IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2021, No. 2, pp. 222–248. DOI:10.46586/tosc.v2021.i2.222-248

(Quantum) Collision Attacks on Reduced Simpira v2

Boyu Ni[1], Xiaoyang Dong[2] ✉, Keting Jia[3] ✉ and Qidi You[4]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, China,
niboyu@mail.sdu.edu.cn
[2] Institute for Advanced Study, BNRist, Tsinghua University, Beijing, China,
xiaoyangdong@tsinghua.edu.cn
[3] Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing, China,
ktjia@mail.tsinghua.edu.cn
[4] Department of Computer Science and Technology, Tsinghua University, Beijing, China

Abstract. Simpira v2 is an AES-based permutation proposed by Gueron and Mouha at ASIACRYPT 2016. In this paper, we build an improved MILP model to count the differential and linear active Sboxes for Simpira v2, which achieves tighter bounds of the minimum number of active Sboxes for a few versions of Simpira v2. Then, based on the new model, we find some new truncated differentials for Simpira v2 and give a series (quantum) collision attacks on two versions of reduced Simpira v2.

Keywords: Collision · Rebound Attack · Simpira v2 · Quantum Attack · MILP

1 Introduction

Simpira v2 is a family of cryptographic permutations designed by Gueron and Mouha in ASIACRYPT 2016 [GM16] that accepts arbitrarily large input sizes of $b \times 128$ bits, where b is an positive number. In order to take advantage of the Intel AES-NI instruction set for optimized software implementations, Simpira v2 uses two-round AES as its build blocks.

There have been several cryptanalysis papers on Simpira. Rønjom [Røn16] proposed an invariant subspace attack on Simpira v1, an informal version of Simpira. At SAC 2016, Dobraunig et al. [DEM16] provided the differential trails with only 40 (instead of 75) active Sboxes for Simpira v1 $[b = 4]$ for the recommended 15 rounds. Zong et al. [RZW18] presented the impossible differential cryptanalysis of Simpira v2 $[b = 4]$. At ACISP 2017, Tjuawinata et al. [THW17] proposed truncated differential analysis on Simpira v2 $[b = 3]$. The authors of Simpira v2 proposed SPHINCS-Simpira [GM17], which is a variant of the SPHINCS [BHH$^+$15] signature scheme with Simpira in DM hashing mode as a building block. At PQCrypto 2018, Stefan Kölbl [Köl18] compared the performance of SPHINCS [BHH$^+$15] instantiated with several cryptographic hash functions, where Simpira and Haraka [KLMR16] outperform other hash functions, including ChaCha, Keccak and SHA256. When building hash functions with Simpira v2, we have to understand its security against collision attacks, (second) preimage attacks, etc. In this paper, we focus on the security of Simpira v2 against (quantum) collision attacks.

Post-quantum cryptography has received much attention due to the Shor's seminal work [Sho94]. For the popular public-key cryptosystems, like RSA and ECC, their security are often reduced to some mathematical problems, i.e., factor numbers and compute discrete logarithms, which are directly affected by Shor's quantum algorithm. The study on symmetric cryptography aganist quantum computers has been not as active as public-key cryptography, until recently a series of quantum polynomial time attacks on symmetric

Received: 2021-03-01 Accepted: 2021-05-01 Published: 2021-06-11

schemes, such as 3-round Feistel [KM10], EM construction [KM12] and a lot of MACs and authenticated encryption schemes [KLLN16a]. Since then, a lot of quantum cryptanalysis results [KLLN16b, LM17, BHN+19, CNS17, HS18a, HS18b, GNS18, BNS19a, BNS19b, NS20] on symmetric ciphers have appeared.

For generic (quantum) collision attacks, the parallel rho method [vOW94] gave the tradeoff time complexity $T = 2^{n/2}/S$ in both classical setting and quantum setting [Ber]. In the quantum setting, BHT algorithm [BHT98] found collisions with a query complexity of $O(2^{n/3})$ and the quantum random access memory (qRAM) is $O(2^{n/3})$-qubit. But researchers generally agree that large amounts of memory are not easy to make. Therefore, it also makes sense when there are algorithms that require only a small amount of memory but have a time complexity greater than $O(2^{n/3})$. Chailloux et al. [CNS17] presented a collision-finding algorithm that runs in time $O(2^{2n/5})$ with a classical memory of size $O(2^{n/5})$. None of these quantum attacks is directed at a specific structure.

When delving into dedicated collision attacks on specific hash functions, such as AES-like hashing, we have rebound attacks introduced by Mendel et al. at FSE 2009 [MRST09]. At ASIACRYPT 2009, Lamberger et al. improved the rebound attacks by proposing multiple inbound phases [LMR+09]. At FSE 2010, Gilbert et al. introduced the Super-Sbox technique to further extend the inbound phase [GP10]. At CRYPTO 2011, Naya-Plasencia [Nay11] improved the rebound attack by introducing clever ways to merging large lists. The rebound attacks have been successfully applied to many hash functions, such as WHIRLPOOL [MRST09, LMR+09], Grøstl [MRST09, JNP12, MRS14, JNP13, SLW+10], ECHO [JNS11, MPRS09], Keccak [DGPW12], Lane [MNN+09], etc.

