (ePrint version) with the time complexity of $2^{d^2c/2+dc}$ to output one conforming pair under the classical setting, they also introduced the quantum variant [HS20, Section 7]

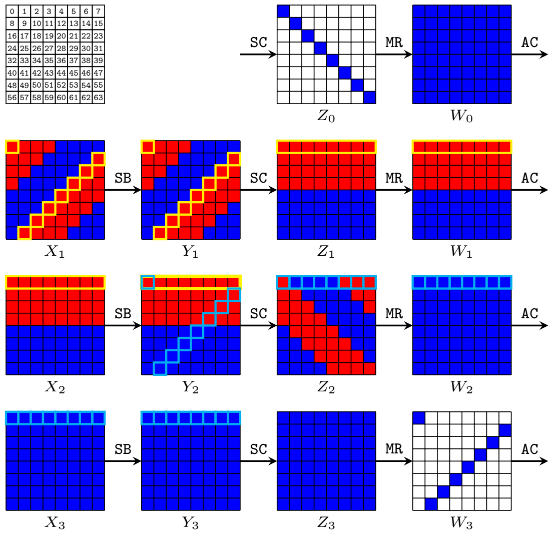


Figure 6: Inbound phase covering 3 full-active rounds

by applying Grover's algorithm to achieve a quadratic speedup, with the time complexity of $2^{d^2c/4+dc/2}$.

Jean *et al.*'s method to compute 3-round inbound [JNP12]. As depicted in Figure 6, ΔX_1 can be linearly computed from ΔZ_0. Similar to super S-box technique, one computes d super S-boxes $\texttt{SSB}^f_j$ $(0 \le j < d)$ covering from X_1 to Y_2. Each $\texttt{SSB}^f_j$ contains 2^{dc} pairs with fixed input differences marked by yellow box in Figure 6. Similarly, the attacker computes d inverse super S-boxes $\texttt{SSB}^b_j$ $(0 \le j < d)$ from Y_3 to Y_2.

One first focuses on a half of the super S-boxes $\texttt{SSB}^f_0, \texttt{SSB}^f_1, ..., \texttt{SSB}^f_{d/2-1}$ (red cells in X_1), which compute to $d^2/2$ red cells at Y_2. For $2^{d^2c/2}$ pairs of red cell values at Y_2, one checks if those values can be produced from $\texttt{SSB}^b_j$. As in each $\texttt{SSB}^b_j$, d-cell values have been fixed due to the pair of red cell values at Y_2, which acts as a dc-bit filter. Since the degrees of freedom in each $\texttt{SSB}^b_j$ is 2^{dc}, one match[2] on average for each $\texttt{SSB}^b_j$ is expected. Now, the pair at Y_2 is fully fixed.

Then, one computes the pair backward to X_1 to verify if the pair conforms to the fixed difference ΔX_1, and the differences of blue cells at X_1 acts as a filter of $2^{-d^2c/2}$. Hence, by exhaustively searching of the $2^{d^2c/2}$ values of $\texttt{SSB}^f_0, \texttt{SSB}^f_1, ..., \texttt{SSB}^f_{d/2-1}$ mentioned above, one expects to find a solution for this 3-round inbound phase, which conforms to the fixed input and output differences. The overall time complexity[3] is bounded by the match in the middle, which is about $2^{d^2c/2}$. The memory is $2d \cdot 2^{dc}$ to store the super S-boxes. For a better understanding, we provide the pseudo-code in Algorithm 8 in Appendix B.

Hosoyamada and Sasaki's memoryless variant and quantum variant [HS20]. Given input and output differences ΔX_1 and ΔY_3, Hosoyamada and Sasaki's memoryless variant is composed of two steps:

- Step I: Without storing the super S-boxes, Hosoyamada and Sasaki exhaustively search for $2^{d^2c/2}$ values of X_1[🟥], then compute Z_2[🟥] and Z'_2[🟥] by X_1[🟥] and $X'_1[🟥] = \Delta X_1[🟥] \oplus X_1[🟥]$, respectively.

[2] The match between a pair of values in red cells at Y_2 and $\texttt{SSB}^b_j$.

[3] The factor d is ignored when evaluating by round or reduced operation.

- Step II: For each row in Z_2, exhaustively search 2^{dc} values of Z_2[🟦] and Z'_2[🟦], together with Z_2[🟥] and Z'_2[🟥] derived in Step I, compute forward to the corresponding row of Y_3 and check if the difference equals to the given ΔY_3.

The total time complexity is around $2^{d^2c/2+dc}$ with negligible memory. Then, Hosoyamada and Sasaki converted this memoryless variant into a quantum variant by applying Grover's algorithm with around $2^{d^2c/4+dc/2}$ time complexity, achieving a quadratic speedup.

4.2 Our Improved Algorithm

Now we introduce an improved memoryless algorithm for solving the 3-round inbound phase. The difference between our algorithm and Hosoyamada-Sasaki's is the Step II. Instead of exhaustively searching for 2^{dc} values of Z_2[🟦] and Z'_2[🟦], we only exhaustively search for $2^{dc/2}$ values of Z_2[🟦] for each row. The detailed steps are given in Algorithm 1.

Algorithm 1: Improved memoryless classic algorithm to solve 3-round inbound

```
   Input: $\Delta Z_0$ and $\Delta W_3$
   Output: $(Z_0, Z'_0)$ and $(W_3, W'_3)$, with $\Delta Z_0 = Z_0 \oplus Z'_0$ and $\Delta W_3 = W_3 \oplus W'_3$
 1 Compute $\Delta X_1$ from $\Delta Z_0$ and $\Delta Y_3$ from $\Delta W_3$;
 2 for $2^{d^2c/2}$ values $X_1$[🟥] in Figure 6 do
 3     Compute $X'_1$[🟥] $= \Delta X_1$[🟥] $\oplus X_1$[🟥];
 4     Compute forward to get $Z_2$[🟥] and $Z'_2$[🟥] ;
 5     for row $j \in 0, 1, 2, ..., d-1$ in $Z_2$ do
 6         for at row $j$: $2^{dc/2}$ values of $Z_2$[🟦] do
 7             Together with $Z_2$[🟥], compute forward to get row $j$ of $Y_3$;
 8             Compute row $j$ of $Y'_3 = \Delta Y_3 \oplus Y_3$;
 9             Compute backward to get row $j$ of $Z'_2$ by $Y'_3$;
10             if row $j$ of $Z'_2$[🟥] in Line 9 is not equal to that in Line 4 then
11                 go to Line 6;
12     In this step, all cells of $Z_2, Z'_2$ are fixed, compute backward to get $\Delta X_1$;
13     if $\Delta X_1$[🟦] computed from $Z_2, Z'_2$ is equal to that computed from $\Delta Z_0$ computed
        in Line 1 then
14         return the pair as output;
```

In Line 6, only $2^{dc/2}$ values of Z_2[🟦] are exhausted for each row. Therefore, the total time complexity of Algorithm 1 is around $2^{d^2c/2+dc/2}$ with negligible memory under the classical setting, which is expected to find one conforming pair for this 3-round inbound phase. Then by applying Grover's algorithm, the time complexity of its quantum variant is about $2^{d^2c/4+dc/4}$. The detailed quantum variant will also be demonstrated in the CPC attack on 6-round `Whirlpool` in Section 5.1.

Remark. For `Whirlpool`, since both the 6-round quantum collision attack [HS20] and the 9-round quantum free-start collision attack [Don+21b] are based on Hosoyamada and Sasaki's memoryless algorithm [HS20] to solve the 3-round inbound part, these two collision attacks can be immediately improved by Algorithm 1, for which the detailed complexities are provided in Section 5.2.

5 Application to Round-Reduced Whirlpool

Whirlpool is a block-cipher-based hash function with a 512-bit hash value, which was designed by Rijmen and Barreto [BR00] as a submission to the NESSIE competition and later adopted as an ISO/IEC standard [ISO04]. Whirlpool adopts a 10 round AES-like block cipher with 8×8 byte encryption and key states in MP mode. Its encryption and key schedule essentially use the same round function with SB, SC, MR and AK operations, except for the key schedule replaced with the round constant addition AC, as illustrated in Figure 8. For more details, we refer the readers to the design paper [BR00].

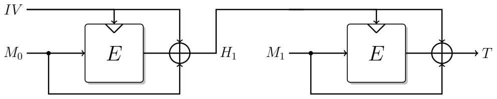


Figure 7: Two blocks of Whirlpool

Before we dive into the details, we clarify the following notions. We specify the input and output of quantum algorithms in the format of $|c_0, c_1, \ldots, c_m; x\rangle\, |y\rangle$, where x and y denote the input and output, and $c_0, c_1, \ldots, c_m$ denotes the set of constants involved. We use C and C' to denote the lists of round keys generated by the key pairs. C_r and C'_r denote the round key of round r. Furthermore, we denote the j-th row of state C_r as $C_r^{(j)}$. This notion is used similarly for other intermediate states (*e.g.*, X, Y, Z, A, B). And the notion of [🟥] denotes the subset of bytes marked red, similarly [🟦] marks blue, and [▨] marks gray.

5.1 Quantum CPC Attack on 6-round Whirlpool

With a two-block Whirlpool depicted in Figure 7, our quantum CPC attack on Whirlpool consists of the following two steps:

1. The first step is to find two message blocks (M_0, M'_0), such that the output pair of chaining values (H_1, H'_1) satisfies that ΔC_3[▨] $\neq 0$ and the other bytes of ΔC_3 in key states are zero differences as shown in Figure 8.

2. The second step is to find a message pair of (M_1, M'_1) by the rebound attack, so that the differences cancel at the target T, thus leading to a collision.

In the first step, such pair (M_0, M'_0) can be found with the birthday attack with time complexity $2^{8\cdot(64-16)/2} = 2^{192}$ and with negligible memory [Flo67]. We thus focus on illustrating the second step (or the rebound phase), which is as shown in Figure 8.

Probability and degree of freedom. The inbound phase covers from X_1 to Z_3. For the outbound phase, The probability of canceling the active differences in ΔY_5 to obtain ΔZ_5 is 2^{-128}, which utilizes the equivalent key addition and the feed-forward state ΔX_0 with zero difference. Since the number of choices for ΔZ_3 is 2^{128}, we have enough DoF to find one collision.

Improved memoryless quantum variant to solve 3-round inbound. Following Algorithm 1, we detail the improved quantum variant $\mathcal{U}_G$ to solve 3-round inbound in Algorithm 3, which marks the compatible 🟥 cells of (X_1, X'_1) in Figure 8 for a given input difference ΔX_1 and output difference ΔZ_3. We first define boolean functions $g^{(j)}(C_2, C'_2, \Delta Z_3, Y_2[🟥], Y'_2[🟥]; \cdot) : \mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$ for each row index $j \in [0, 7]$. When $j = 0$, $g^{(0)}(C_2, C'_2, \Delta Z_3, Y_2[🟥], Y'_2[🟥]; Y_2^{(0)}[🟦]) = 1$ if and only if $\mathtt{SB}(\mathtt{MR}(C_2^{(0)} \oplus Y_2^{(0)})) \oplus \mathtt{SB}(\mathtt{MR}(C_2'^{(0)} \oplus Y_2'^{(0)})) = \mathtt{SC}^{-1}(\Delta Y_3^{(0)})$. The quantum algorithm of the Grover search on $g^{(0)}$ is implemented

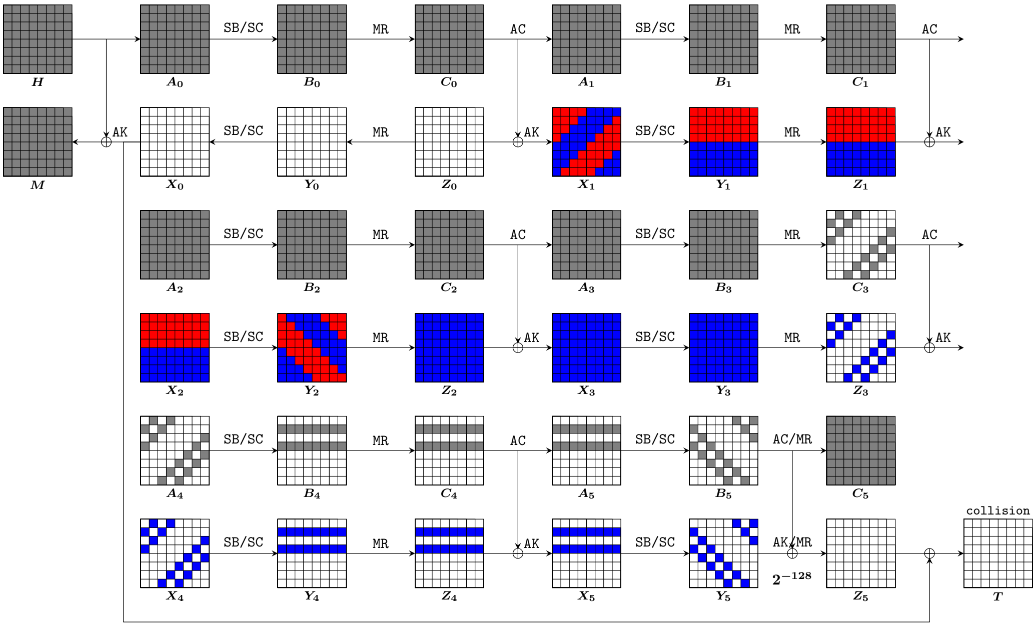


Figure 8: Quantum rebound attack on 6-round `Whirlpool`

as $\mathcal{U}_{g^{(0)}}$ and given in Algorithm 2. Similarly, for $j \in (0,7]$, we can define $g^{(j)}$ and implement $\mathcal{U}_{g^{(j)}}$. Then in $\mathcal{U}_G$, we call all $\mathcal{U}_{g^{(j)}}$ for $j \in [0,7]$ to solve the 3-round inbound and generate starting point for the outbound phase when the input and output differences are given.

Algorithm 2: Implementation of $\mathcal{U}_{g^{(0)}}$ without using qRAMs

```
Input: $|C_2, C_2', \Delta Z_3, Y_2[\text{🟥}], Y_2'[\text{🟥}]; Y_2^{(0)}[\text{🟦}]\rangle\,|y\rangle$
Output: $|C_2, C_2', \Delta Z_3, Y_2[\text{🟥}], Y_2'[\text{🟥}]; Y_2^{(0)}[\text{🟦}]\rangle\,|y \oplus g^{(0)}(C_2, C_2', \Delta Z_3, Y_2[\text{🟥}], Y_2'[\text{🟥}]; Y_2^{(0)}[\text{🟦}])\rangle$
1 Compute $\Delta Y_3$ from $\Delta Z_3$;
2 Compute $\mathtt{SC}^{-1}(Y_3^{(0)}) = \mathtt{SB}(C_2^{(0)}) \oplus \mathtt{MR}(Y_2^{(0)})$;
3 Compute $\mathtt{SC}^{-1}(Y_3'^{(0)}) = \mathtt{SC}^{-1}(Y_3^{(0)}) \oplus \mathtt{SC}^{-1}(\Delta Y_3^{(0)})$;
4 Compute $Y_2''^{(0)} = \mathtt{MR}^{-1}(\mathtt{SB}^{-1}(\mathtt{SC}^{-1}(Y_3'^{(0)})) \oplus C_2^{(0)})$;
5 if $Y_2''^{(0)}[\text{🟥}] \oplus Y_2'^{(0)}[\text{🟥}] = 0$ then
6     return $|C_2, C_2', \Delta Z_3, Y_2[\text{🟥}], Y_2'[\text{🟥}]; Y_2^{(0)}[\text{🟦}]\rangle\,|y \oplus 1\rangle$;
7 else
8     return $|C_2, C_2', \Delta Z_3, Y_2[\text{🟥}], Y_2'[\text{🟥}]; Y_2^{(0)}[\text{🟦}]\rangle\,|y\rangle$;
9 Uncompute from Line 1 to 4;
```

Complexity of $\mathcal{U}_G$. In Algorithm 2, Line 1-4 takes 16 S-box evaluations, as a pair of 4-byte values are computed with 2 S-box operations. Given $(\Delta Z_3, Y_2^{(j)}[\text{🟥}], Y_2'^{(j)}[\text{🟥}])$, $Y_2^{(j)}[\text{🟦}]$ satisfies $g^{(j)}$ with probability 2^{-32}. Hence, taking uncomputation into account, Algorithm 2 runs in time approximately as:

$$T_g \approx \frac{\pi}{4} \cdot 2^{32/2} \cdot (2 \cdot 16) \approx 2^{20.65} \text{ S-box evaluations.}$$

In Algorithm 3, the computation of all $Y_2'^{(j)}$ (Line 7) for $j \in [0,7]$ takes $16 \cdot 8 = 128$ S-box

Algorithm 3: Implementation of $\mathcal{U}_G$ without using qRAMs

```
   Input: $|C, C', \Delta Z_3; X_1[🟥]\rangle\,|y\rangle$
   Output: $|C, C', \Delta Z_3; X_1[🟥]\rangle\,|y \oplus G(C, C', \Delta Z_3; X_1[🟥])\rangle$
1  Get the constant input difference to the inbound part $\Delta X_1 = \Delta C_0$;
2  Compute $X'_1[🟥] = \Delta X_1[🟥] \oplus X_1[🟥]$;
3  Compute $Y_2[🟥], Y'_2[🟥]$ from $X_1[🟥], X'_1[🟥]$;
4  for $j \in [0, 7]$ do
5      Run the Grover search on $g^{(j)}(C_2, C'_2, \Delta Z_3, Y_2[🟥], Y'_2[🟥]; \cdot) : \mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$, get
        output $Y_2^{(j)}[🟦]$;
6      // Now we have the whole $Y_2^{(j)}$
7      Compute $Y_2'^{(j)}[🟦]$ (i.e., Line 2 to 4 of Algorithm 2);
8  // Now we have the whole $Y_2$ and $Y'_2$.
9  Compute $X_1[🟦], X'_1[🟦]$ from $Y_2, Y'_2$;
10 if $X_1[🟦] \oplus X'_1[🟦] = \Delta X_1[🟦]$ then
11     return $|C, C', \Delta Z_3; X_1[🟥]\rangle\,|y \oplus 1\rangle$;
12 else
13     return $|C, C', \Delta Z_3; X_1[🟥]\rangle\,|y\rangle$;
14 Uncompute from Line 1 to 9;
```

evaluations. In the computation of $X_1[🟦]$ and $X'_1[🟦]$ (Line 9), as a pair of 32-byte values are computed with 2 S-box operations, this step takes $32 \cdot 2 \cdot 2 = 128$ S-box evaluations. Given $(\Delta X_1, Z_3)$, $X_1[🟥]$ satisfies G with probability 2^{-256}. Hence, taking uncomputation into account (note that the uncomputation of $\mathcal{U}_{g^{(j)}}$ is included in T_g), Algorithm 3 runs in time approximately equivalent to:

$$T_G \approx \frac{\pi}{4} \cdot 2^{256/2} \cdot (8 \cdot T_g + 2 \cdot (128 + 128)) \approx 2^{151.3} \text{ S-box evaluations.}$$

Algorithm 4: Implementation of $\mathcal{U}_f$ without using qRAMs

```
   Input: $|C, C'; \Delta Z_3\rangle\,|y\rangle$
   Output: $|C, C'; \Delta Z_3\rangle\,|y \oplus f(C, C'; \Delta Z_3)\rangle$
1  Run the Grover search on $G(C, C', \Delta Z_3; \cdot) : \mathbb{F}_2^{32\times 8} \mapsto \mathbb{F}_2$, get output $X_1[🟥]$;
2  Compute $X_1, X'_1$ (i.e., run Line 1-9 of Algorithm 3);
3  Compute $T, T'$ from $X_1, X'_1$;
4  if $T \oplus T' = 0$ then
5      return $|C, C'; \Delta Z_3\rangle\,|y \oplus 1\rangle$;
6  else
7      return $|C, C'; \Delta Z_3\rangle\,|y\rangle$;
8  Uncompute from Line 1 to 3;
```

Implementation of the quantum CPC attack. We provide a detailed quantum algorithm $\mathcal{U}_f$ to perform the quantum CPC attack on 6-round `Whirlpool` and its implementation in Algorithm 4. The complexity of $\mathcal{U}_f$ is evaluated as follows. The recovery of the full X_1, X'_1 (Line 2) takes about 1 iteration of $\mathcal{U}_G$, which runs in time approximately equivalent to $8 \cdot T_g$ S-box evaluations (uncomputation included). And the computation of T and T' (Line 3) needs 2 computations with 6-round `Whirlpool`. Note that one computation with 6-round Whirlpool needs $6 \times 64 \times 2 = 768$ S-box evaluations. Given the outbound probability is 2^{-128}, taking uncomputation into account (note that the uncomputation of $\mathcal{U}_G$ is included in T_G), the total complexity of the rebound phase is thus:

$$T_f \approx \frac{\pi}{4} \cdot 2^{128/2} \cdot ((T_G + 8 \cdot T_g)/768 + 2 \cdot 2) \approx 2^{205.36}.$$

The complexity of the birthday attack finding a pair of desired (C, C') is $2^{(512-128)/2} = 2^{192}$. Therefore, the final time complexity of the quantum CPC attack on 6-round Whirlpool is about $T_f + 2^{192} \approx 2^{205.4}$.

5.2 Improved Quantum Attacks on Whirlpool

For Whirlpool, with our improved method of solving 3-round inbound part (Algorithm 1 and Algorithm 3), we further improve Hosoyamada and Sasaki's 6-round collision attack [HS20] and Dong *et al.*'s 9-round quantum free-start collision attack [Don+21b].

Improved 6-round collision attack on Whirlpool. We reuse the 6-round truncated differential proposed by Hosoyamada and Sasaki [HS20], which includes a 3-round inbound part (Figure 6) and the rest outbound part. The probability of the outbound part is 2^{-120}. Similar to Section 8, the time complexity of the 6-round collision attack is thus

$$\frac{\pi}{4} \cdot 2^{120/2} \cdot T_G \approx 2^{210.95} \text{ S-box evaluations.} \tag{1}$$

which is about $2^{210.95}/768 = 2^{201.4}$ computations with 6-round Whirlpool. We note that this improved attack is better than the CNS collision finding algorithm [CNS17].

Improved 9-round free-start collision attack on Whirlpool. We reuse the same 9-round truncated differential given by Dong *et al.* [Don+21b] and follow their quantum attack framework. We only replace our method to solve the 3-round inbound part. Therefore, the complexity equation from [Don+21b, Equation 27] (ePrint version) becomes:

$$\frac{\pi}{4} \cdot 2^{(64-8)/2} \cdot T_G + \frac{\pi}{4} \cdot 2^{64/2} \cdot T_G \approx 2^{183.04} \text{ S-box evaluations.} \tag{2}$$

The complexity equation from [Don+21b, Equation 28] (ePrint version) becomes:

$$\frac{\pi}{4} \cdot 2^{64/2} \cdot 2^{183.04} \approx 2^{214.7} \text{ S-box evaluations,} \tag{3}$$

which is about $2^{214.7}/1152 = 2^{204.53}$ 9-round compression functions of Whirlpool.

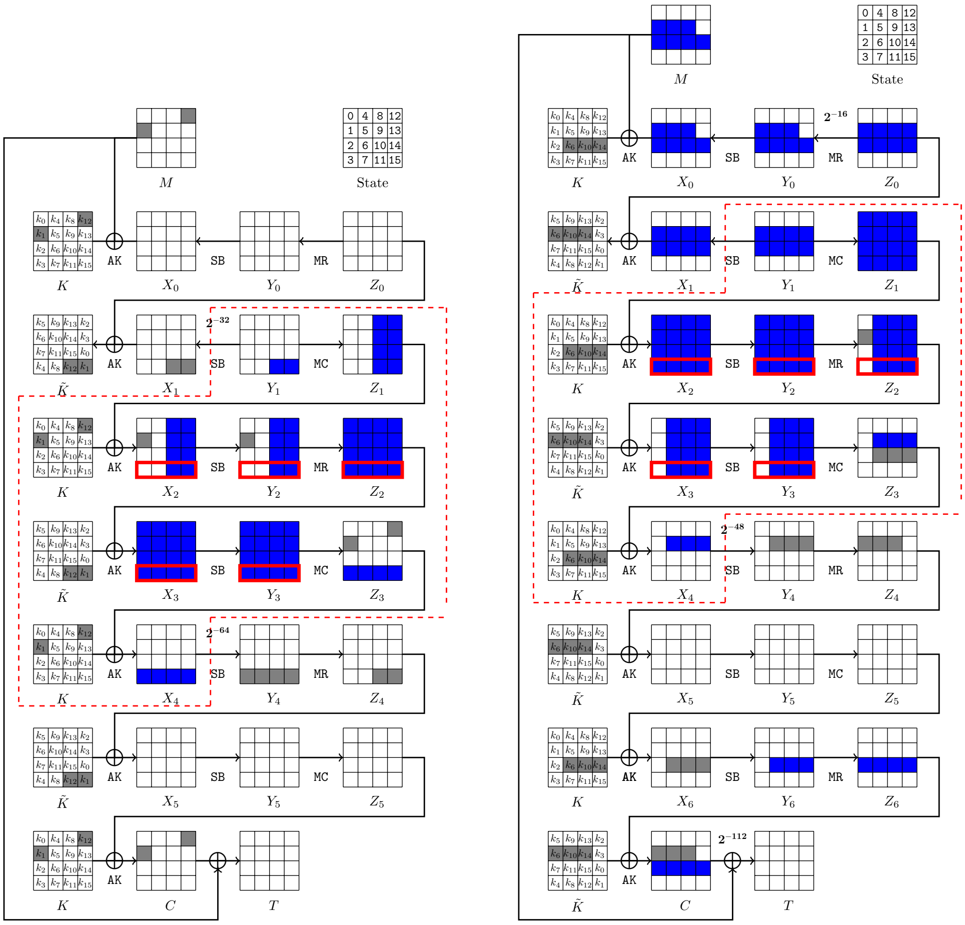
6 Application to Round-Reduced Saturnin-hash

Saturnin is a suite of lightweight symmetric algorithms proposed by Canteaut *et al.* [Can+20]. It is among the 2nd round candidates of the NIST LWC. Based on a 256-bit AES-like block cipher with the 256-bit key, two authenticated ciphers and a hash function are then designed. In this section, we focus on its hash function Saturnin-hash adopting MMO mode. The round function of Saturnin-hash consists of AK, SB layer and alternatively used linear layers MR (even round) and MC (odd round). Its key schedule is simple, the master key K is used in even rounds, and $\tilde{K}$ (rotating K by 5 cells) is used in odd rounds.

6.1 Classical CPC/Collision Attack on 6-round `Saturnin-hash`

We first provide CPC and collision attacks on 6-round **`Saturnin-hash`** under the classical setting, as shown in Figure 9 (a), any key pair with active cells in $K[1, 12]$ can be fed into our 6-round attacks. Therefore, the time complexity to find such a key pair is $2^{(256-32)/2} = 2^{112}$ by the birthday search from the first block.

When a proper key pair (K, K') is found, a rebound attack is then performed in the second block. In Figure 9 (a), the differences of ■ cells in key states are fixed given the found key pair (K, K'). The inbound phase covers from state Y_1 to X_4. The number of input and output differences $(\Delta Y_1, \Delta Z_3)$ of the inbound phase is $2^{16 \cdot 2 + 16 \cdot 4} = 2^{96}$. For the outbound phase, the probability is $2^{-32-64} = 2^{-96}$. Hence, the degree of freedom of the inbound phase is enough to find one collision. The 6-round classical rebound attack is given in Algorithm 5 with the time complexity of around $2^{32+64} = 2^{96}$ and memory complexity of 2^{64}. Finally, the overall time complexity of this 6-round CPC attack on `Saturnin-hash` is $2^{112} + 2^{96} \approx 2^{112}$ and with 2^{64} memory complexity.



(a) Classical rebound attack on 6-round `Saturnin-hash`.

(b) Quantum rebound attack on 7-round `Saturnin-hash`.

Figure 9: Rebound attacks on `Saturnin-hash`.

Algorithm 5: Classical Rebound Attack on 6-round `Saturnin-hash`

```
Input: A key pair $(K, K')$
Output: A collision pair $(M, M')$
/* Note that the differences marked by ■ in $X_2$ and $Z_3$ are fixed due
   to the fixed key pair $(K, K')$ */
for $2^{32}$ differences of $\Delta Y_1$ do
    Compute $\Delta X_2$ from $\Delta Y_1$;
    /* Prepare 4 super S-boxes */
    for $i = 0, 1, 2, 3$ do
        $\mathbb{L}^{(i)} \leftarrow [\,]$;
        for $x \in \mathbb{F}_2^{64}$ do
            Compute $\Delta Y_3^{(i)} = \mathtt{SSB}_i(x \oplus \Delta X_2^{(i)}) \oplus \mathtt{SSB}_i(x)$;
            $\mathbb{L}^{(i)}[\Delta Y_3^{(i)}] \leftarrow (x, x \oplus \Delta X_2^{(i)})$;
    for $2^{64}$ differences of $\Delta Z_3$ do
        Compute $\Delta Y_3$ from $\Delta Z_3$;
        Lookup tables $\mathbb{L}^{(0)}[\Delta Y_3^{(0)}]$, $\mathbb{L}^{(1)}[\Delta Y_3^{(1)}]$, $\mathbb{L}^{(2)}[\Delta Y_3^{(2)}]$, and $\mathbb{L}^{(3)}[\Delta Y_3^{(3)}]$ to
          build the full state of $X_2$ and $X_2'$;
        if $(X_2, X_2')$ leads to a collision then
            Return $(M, M')$;
```

6.2 Quantum CPC Attack on 7-round `Saturnin-hash`

We now briefly provide a quantum CPC attack on 7-round `Saturnin-hash`, as shown in Figure 9 (b), we have the following conditions on the key pairs (*i.e.*, K, K'):

$$\begin{cases} \Delta\tilde{K}[1,5,9] = \Delta Z_4[1,5,9], \\ \mathtt{MR}^{-1}(\Delta Z_4[1,5,9], 0) = (0,*,*,*) = \Delta Y_4[1,5,9,13], \end{cases} \tag{4}$$

which consumes 1-cell DoF of the difference in key states and leads to 2-cell DoF of the differences for the target key pair. Therefore, with the birthday attack, the first block can produce a key pair (K, K') satisfying the condition of Equation 4 with the time complexity $2^{(256-2\cdot 16)/2} = 2^{112}$.

With the found key pair (K, K') above, ΔY_4, $\Delta Z_3[6,10,14]$ and $\Delta Z_2[1]$ are then determined. The probability of the outbound phase is $2^{-112-48-16} = 2^{-176}$. For the inbound phase, the degree of the freedom of the input and output differences $(\Delta Y_1, \Delta Z_3)$ is $2^{128+48} = 2^{176}$. Therefore, there is enough degree of freedom to produce one collision. A detailed analysis of this quantum CPC attack on 7-round `Saturnin-hash` is provided in Appendix F.

7 Application to Round-Reduced `AES-MMO/MP`

In this section, due to the limited space, we only provide the CPC attack on 5-round `AES-MP` under the classical setting, which slightly improves the time complexity of the 5-round collision attack on `AES-MP` from 2^{56} to 2^{52}. The details of the CPC attack on 5-round `AES-MMO` (classical) and 6-round `AES-MMO/MP` (quantum) are provided in Appendix D and Appendix E, respectively.

For the CPC attack on 5-round `AES-MP`, as illustrated in Figure 10, any key pair with active cells in $K_4[3,8,12]$ can be applied to this 5-round attack. Therefore, the time complexity to find such a key pair is $2^{(128-24)/2} = 2^{52}$ by the birthday search from the first

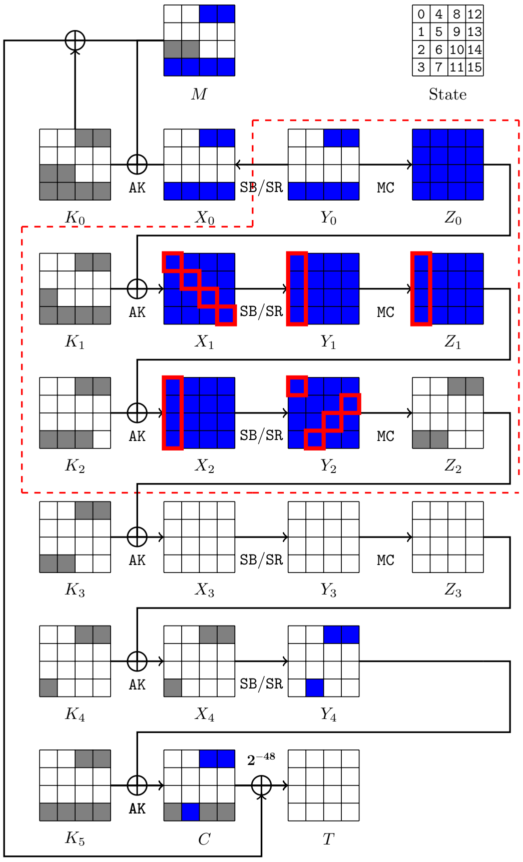


Figure 10: Classical rebound attack on 5-round `AES-MP`

```
Algorithm 6: Classical Rebound Attack on 5-round AES-MP
Input: A key pair $(K, K')$
Output: A collision pair $(M, M')$
/* Note that $\Delta Z_2$ is fixed to ■ cells, we start to build the super
   S-box from backward                                                   */
Compute $\Delta Y_2 = \mathtt{MC}^{-1}(\Delta Z_2)$;
for $i = 0, 1, 2, 3$ do
    $\mathbb{L}^{(i)} \leftarrow [\,]$;
    for $x \in \mathbb{F}_2^{32}$ do
        Compute $\Delta X_1^{(i)} = \mathtt{SSB}_i^{-1}(x \oplus \Delta Y_2^{(i)}) \oplus \mathtt{SSB}_i^{-1}(x)$;
        $\mathbb{L}^{(i)}[\Delta X_1^{(i)}] \leftarrow (x, x \oplus \Delta Y_2^{(i)})$;
for $2^{48}$ differences of $\Delta Y_0$ do
    Obtain the full states of $Y_2$ and $Y_2'$ by accessing tables $\mathbb{L}^{(i)}$;
    if $(Y_2, Y_2')$ leads to a collision then
        Return $(M, M')$;
```

block. When a proper key pair (K, K') is found, the rebound attack is performed in the second block. In Figure 10, the inbound phase also covers from state Y_0 to Z_2. The number of input and output differences $(\Delta Y_0, \Delta Z_2)$ of the inbound phase is $2^{8 \cdot 6} = 2^{48}$. In the outbound phase, the probability of the differential is 2^{-48}. Hence, one collision is expected from this rebound attack. Details of this classical rebound attack on 5-round AES-MP is given in Algorithm 6 with the time complexity of around 2^{48} and memory complexity of 2^{32}. Thus, the overall time complexity of our 5-round CPC attack on AES-MP is $2^{52} + 2^{48} \approx 2^{52}$ with 2^{32} memory complexity.

8 Conclusion

In this work, we explore CPC attacks on AES-like hashing. Our attack follows the CPC attack framework on MD-SHA family while replaces the message modification technique on the second block with a related-key rebound attack. We present the first CPC attacks on reduced Whirlpool, Saturnin-hash and AES-MMO/MP in classic and quantum settings. From the results, we have spotted the difference on MMO and MP modes in related-key rebound attacks, which are generally believed to be equivalent.

The framework can be used to convert a specific class of free-start collision attacks into a collision attack. However, there must be sufficient DoF in the difference of the chaining values, of which current free-start collision attacks lacks. As proof of concept, we improve the classical collision attack on Saturnin-hash from 5 to 6 rounds.

As an independent contribution, we improve the quantum algorithm for solving the 3-round inbound phase, which has resulted in the first 6-round memoryless quantum attack on Whirlpool better than generic CNS collision finding algorithm when exponential-size qRAM is not available but exponential-size classic memory is available.

Acknowledgments

We would like to thank all the anonymous reviewers of ToSC for their valuable comments and suggestions, and specially thank our shepherd for helping us improve the quality of this paper. This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative, the Natural Science Foundation of China (62272257) and the Ministry of Education in Singapore under Grant RG93/23. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

References

[AGL22] Akshima, Siyao Guo, and Qipeng Liu. “Time-Space Lower Bounds for Finding Collisions in Merkle-Damgård Hash Functions”. In: *CRYPTO*. 2022, pp. 192–221.

[All17] ZigBee Alliance. *zigbee Specification Revision 22 1.0*. Tech. rep. ZigBee Alliance, Apr. 2017.

[BDPA13] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. “Keccak”. In: *EUROCRYPT*. 2013, pp. 313–314.

[Ber09] Daniel J Bernstein. “Cost analysis of hash collisions: Will quantum computers make SHARCS obsolete”. In: *SHARCS* 9 (2009), p. 105.

[BHNSS19] Xavier Bonnetain, Akinori Hosoyamada, Marıa Naya-Plasencia, Yu Sasaki, and André Schrottenloher. “Quantum Attacks Without Superposition Queries: The Offline Simon’s Algorithm”. In: *ASIACRYPT*. 2019, pp. 552–583.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. “Quantum Cryptanalysis of Hash and Claw-Free Functions”. In: *LATIN*. 1998, pp. 163–169.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent. “Transcript Collision Attacks: Breaking Authentication in TLS, IKE and SSH”. In: *23rd Annual Network and Distributed System Security Symposium, NDSS*. 2016.

[BLNS21] Xavier Bonnetain, Gaëtan Leurent, Marıa Naya-Plasencia, and André Schrottenloher. “Quantum Linearization Attacks”. In: *ASIACRYPT*. Ed. by Mehdi Tibouchi and Huaxiong Wang. 2021, pp. 422–452.

[BR00] Paulo S.L.M. Barreto and Vincent Rijmen. “The WHIRLPOOL Hashing Function”. In: *Submitted to NESSIE* (2000). URL: http://www.karljapetre.com/whirlpool/whirlpool.pdf.

[Can+20] Anne Canteaut, Sébastien Duval, Gaëtan Leurent, Marıa Naya-Plasencia, Léo Perrin, Thomas Pornin, and André Schrottenloher. “Saturnin: a suite of lightweight symmetric algorithms for post-quantum security”. In: *IACR Trans. Symmetric Cryptol.* 2020.S1 (2020), pp. 160–207.

[CGLSZ24] Shiyao Chen, Jian Guo, Eik List, Danping Shi, and Tianyu Zhang. “Diving Deep into the Preimage Security of AES-Like Hashing”. In: *EUROCRYPT*. 2024, pp. 398–426.

[CHPSS17] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. “A Security Analysis of Deoxys and its Internal Tweakable Block Ciphers”. In: *IACR Trans. Symmetric Cryptol.* 2017.3 (2017), pp. 73–107.

[CNS17] André Chailloux, Marıa Naya-Plasencia, and André Schrottenloher. “An Efficient Quantum Collision Search Algorithm and Implications on Symmetric Cryptography”. In: *ASIACRYPT*. 2017, pp. 211–240.

[CR06] Christophe De Cannière and Christian Rechberger. “Finding SHA-1 Characteristics: General Results and Applications”. In: *ASIACRYPT*. 2006, pp. 1–20.

[Dam] Ivan Damgård. “A Design Principle for Hash Functions”. In: *CRYPTO ’89*. Ed. by Gilles Brassard. Vol. 435, pp. 416–427.

[DDKS12] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. “Efficient Dissection of Composite Problems, with Applications to Cryptanalysis, Knapsacks, and Combinatorial Search Problems”. In: *CRYPTO*. Vol. 7417. 2012, pp. 719–740.

[DGLP22] Xiaoyang Dong, Jian Guo, Shun Li, and Phuong Pham. “Triangulating Rebound Attack on AES-like Hashing”. In: *CRYPTO*. 2022, pp. 94–124.

[Don+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: *ASIACRYPT*. 2020, pp. 727–757.

[Don+21a] Xiaoyang Dong, Jialiang Hua, Siwei Sun, Zheng Li, Xiaoyun Wang, and Lei Hu. “Meet-in-the-Middle Attacks Revisited: Key-Recovery, Collision, and Preimage Attacks”. In: *CRYPTO*. 2021, pp. 278–308.

[Don+21b] Xiaoyang Dong, Zhiyu Zhang, Siwei Sun, Congming Wei, Xiaoyun Wang, and Lei Hu. “Automatic Classical and Quantum Rebound Attacks on AES-Like Hashing by Exploiting Related-Key Differentials”. In: *ASIACRYPT*. 2021, pp. 241–271.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[Fló+20] Antonio Flórez-Gutiérrez, Gaëtan Leurent, Marıa Naya-Plasencia, Léo Perrin, André Schrottenloher, and Ferdinand Sibleyras. “New Results on Gimli: Full-Permutation Distinguishers and Improved Collisions”. In: *ASIACRYPT 2020*. 2020, pp. 33–63.

[Flo67] Robert W. Floyd. “Nondeterministic Algorithms”. In: *J. ACM* 14.4 (1967), pp. 636–644.

[Gau+09] Praveen Gauravaram, Lars R. Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. “Grøstl - a SHA-3 candidate”. In: *Symmetric Cryptography*. 2009.

[GMD06] Praveen Gauravaram, Adrian McCullagh, and Edward Dawson. “Collision Attacks on MD5 and SHA-1: Is this the’Sword of Damocles’ for Electronic Commerce?” In: *Proceedings of the AusCERT Asia Pacific Information Technology Security Conference*. 2006, pp. 73–88.

[GP10] Henri Gilbert and Thomas Peyrin. “Super-Sbox Cryptanalysis: Improved Attacks for AES-Like Permutations”. In: 2010, pp. 365–383.

[Gro96] Lov K. Grover. “A Fast Quantum Mechanical Algorithm for Database Search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*. 1996, pp. 212–219.

[HI21] Akinori Hosoyamada and Tetsu Iwata. “On Tight Quantum Security of HMAC and NMAC in the Quantum Random Oracle Model”. In: *CRYPTO*. 2021, pp. 585–615.

[HS20] Akinori Hosoyamada and Yu Sasaki. “Finding Hash Collisions with Quantum Computers by Using Differential Trails with Smaller Probability than Birthday Bound”. In: *EUROCRYPT*. 2020, pp. 249–279.

[HS21] Akinori Hosoyamada and Yu Sasaki. “Quantum Collision Attacks on Reduced SHA-256 and SHA-512”. In: *CRYPTO*. 2021, pp. 616–646.

[HSX17] Akinori Hosoyamada, Yu Sasaki, and Keita Xagawa. “Quantum Multicollision-Finding Algorithm”. In: *ASIACRYPT*. 2017, pp. 179–210.

[ISO04] ISO/IEC. *ISO/IEC 10118-3: 2004. IT Security techniques - Hash-functions - Part 3: Dedicated hash-functions*. 2004.

[ISO10] ISO/IEC. *ISO/IEC 10118-2: 2010. IT Security techniques - Hash-functions - Part 2: Hash-functions using an n-bit block cipher*. 2010.

[JNP12] Jérémy Jean, Marıa Naya-Plasencia, and Thomas Peyrin. “Improved Rebound Attack on the Finalist Grøstl”. In: *FSE 2012*. 2012, pp. 110–126.

[KLLN16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and Marıa Naya-Plasencia. “Breaking Symmetric Cryptosystems Using Quantum Period Finding”. In: *CRYPTO*. 2016, pp. 207–237.

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

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

[LM17] Gregor Leander and Alexander May. “Grover Meets Simon - Quantumly Attacking the FX-construction”. In: *ASIACRYPT*. 2017, pp. 161–178.

[LMRRS09] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. “Rebound Distinguishers: Results on the Full Whirlpool Compression Function”. In: *ASIACRYPT*. 2009, pp. 126–143.

[LP19] Gaëtan Leurent and Thomas Peyrin. “From Collisions to Chosen-Prefix Collisions Application to Full SHA-1”. In: *EUROCRYPT*. 2019, pp. 527–555.

[LP20] Gaëtan Leurent and Thomas Peyrin. “SHA-1 is a Shambles: First Chosen-Prefix Collision on SHA-1 and Application to the PGP Web of Trust”. In: *USENIX Security Symposium*. 2020, pp. 1839–1856.

[LZ19] Qipeng Liu and Mark Zhandry. “On Finding Quantum Multi-collisions”. In: *EUROCRYPT*. 2019, pp. 189–218.

[Mer89] Ralph C. Merkle. “A Certified Digital Signature”. In: *CRYPTO*. 1989, pp. 218–238.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. “The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl”. In: *FSE*. Vol. 5665. 2009, pp. 260–276.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. “Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming”. In: *Inscrypt 2011*. 2011, pp. 57–76.

[Nay11] Marıa Naya-Plasencia. “How to Improve Rebound Attacks”. In: *CRYPTO 2011*. 2011, pp. 188–205.

[OW99] Paul C. van Oorschot and Michael J. Wiener. “Parallel Collision Search with Cryptanalytic Applications”. In: *J. Cryptol.* 12.1 (1999), pp. 1–28.

[PGV93] Bart Preneel, René Govaerts, and Joos Vandewalle. “Hash Functions Based on Block Ciphers: A Synthetic Approach”. In: *CRYPTO*. 1993, pp. 368–378.

[Riv92] Ronald L. Rivest. “The MD5 Message-Digest Algorithm”. In: *RFC* 1321 (1992), pp. 1–21.

[SLW07] Marc Stevens, Arjen K. Lenstra, and Benne de Weger. “Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities”. In: *EUROCRYPT*. Vol. 4515. 2007, pp. 1–22.

[SLW12] Marc Stevens, Arjen K. Lenstra, and Benne de Weger. “Chosen-prefix collisions for MD5 and applications”. In: *Int. J. Appl. Cryptogr.* 2.4 (2012), pp. 322–359.

[SLWSO10] Yu Sasaki, Yang Li, Lei Wang, Kazuo Sakiyama, and Kazuo Ohta. “Non-full-active Super-Sbox Analysis: Applications to ECHO and Grøstl”. In: *ASIACRYPT 2010*. 2010, pp. 38–55.

[SS24] André Schrottenloher and Marc Stevens. “Quantum Procedures for Nested Search Problems”. In: *IACR Communications in Cryptology* 1.3 (Oct. 7, 2024). ISSN: 3006-5496.

[ST15] National Institute of Standards and Technology. *FIPS 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. 2015.

[ST95] National Institute of Standards and Technology. *FIPS 180-1: Secure Hash Standard*. 1995.

[Ste+09] Marc Stevens, Alexander Sotirov, Jacob Appelbaum, Arjen K. Lenstra, David Molnar, Dag Arne Osvik, and Benne de Weger. “Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate”. In: *CRYPTO*. Vol. 5677. 2009, pp. 55–69.

[Ste13] Marc Stevens. "New Collision Attacks on SHA-1 Based on Optimal Joint Local-Collision Analysis". In: *EUROCRYPT*. 2013, pp. 245–261.

[SWWW12] Yu Sasaki, Lei Wang, Shuang Wu, and Wenling Wu. "Investigating Fundamental Security Requirements on Whirlpool: Improved Preimage and Collision Attacks". In: *ASIACRYPT*. 2012, pp. 562–579.

[TSITI24] Kodai Taiyama, Kosei Sakamoto, Ryoma Ito, Kazuma Taka, and Takanori Isobe. "Key Collisions on AES and Its Applications". In: *IACR Cryptol. ePrint Arch.* (2024), p. 1508. URL: https://eprint.iacr.org/2024/1508.

[WLFCY05] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. "Cryptanalysis of the Hash Functions MD4 and RIPEMD". In: *EUROCRYPT*. 2005, pp. 1–18.

[WY05] Xiaoyun Wang and Hongbo Yu. "How to Break MD5 and Other Hash Functions". In: *EUROCRYPT*. 2005, pp. 19–35.

[WYY05a] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. "Finding Collisions in the Full SHA-1". In: *CRYPTO*. 2005, pp. 17–36.

[WYY05b] Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. "Efficient Collision Search Attacks on SHA-0". In: *CRYPTO*. 2005, pp. 1–16.

A Example of Non-Full-Active Solving Algorithm

We provide the pseudo-code of solving non-full-active super S-box in Figure 4 (b) in Algorithm 7.

B Pseudocode of Solving for 3-round Inbound

We provide the pseudo-code of solving 3-round inbound phase in Figure 6 in Algorithm 8.

C Automatic Search Model of Related-key Rebound Collision

Here, we introduce the automatic model for our CPC attack framework on `AES-like` hashing, which is a unified automatic model covering Step 2 (birthday phase) and Step 3 (rebound phase). We first briefly revisit the model of truncated differential related-key differential by Mouha *et al.*'s model [MWGP11], and models of inbound and outbound phase proposed by Dong *et al.* [Don+21b]. Then the models of the birthday phase and the objective in terms of the attack time complexity are provided.

Related-key Truncated Differential Model. For an `Nr`-round `AES-like` structure as depicted in Figure 2, we introduce binary variables $x_r^{i,j}$, $y_r^{i,j}$, $z_r^{i,j}$, and $w_r^{i,j}$ to respectively denote the activeness of the c-bit cell at the i-th row and j-th column ($0 \leq i, j < d$) of input state to `SB`, `SR`, `MC` (resp. `MR`), and `AK` in the r-th round ($0 \leq r < \mathtt{Nr}$).

The constraints for `SB` and `SR` are straightforward as identity and circular shift. Following Mouha *et al.*'s model [MWGP11], the `MC` operation can be easily modelled by the branch number[4]. The constraint of key addition `AK` is similar to that of bit XOR operation, except that the addition of two active cells can be active or inactive. Now the model to describe the propagation of the truncated differential is built, one just needs to introduce

[4] The sum of input and output activeness indicators should be greater than the branch number or just equal to zero.

Algorithm 7: Non-full-active super S-box solving algorithm

```
Input: $\Delta Z_0$ and $\Delta W_2$
Output: $(Z_0, Z_0')$ and $(W_2, W_2')$, with $\Delta Z_0 = Z_0 \oplus Z_0'$ and $\Delta W_2 = W_2 \oplus W_2'$
Compute $\Delta X_1$ from $\Delta Z_0$ and $\Delta Y_2$ from $\Delta W_2$;
// Here, the parameters are $(d, s, c) = (4, 3, 8)$
// column $i = 0$, SSB$_0$ of state $Z_1$, 1 conforming pair on average
for $\Delta Z_1[0] \in \mathbb{F}_{2^8}$ do
    Compute $\Delta Z_1[2]$ and $\Delta W_1[1, 2, 3]$ according to Property 1;
    Derive $\Delta Y_1[0, 10] = \Delta Z_1[0, 2]$, and $\Delta X_2[1, 2, 3] = \Delta W_1[1, 2, 3]$;
    Deduce input and output pairs $(X_1[0, 10], X_1'[0, 10])$ and $(Y_1[0, 10], Y_1'[0, 10])$
      from $(\Delta X_1[0, 10], \Delta Y_1[0, 10])$ according to DDT;
    Deduce input and output pairs $(X_2[1, 2, 3], X_2'[1, 2, 3])$ and
      $(Y_2[1, 2, 3], Y_2'[1, 2, 3])$ from $(\Delta X_2[1, 2, 3], \Delta Y_2[1, 2, 3])$ according to DDT;
    Obtain $(Z_1[0, 2], Z_1'[0, 2])$ and $(W_1[1, 2, 3], W_1'[1, 2, 3])$;
    Check $Z_1[0, 2]$ and $W_1[1, 2, 3]$ through MC (with probability $2^{-8}$), then compute
      $(Z_1[1, 3], Z_1'[1, 3])$ and $(W_1[0], W_1'[0])$;
  ⋮
// column $i = 3$, SSB$_3$ of state $Z_1$, 1 conforming pair on average
for $\Delta Z_1[15] \in \mathbb{F}_{2^8}$ do
    Compute $\Delta Z_1[13]$ and $\Delta W_1[12, 13, 14]$ according to Property 1;
    Derive $\Delta Y_1[1, 11] = \Delta Z_1[13, 15]$, and $\Delta X_2[12, 13, 14] = \Delta W_1[12, 13, 14]$;
    Deduce input and output pairs $(X_1[1, 11], X_1'[1, 11])$ and $(Y_1[1, 11], Y_1'[1, 11])$
      from $(\Delta X_1[1, 11], \Delta Y_1[1, 11])$ according to DDT;
    Deduce input and output pairs $(X_2[12, 13, 14], X_2'[12, 13, 14])$ and
      $(Y_2[12, 13, 14], Y_2'[12, 13, 14])$ from $(\Delta X_2[12, 13, 14], \Delta Y_2[12, 13, 14])$ according
      to DDT;
    Obtain $(Z_1[13, 15], Z_1'[13, 15])$ and $(W_1[12, 13, 14], W_1'[12, 13, 14])$;
    Check $Z_1[13, 15]$ and $W_1[12, 13, 14]$ through MC (with probability $2^{-8}$), then
      compute $(Z_1[12, 14], Z_1'[12, 14])$ and $(W_1[15], W_1'[15])$;
Derive $(Z_0, Z_0')$ and $(W_2, W_2')$ from known $(X_1, X_1')$ and $(Y_2, Y_2')$;
return $(Z_0, Z_0')$ and $(W_2, W_2')$;
// Time complexity: $2^{(d-s)c} = 2^8$ computations of the inbound
// Memory complexity: $2^{2c} = 2^{16}$ for DDT
```

some extra variables to trace the consumption of DoF when passing MC and AK operations, and finally optimizes the objective related to such DoF costs.

This model works well under the single-key setting, however, as observed by Cid *et al.* [CHPSS17] and Dong *et al.* [Don+21b], this simple model may lead to a lower probability evaluation than the reality under the related-key setting, which is caused by compensating the cost of the DoF in the encryption for the extra cost in the key schedule through the key addition. To evaluate such extra DoF compensation in the key schedule is very important to the overall time complexity due to the birthday phase in Step 2 of our rebound-based collision attack framework, and we defer the more accurate model for the probability evaluation provided in [Don+21b] for related-key rebound attacks in the following outbound phase.

The Outbound Phase. We now introduce the Dong *et al.*'s model [Don+21b] of the outbound phase, a variable $\Pr_r^j$ is introduced in [Don+21b] to record the real differential probability for the outbound phase. To be more specific, $\Pr_r^j$ is used to count the number

Algorithm 8: Jean *et al.*'s algorithm for solving 3-round inbound phase

```
Input: $\Delta Z_0$ and $\Delta W_3$
Output: $(Z_0, Z'_0)$ and $(W_3, W'_3)$, with $\Delta Z_0 = Z_0 \oplus Z'_0$ and $\Delta W_3 = W_3 \oplus W'_3$
Compute $\Delta X_1$ from $\Delta Z_0$ and $\Delta Y_3$ from $\Delta W_3$;
for $j = 0, 1, ..., d-1$ do
    // Pre-compute $\texttt{SSB}^f_j$ and $\texttt{SSB}^b_j$
    $\mathbb{L}^f_j \leftarrow [], \mathbb{L}^b_j \leftarrow []$;
    for $j$-th $\texttt{SSB}^f_j$: $x \in \mathbb{F}_2^{dc}$ values of $X_{1,j}$ do
        Compute to $j$-th row $\Delta Y^f_{2,j} = \texttt{SSB}^f_j(x) \oplus \texttt{SSB}^f_j(x \oplus \Delta X_{1,j})$;
        $\mathbb{L}^f_j[\Delta Y^f_{2,j}] \leftarrow (x, x \oplus \Delta X_{1,j})$;
    for $j$-th $\texttt{SSB}^b_j$: $x \in \mathbb{F}_2^{dc}$ values of $Y_{3,j}$ do
        Compute to $j$-th antidiagonal $\Delta Y^b_{2,j} = \texttt{SSB}^b_j(x) \oplus \texttt{SSB}^b_j(x \oplus \Delta Y_{3,j})$;
        $\mathbb{L}^b_j[\Delta Y^b_{2,j}] \leftarrow (x, x \oplus \Delta Y_{3,j})$;
for difference of red cells $\Delta x \in \mathbb{F}_2^{d^2c/2}$ in $Y_2$ do
    Lookup $\Delta x$ in $\mathbb{L}^f_{0,...,d/2-1}$ to get a pair $(x, x')$ for first $d/2$ rows of $Y_2$;
    // Each $\mathbb{L}^b_j$ with DoF $2^{dc}$ and filtered with probability $2^{-dc}$
    Check corresponding cells of $\Delta x$ and $(x, x')$ in each $\mathbb{L}^b_j$;
    // One candidate of $Y_2$ is expected here
    Obtain full states $Y_2, Y'_2$ and compute backward to $X_1, X'_1$;
    if conforming with the fixed difference in $\Delta X_1[■]$ then
        Compute to recover all states of inbound phase;
        return $(Z_0, Z'_0)$ and $(W_3, W'_3)$;
// Time complexity: $2^{d^2c/2}$ computations of the inbound
// Memory complexity: $2d \cdot 2^{dc}$ for SSB
```

of consumed cells of the j-th column in the r-th round in the outbound part (encryption states only).

The set of forward rounds is denoted as `fwd` and backward as `bwd`. In forward rounds, the following variables are adopted to record the DoF cost of `MC` and `AK` operations

- c^j_r: the number of cells cancelled after the r-th round `MC` in column j;
- $\tilde{c}^j_r$: the number of cells cancelled after the r-th round next `AK` in column j.

Then, the probability of the j-th column $\Pr^j_r$ for forward rounds has two cases:

- $\sum_i x^{i,j}_r \geq c^j_r + \tilde{c}^j_r$ indicates no extra cost in key schedule, the probability can be estimated by $c^j_r + \tilde{c}^j_r$;
- $\sum_i x^{i,j}_r < c^j_r + \tilde{c}^j_r$ indicates extra costs exist in key schedule[5], the probability can be estimated by $\sum_i x^{i,j}_r$, with extra DoF consumption in key states as $c^j_r + \tilde{c}^j_r - \sum_i x^{i,j}_r$.

Thus for the forward outbound part, the real probability can be formulated by

$$\Pr^j_r = \min(c^j_r + \tilde{c}^j_r, \sum\nolimits_i x^{i,j}_r). \tag{5}$$

[5] For this case under the related-key setting (please refer to [Don+21b, Figure 5 and Figure 6]), one can consume the DoF in key states to enhance the probability of the outbound phase, and such possible DoF costs in key states will be recorded to evaluate the birthday phase later.

In backward rounds, similar to the forward part, the following variables are adopted to record the DoF cost of $\texttt{MC}^{-1}$ and $\texttt{AK}$ operations

- c_r^j: the number of cells cancelled after the r-th round $\texttt{MC}^{-1}$ in column j;
- $\tilde{c}_r^j$: the number of cells cancelled before the r-th round next $\texttt{AK}$ in column j.

Then, the probability of the j-th column Pr_r^j for backward rounds has two cases:

- $\sum_i x_r^{i,j} \geq c_r^j$ indicates no extra cost in key schedule, the probability of $\texttt{MC}^{-1}$ in this column can be estimated by $c_r^j + \tilde{c}_r^j$;
- $\sum_i x_r^{i,j} < c_r^j$ indicates leading extra cost in key schedule, the probability of $\texttt{MC}^{-1}$ in this column can be estimated by $\sum_i x_r^{i,j}$, with extra DoF consumption in key states as $c_r^j - \sum_i x_r^{i,j}$.

Thus for the backward outbound part, the real probability is as below

$$\mathrm{Pr}_r^j = \min(c_r^j, \sum\nolimits_i x_r^{i,j}). \tag{6}$$

Combined with the collided cells in the feed-forward cancellation of rebound attacks, denoted by $\sum x_0^{i,j}$, the number of starting points needed for a collision in the outbound phase is denoted by $\tau_{\texttt{OB}}$ with expression:

$$\log_2(\tau_{\texttt{OB}}) = c \cdot \sum\nolimits_{r,j} \mathrm{Pr}_r^j + c \cdot \sum\nolimits_{i,j} x_0^{i,j}, \tag{7}$$

which corresponds to the probability of the outbound phase being $p = 1/\tau_{\texttt{OB}}$. And the DoF consumption in key schedule by the states is computed by:

$$cost_{\texttt{ST}} = c \cdot \sum\nolimits_{r \in \texttt{fwd},j} (c_r^j + \tilde{c}_r^j - \mathrm{Pr}_r^j) + c \cdot \sum\nolimits_{r \in \texttt{bwd},j} (c_r^j - \mathrm{Pr}_r^j), \tag{8}$$

which will be considered later to evaluate the complexity of the birthday phase.

The Inbound Phase. Let $r_{\texttt{IB}}$ be the number of rounds for the inbound phase, and it starts from round l with the input difference set $\Delta_{in} \subseteq \mathbb{F}_2^{d^2c}$ and ends at round $l' = l + r_{\texttt{IB}}$ with the output difference set $\Delta_{out} \subseteq \mathbb{F}_2^{d^2c}$. It usually has two choices for $r_{\texttt{IB}}$, i.e., $r_{\texttt{IB}} = 2$ or 3, which have different evaluations of the time complexity $\mathcal{T}_{\texttt{IB}}$ to find a conforming pair of the inbound phase. In the following, we will introduce how to model the inbound phase for two cases of $r_{\texttt{IB}}$ under both classical and quantum settings.

For $r_{\texttt{IB}} = 2$, as introduced in Section 2.4, the super S-box technique is used to solve a 2-round inbound. In the classical setting, solving the inbound phase can be done in constant time with additional memory to store the super S-boxes. Thus, it has $\mathcal{T}_{\texttt{IB}}^c = 1$ under the classical setting; However, for non-full-active super S-boxes, Sasaki *et al.* [SLWSO10] found a method to reduce the memory complexity further, but the time complexity is $\mathcal{T}_{\texttt{IB}}^c = 2^{c(d-\min_j s_j)}$ to find a conforming pair, where s_j means the number of inactive or fixed difference cells in the j-th super S-box; In the quantum setting without qRAM, for 2-round full-active super S-box inbound, it takes the time complexity $\mathcal{T}_{\texttt{IB}}^q = 2^{dc/2}$ to find a right pair. While for 2-round non-full-active super S-box inbound, as observed by Dong *et al.* [Don+20] at Asiacrypt 2020, the time complexity could be reduced further to $\mathcal{T}_{\texttt{IB}}^q = 2^{c(d-\min_j s_j)/2}$ by applying Grover's algorithm to gain a quadratic speedup.

For the case of $r_{\texttt{IB}} = 3$, we only consider the inbound part with three full-active rounds under the quantum setting [HS20], and we present an improved algorithm to solve 3-round full-active inbound phase in Section 4.2. According to our improved algorithm, by applying Grover's algorithm, the time complexity is $\mathcal{T}_{\texttt{IB}}^q = 2^{d^2c/4+dc/4}$ without memory usage.

Besides the time complexity to solve the inbound phase, the DoF of the inbound part, denoted by $\tau_{\mathtt{IB}}$, should be more accurately evaluated to guarantee that one can generate enough conforming pairs for the outbound phase with $\tau_{\mathtt{OB}}$ complexity given in Equation 7. For both 2-/3-round full-active inbound part, considering that there is no inside DoF cost, with the size of input difference $|\Delta_{in}| = c \cdot \sum z_l^{i,j}$ and the size of output difference $|\Delta_{out}| = c \cdot \sum w_l^{i,j}$, it has

$$\log_2(\tau_{\mathtt{IB}}) = |\Delta_{in}| + |\Delta_{out}|. \tag{9}$$

While, for 2-round non-full-active inbound part, it may have some DoF costs inside the inbound phase, *i.e.*, $\mathtt{MC}$ operation at the l-th round, the next $\mathtt{AK}$ operation at the $(l+1)$-th round, and $\mathtt{MC}^{-1}$ operation at the $(l+2)$-th round, then it has to consider such DoF costs

$$\log_2(\tau_{\mathtt{IB}}) = |\Delta_{in}| + |\Delta_{out}| - c \cdot \sum\nolimits_j c_l^j - c \cdot \sum\nolimits_j \tilde{c}_l^j - c \cdot \sum\nolimits_j c_{l+2}^j. \tag{10}$$

Thus, to ensure the inbound phase with enough DoF, it has

$$\tau_{\mathtt{IB}} \geq \tau_{\mathtt{OB}}. \tag{11}$$

The Birthday Phase. With the configured inbound phase and outbound phase for the rebound attack in Step 3 of our attack framework, the predetermined difference set $\mathcal{S}$ is known, and the space of $\mathcal{S}$ is directly related to the DoF cost in key states, which is provided in Equation 8. Thus, the size of $\mathcal{S}$ can be evaluated from the DoF of the initial key state $k_{\mathtt{init}}$ targeted in the birthday phase by

$$\log_2(|\mathcal{S}|) = c \cdot \sum\nolimits_{i,j} k_{\mathtt{init}}^{i,j} - cost_{\mathtt{ST}}. \tag{12}$$

One then applies the birthday search to obtain a satisfied injecting difference $\delta_i \in \mathcal{S}$ with the time complexity

$$\tau_{\mathtt{BB}} = \sqrt{2^{d^2 c}/\mathcal{S}}. \tag{13}$$

Overall Time Complexity and Objective. Finally, we unify the time complexity of Step 2 (birthday phase) and Step 3 (rebound attack phase) together into an overall attack time complexity as the objective, denoted by $\tau_{\mathtt{OBJ}}$.

In the classical setting, since we only consider $r_{\mathtt{IB}} = 2$ with full-active super S-boxes due to the efficient generation of conforming pairs, the time complexity of rebound attack is $\tau_{\mathtt{OB}}$ as $\mathcal{T}_{\mathtt{IB}}^c = 1$, then the objective below should be *minimized*

$$\tau_{\mathtt{OBJ}}^c = \max(\tau_{\mathtt{OB}}, \tau_{\mathtt{BB}}). \tag{14}$$

In the quantum setting without qRAM, the time complexity[6] of rebound attack is $\mathcal{T}_{\mathtt{IB}}^q \cdot \sqrt{\tau_{\mathtt{OB}}}$, then the objective below should be *minimized*

$$\tau_{\mathtt{OBJ}}^q = \max(\mathcal{T}_{\mathtt{IB}}^q \cdot \sqrt{\tau_{\mathtt{OB}}}, \tau_{\mathtt{BB}}). \tag{15}$$

In both classical and quantum settings, the time complexity $\tau_{\mathtt{BB}}$ of birthday phase is the same, as given in Equation 13. And it should be noted that, the time complexity of the outbound phase $\tau_{\mathtt{OB}}$ in Equation 7 can gain a quadratic speedup under the quantum setting.

D Classical CPC Attack on 5-round AES-MMO

In this section, we provide the CPC attack on 5-round AES-MMO under the classical setting, as shown in Figure 11, any key pair with active cells in $K_4[0,4]$ can be applied to this

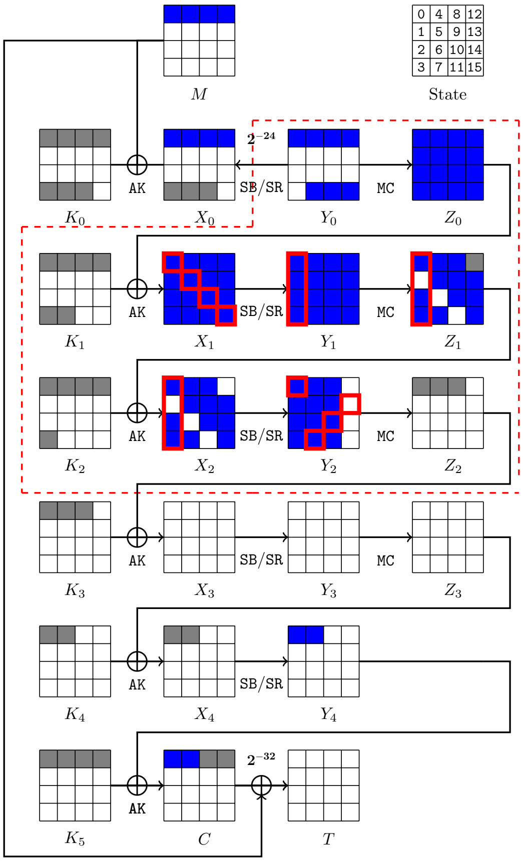


Figure 11: Classical rebound attack on 5-round `AES-MMO`

5-round attack. Therefore, the time complexity to find such a key pair is $2^{(128-16)/2} = 2^{56}$ by the birthday search from the first block.

When a proper key pair (K, K') is found, the rebound attack is performed in the second block. In Figure 11, the inbound phase covers from state Y_0 to Z_2. The number of input and output differences $(\Delta Y_0, \Delta Z_2)$ of the inbound phase is $2^{8 \cdot 7} = 2^{56}$. In the outbound phase, the probability is 2^{-56}. Hence, one collision is expected from this rebound attack. Details of this classical rebound attack on 5-round `AES-MMO` is given in Algorithm 9 with the time complexity of around 2^{56} and memory complexity of 2^{32}. Thus, the overall time complexity of our 5-round CPC attack on `AES-MMO` is $2^{56} + 2^{56} = 2^{57}$ with 2^{32} memory complexity.

E Quantum CPC Attack on 6-round `AES-MMO`

Probability and degree of freedom. As shown in Figure 12, the inbound phase covers 2 rounds from Y_0 to Z_2. The DoF of the inbound phase is provided by the bytes marked blue in Y_0 to Z_2, that is $2^{(7+4)\cdot 8} = 2^{88}$. The probability of the outbound phase is $2^{-(2+1+1+7)\cdot 8} = 2^{-88}$. Therefore, one collision is expected.

Quantum implementation of the rebound phase. We provide a detailed implementation of $\mathcal{U}_f$ to solve the rebound phase in quantum as Algorithm 10. Hereinafter, we

[6] Without memory consumption, it takes more time to obtain conforming pairs from the inbound phase.

Algorithm 9: Classical Rebound Attack on 5-round `AES-MMO`

```
Input: A key pair $(K, K')$
Output: A collision pair $(M, M')$
/* Note that $\Delta Z_2$ is fixed to ■ cells, we start to build the
   backward super S-box                                              */
Compute $\Delta Y_2 = \mathtt{MC}^{-1}(\Delta Z_2)$;
for $i = 0, 1, 2, 3$ do
    $\mathbb{L}^{(i)} \leftarrow [\,]$;
    for $x \in \mathbb{F}_2^{32}$ do
        Compute $\Delta X_1^{(i)} = \mathtt{SSB}_i^{-1}(x \oplus \Delta Y_2^{(i)}) \oplus \mathtt{SSB}_i^{-1}(x)$;
        $\mathbb{L}^{(i)}[\Delta X_1^{(i)}] \leftarrow (x, x \oplus \Delta Y_2^{(i)})$;
for $2^{56}$ differences of $\Delta Y_0$ do
    Obtain the full states of $Y_2$ and $Y_2'$ by accessing tables $\mathbb{L}^{(i)}$;
    if $(Y_2, Y_2')$ leads to a collision then
        Return $(M, M')$;
```

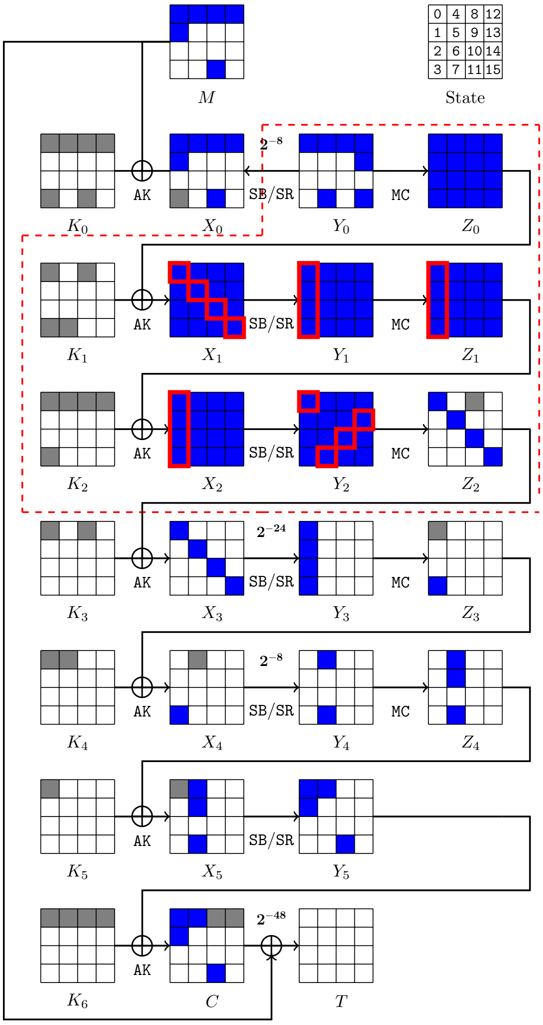


Figure 12: Quantum rebound attack on 6-round `AES-MMO`

use K and K' to denote the lists of round keys generated by the key pairs found in the birthday phase, which are fixed constants in the rebound phase. The super S-box operation with constant round keys is defined as $\mathtt{SSB}_j(K;\cdot) : \mathbb{F}_2^{32} \mapsto \mathbb{F}_2^{32}$ for $j \in [0, 1, 2, 3]$ (*e.g.*, $\mathtt{SSB}_0$ computes the bytes marked by the red rectangle in Figure 12). The input and output to $\mathtt{SSB}_j$ are expressed as $\delta x_j = x_{0,j}||x_{1,j}||x_{2,j}||x_{3,j}$ and $\delta y_j = y_{0,j}||y_{1,j}||y_{2,j}||y_{3,j}$, where $x_{i,j} = \Delta X_1[(i + j) \bmod 4 + 4j]$ and $y_{i,j} = \Delta Y_2[(i - j) \bmod 4 + 4j]$, for $i, j \in [0, 1, 2, 3]$. We further assume there are exactly 8 starting points (*i.e.*, Y_0, Y_0') for given $\Delta Y_0, \Delta Z_2$ and use $\alpha = \alpha_1||\alpha_2||\alpha_3 \in \mathbb{F}_2^3$ as index.

We then define $g_j : \mathbb{F}_2^{32} \mapsto \mathbb{F}_2$ for $j \in [0, 1, 2, 3]$ and apply Grover search on g_j to generate starting points for the 2-round inbound. When $j = 0$, $g_j(K, K', \delta x_j, \delta y_j; x_j) = 1$ if and only if $\mathtt{SSB}_j(K; x_j) \oplus \mathtt{SSB}_j(K'; x_j \oplus \delta x_j) = \delta y_j$. When $j > 0$, $g_j(K, K', \delta x_j, \delta y_j, \alpha_j; x_j) = 1$ if and only if $\mathtt{SSB}_j(K; x_j) \oplus \mathtt{SSB}_j(K'; x_j \oplus \delta x_j) = \delta y_j$ and $\alpha_j = 0, x_j < x_j \oplus \delta x_j$ or $\alpha_j = 1, x_j > x_j \oplus \delta x_j$.

Algorithm 10: Implementation of $\mathcal{U}_f$ without using qRAMs for 6-round `AES-MMO`

```
   Input: $|K, K'; \Delta Y_0, \Delta Z_2, \alpha\rangle\, |y\rangle$
   Output: $|K, K'; \Delta Y_0, \Delta Z_2, \alpha\rangle\, |y \oplus f(K, K'; \Delta Y_0, \Delta Z_2, \alpha)\rangle$
 1 Get $\delta x_i, \delta y_i$ for $i \in [0, 1, 2, 3]$ from the corresponding values of $\Delta X_1$ and $\Delta Y_2$;
 2 Run Grover search on $g_0(K, K', \delta x_0, \delta y_0; \cdot)$: $\mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$, get output $x_0$;
 3 Run Grover search on $g_1(K, K', \delta x_1, \delta y_1, \alpha_1; \cdot)$: $\mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$, get output $x_1$;
 4 Run Grover search on $g_2(K, K', \delta x_2, \delta y_2, \alpha_2; \cdot)$: $\mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$, get output $x_2$;
 5 Run Grover search on $g_3(K, K', \delta x_3, \delta y_3, \alpha_3; \cdot)$: $\mathbb{F}_2^{4\times 8} \mapsto \mathbb{F}_2$, get output $x_3$;
 6 /* Now we have the whole $Y_0$ */
 7 Compute $x_j' = x_j \oplus \delta x_j$ for $j \in [0, 1, 2, 3]$, get $X_1, X_1'$, then $Y_0, Y_0'$;
 8 Compute to $T$ and $T'$ from $Y_0$, $Y_0'$;
 9 if $(Y_0, Y_0')$ is a starting point for $(\Delta Y_0, \Delta Z_2, \alpha)$ and $T \oplus T' = 0$ then
10     return $|K, K'; \Delta Y_0, \Delta Z_2, \alpha\rangle\, |y \oplus 1\rangle$
11 else
12     return $|K, K'; \Delta Y_0, \Delta Z_2, \alpha\rangle\, |y\rangle$
13 Uncompute from Line 1 to 7;
```

Complexity analysis. We detail the complexity analysis of Algorithm 10. As the domain of each g_j is 2^{32}, the Grover search on each g_j (Line 2-5) takes about $\frac{\pi}{4} \cdot 2^{16}$ iterations. Each iteration of g_j needs $2 \cdot 2 \cdot 4 = 16$ `AES` S-box applications (a pair of 4-byte values are computed with 2 `AES` rounds). Hence, taking uncomputation into account, Line 2-5 runs in time approximately as $T_g \approx 4 \cdot \frac{\pi}{4} \cdot 2^{16} \cdot (2 \cdot 16)/120 \approx 2^{15.75}$ encryptions with 6-round `AES` (one computation of `AES` round function requires 20 S-boxes).

As the domain of f is $2^{88+3=91}$, about $\frac{\pi}{4} \cdot 2^{45.5}$ iterations of $\mathcal{U}_f$ find a collision. Checking if a starting point is valid and if a collision is found (Line 2-3) each takes about 2 encryptions with 6-round `AES`. Thus, taking uncomputation into account, the total time complxity of the rebound phase is about $\frac{\pi}{4} \cdot 2^{45.5} \cdot (T_g + 2 \cdot (2 + 2)) \approx 2^{60.9}$ (note that uncomputation for g_j is included in T_g). The time complexity of the birthday phase (*i.e.*, to find a desired pair of K, K') is $2^{(128-8)/2} = 2^{60}$. Therefore, the overall time complexity of this quantum CPC attack on 6-round `AES-MMO` is about $2^{60} + 2^{60.9} \approx 2^{61.5}$.

F Quantum CPC Attack on 7-round `Saturnin-hash`

Probability and degree of freedom. As shown in Figure 9 (b), the inbound phase covers 2 rounds from Y_1 to Z_3. The DoF of the inbound phase is provided by the bytes marked blue in Y_1 to Z_3, that is $2^{(8+3)\cdot 16} = 2^{176}$. The probability of the outbound phase is $2^{-(1+3+7)\cdot 16} = 2^{-176}$.

Quantum implementation of the rebound phase. Similarly, we use K and K' to denote the lists of round keys generated by the key pairs found in the birthday phase, which are fixed constants in the rebound phase. Here, the j-th super S-box is denoted by $\texttt{SSB}_j$ and its input difference is $\Delta X_r^{(j)}$, *e.g.*, $\texttt{SSB}_3$ marked by red circle in Figure 9 (b) from the third row of state X_2 with input difference $\Delta X_2^{(3)}$, and we use $X_2^{(3)}[0,1,2,3]$ to denote its four cells $X_2[3,7,11,15]$ in state X_2.

We assume there are exactly 8 starting points for the inbound part, and use $\alpha = (\alpha_0, \alpha_1, \alpha_2)$ as index. We then define $G^{(j)} : \mathbb{F}_2^{48} \mapsto \mathbb{F}_2$ for $j \in [0,1,2,3]$ and apply Grover search on $G^{(j)}$ to generate starting points for the 2-round inbound. When $j \in [0,1,2]$, $G^{(j)}(K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2]) = 1$ if and only if $\texttt{SSB}_j(K; X_2^{(j)}) \oplus \texttt{SSB}_j(K'; X_2^{(j)} \oplus \Delta X_2^{(j)}) = \Delta Y_3^{(j)}$ and $\alpha_j = 0, X_2^{(j)} < X_2^{(j)} \oplus \Delta X_2^{(j)}$ or $\alpha_j = 1, X_2^{(j)} > X_2^{(j)} \oplus \Delta X_2^{(j)}$. When $j = 3$, $G^{(j)}(K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}; X_2^{(j)}[0,1,2]) = 1$ if and only if $\texttt{SSB}_j(K; X_2^{(j)}) \oplus \texttt{SSB}_j(K'; X_2^{(j)} \oplus \Delta X_2^{(j)}) = \Delta Y_3^{(j)}$. Then implementation of $G^{(j)}$ when $j \in [0,1,2]$ is given in Algorithm 11. $G^{(3)}$ can be implemented similarly without indicators α_j and `flag` (and related constraints).

Based on $\mathcal{U}_{G^{(j)}}$, we now provide a detailed implementation of $\mathcal{U}_f$ to solve the quantum rebound phase for 7-round `Saturnin-hash` in Algorithm 12.

Algorithm 11: Implementation of $\mathcal{U}_{G^{(j)}}$ without using qRAMs

```
Input: $|K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2]\rangle\,|y\rangle$
Output: $|K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2]\rangle$
$|y \oplus G^{(j)}(K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2])\rangle$
1  // For $j$-th SSB in 2-round inbound phase in Figure 9 (b)
2  $Y_2^{(j)}[i] \leftarrow S(X_2^{(j)}[i])$ for $i \in [0,1,2]$;
3  $\Delta Y_2^{(j)}[i] \leftarrow Y_2^{(j)}[i] \oplus S(X_2^{(j)}[i] \oplus \Delta X_2^{(j)}[i])$ for $i \in [0,1,2]$;
4  Solving the system of equations $\texttt{MR}(\Delta Y_2^{(j)}) = \Delta Z_2^{(j)}$ with the knowledge of deduced
     $\Delta Y_2^{(j)}[0,1,2]$ and $\Delta Z_2^{(j)}[0] = 0$ to obtain $\Delta Y_2^{(j)}[3]$ and $\Delta Z_2^{(j)}[1,2,3]$;
5  // Check input and output differences of the S-box in $g^{(j)}$
6  // Define $g^{(j)} : \mathbb{F}_2^{16} \mapsto \mathbb{F}_2$ be a Boolean function such that
       $g^{(j)}(K, K', \delta_{in}, \delta_{out}; x) = 1$ if and only if $S(x) \oplus S(x \oplus \delta_{in}) = \delta_{out}$
7  Run Grover search on $g^{(j)}(K, K', \Delta X_2^{(j)}[3], \Delta Y_2^{(j)}[3]; \cdot)$: $\mathbb{F}_2^{16} \mapsto \mathbb{F}_2$, and let the
     output be $X_2^{(j)}[3]$;
8  Compute $Y_2^{(j)}[3] = S(X_2^{(j)}[3])$ and $Z_2^{(j)} = \texttt{MR}(Y_2^{(j)})$ to derive $X_3^{(j)}$;
9  Set flag $= 1$ if and only if $X_2^{(j)} < X_2^{(j)} \oplus \Delta X_2^{(j)}$ and $\alpha_j = 0$ or
     $X_2^{(j)} > X_2^{(j)} \oplus \Delta X_2^{(j)}$ and $\alpha_j = 1$;
10 // Check if a starting point of $(\Delta X_2^{(j)}, \Delta Y_3^{(j)})$
11 if $S(X_3^{(j)}[i]) \oplus S(X_3^{(j)}[i] \oplus \Delta Z_2^{(j)}[i]) = \Delta Y_3^{(j)}[i]$ for $i \in [1,2,3]$ and flag $= 1$ then
12     return $|K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2]\rangle\,|y \oplus 1\rangle$
13 else
14     return $|K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; X_2^{(j)}[0,1,2]\rangle\,|y\rangle$
15 Uncompute from Line 2 to 9;
```

Complexity analysis. The complexity of Algorithm 11 (one iteration of $G^{(j)}$) is dominated by the Grover search on $g^{(j)}$ in Line 7, which takes about $T_g \approx \frac{\pi}{4} \cdot 2^{16/2} \cdot 2 \cdot 2 \approx 2^{9.65}$ S-box operations (uncomputation included).

In Algorithm 12, the Grover search on $G^{(j)}$ (Line 4 and Line 8) takes about $\frac{\pi}{4} \cdot 2^{(48)/2} \approx 2^{23.65}$ iterations. Hence, the Grover search on $G^{(j)}$ runs in time approximately as $2^{23.65} \cdot (T_g + 2 \cdot (3 \cdot 4 + 1)) \approx 2^{33.35}$ S-box operations, which takes about $T_G \approx 2^{33.35}/(7 \cdot 16) \approx 2^{26.54}$ encryptions with 7-round `Saturnin-hash` (uncomputation included).

In Algorithm 12, Line 5 just replays Algorithm 11 from Line 2 to Line 9 to recover the input state of $X_2^{(j)}$ for non-full-active $\texttt{SSB}_j$, the time complexity of which is about $2^{9.65} + 2 \cdot (3 \cdot 2 + 1) \approx 2^{9.67}$ S-box operations, thus can be reduced to $T_R \approx 2^{9.67}/(7 \cdot 16) \approx 2^{2.87}$ 7-round `Saturnin-hash` encryptions (uncomputation included).

In Algorithm 12, as the domain of f is $2^{11 \cdot 16 + 3} = 2^{179}$, about $\frac{\pi}{4} \cdot 2^{179/2}$ iterations of $\mathcal{U}_f$ finds a collision. The inbound phase (Line 2-9) takes about $2 \cdot 4 \cdot (T_G + T_R) \approx 2^{29.54}$ 7-round `Saturnin-hash` encryptions (uncomputation included). The checking of if a collision is found takes several encryptions of 7-round `Saturnin-hash`, which is negligible compared to the inbound phase. Thus, the time complexity of the rebound phase is about $\frac{\pi}{4} \cdot 2^{179/2} \cdot 2^{29.54} \approx 2^{118.69}$ 7-round `Saturnin-hash` encryptions. The time complexity of the birthday phase, *i.e.*, using quantum birthday attack to find a desired pair of (K, K'), is $2^{(256-32)/2} = 2^{112}$. Therefore, the overall time complexity of this quantum CPC attack on 7-round `Saturnin-hash` is about $2^{112} + 2^{118.69} \approx 2^{118.70}$.

Algorithm 12: Implementation of $\mathcal{U}_f$ without using qRAMs for 7-round `Saturnin-hash`

```
    Input: $|K, K'; \Delta Y_1, \Delta Z_3, \alpha\rangle\, |y\rangle$ with $\alpha = (\alpha_0, \alpha_1, \alpha_2) \in \mathbb{F}_2^3$
    Output: $|K, K'; \Delta Y_1, \Delta Z_3, \alpha\rangle\, |y \oplus f(K, K'; \Delta Y_1, \Delta Z_3, \alpha)\rangle$
 1  // Solve the inbound phase
 2  for $j \in [0, 1, 2]$ do
 3      Compute the corresponding differences $(\Delta X_2^{(j)}, \Delta Y_3^{(j)})$ for non-full-active SSB
          from $(\Delta Y_1, \Delta Z_3)$;
 4      Run Grover search on the function $G^{(j)}(K, K', \Delta X_2^{(j)}, \Delta Y_3^{(j)}, \alpha_j; \cdot) : \mathbb{F}_2^{48} \mapsto \mathbb{F}_2$.
          Let $X_2^{(j)}[0, 1, 2] \in \mathbb{F}_2^{48}$ be the output;
 5      Run Line 1-8 in Algorithm 11 to compute $X_2^{(j)}[3]$;
 6      // Now we have the whole $X_2^{(j)}$ according to four cells $X_2^{(j)}[0, 1, 2, 3]$
 7  Compute the corresponding differences $(\Delta X_2^{(3)}, \Delta Y_3^{(3)})$ for non-full-active SSB from
      $(\Delta Y_1, \Delta Z_3)$;
 8  Run Grover search on the function $G^{(3)}(K, K', \Delta X_2^{(3)}, \Delta Y_3^{(3)}; \cdot) : \mathbb{F}_2^{48} \mapsto \mathbb{F}_2$. Let
      $X_2^{(3)}[0, 1, 2] \in \mathbb{F}_2^{48}$ be the output;
 9  Run Line 1-8 in Algorithm 11 to compute $X_2^{(3)}[3]$;
10  // Derive the starting point
11  $X \to (X_2^{(0)}, X_2^{(1)}, X_2^{(2)}, X_2^{(3)})$;
12  $X' \to (X_2^{(0)} \oplus \Delta X_2^{(0)}, X_2^{(1)} \oplus \Delta X_2^{(1)}, X_2^{(2)} \oplus \Delta X_2^{(2)}, X_2^{(3)} \oplus \Delta X_2^{(3)})$;
13  // Check the outbound phase
14  if $(X, X')$ fulfills the outbound differential then
15      return $|K, K'; \Delta Y_1, \Delta Z_3, \alpha\rangle\, |y \oplus 1\rangle$
16  else
17      return $|K, K'; \Delta Y_1, \Delta Z_3, \alpha\rangle\, |y\rangle$
18  Uncompute from Line 2 to 12;
```