At EUROCRYPT 2020, for the first time, Hosoyamada and Sasaki [HS20] presented dedicated collision attacks on AES-MMO and WHIRLPOOL by applying a quantum version of the rebound attack. Later, Dong et al. [DSS+20] reduced the qRAM significantly by applying a quantum version of non-full Super-Sbox technique [SLW+10] into the quantum rebound attacks on reduced AES-MMO and Grøstl, which outperforms the generic collision attack given by Chailloux et al. [CNS17].

Our Contribution.

In this paper, we present collision attacks on reduced Simpira v2 with Davies-Meyer hashing mode in both classical and quantum setting. First, we build an improved MILP model to count the differential and linear active Sboxes for Simpira v2, which achieves tighter bounds of the minimum number of active Sboxes for a few versions of Simpira v2. Then, based on the new model, we find some new truncated differentials for Simpira v2 and give a series (quantum) collision attacks on two versions of reduced Simpira v2, including the (quantum) collision attacks on 9-round Simpira v2 $[b = 2]$ and 11-round Simpira v2 $[b = 4]$. A summary of the main attacks on Simpira v2 is given in Table 1. Note that for the 11-round quantum collision attacks on branches $[A, D]$ and $[B, D]$ of Simpira-4, the attacks are better than the quantum version of parallel rho algorithm [HS20]. However, the time is slightly larger than the generic attack by Chailloux, Naya-Plasencia, Schrottenloher [CNS17], which needs $2^{102.4}$ time and $2^{51.2}$ classical memory.

2 Preliminaries

2.1 Definition of Simpira v2

Simpira v2 [GM16] is a family of cryptographic permutations that supports inputs of $128 \times b$ bits, where b is number of branches. When $b = 1$, Simpira v2 consists of 12 rounds AES with different constants. When $b \geq 2$, Simpira v2 is a Generalized Feistel Structure (GFS) with the F-function that consists of two rounds of AES. We denote Simpira v2

Table 1: Classical and quantum collision attacks (no QRAM) on Simpira v1 and Simpira v2. The total number of rounds is 15 for $b = 2$, $b = 4$, 21 for $b = 3$. C-Mem: classical memory; QRAM: quantum random access memory. For Simpira-4, the four branches are denoted as (A, B, C, D). "Collision $[A, B]$" means the collision happens in branch A and B.

Simpira v1						
Branches	Setting	Attack	Rounds	Time	C-Mem	Source
$b = 4$	Classical	Collision	15	$2^{82.62}$	2^{33}	[DEM16]
			15	$2^{110.16}$	2^{33}	[DEM16]
Simpira v2						
$b = 2$	Classical	Collision	9	2^{96}	2^{32}	Sect. 4
	Quantum		9	$2^{66.1}$	0	Sect. 5
$b = 3$	Classical	Distinguisher	8	2	-	[THW17]
			9	2^{22}	-	[THW17]
			10	2^{23}	-	[THW17]
		Key-recovery	9	2^{50}	-	[THW17]
			10	2^{70}	-	[THW17]
$b = 4$	Classical	Key-recovery	9	2^{57}	2^{57}	[RZW18]
		Key-recovery	9	2^{170}	2^{170}	[RZW18]
		Collision $[A, B]$	11	2^{64}	2^{32}	Sect. 4
		Collision $[A, C]$	11	2^{96}	2^{32}	Sect. 7
		Collision $[A, D]$	11	✗	✗	✗
		Collision $[B, C]$	11	2^{96}	2^{32}	Sect. 7
		Collision $[B, D]$	11	✗	✗	✗
		Collision $[C, D]$	11	2^{96}	2^{32}	Sect. 7
	Quantum	Collision $[A, B]$	11	$2^{49.85}$	0	Sect. 6
		Collision $[A, C]$	11	$2^{65.85}$	0	Sect. 7
		Collision $[A, D]$	11	$2^{113.85}$	0	Sect. 7
		Collision $[B, C]$	11	$2^{65.85}$	0	Sect. 7
		Collision $[B, D]$	11	$2^{113.85}$	0	Sect. 7
		Collision $[C, D]$	11	$2^{65.85}$	0	Sect. 7
		Collision $[*,\ *]$	11	$2^{102.4}$	$2^{51.2}$	[CNS17]

family members with b branches as Simpira-b. The total number of rounds is 15 for $b = 2$, $b = 4$ and $b = 6$, 21 for $b = 3$, and 18 for $b = 8$. The round function F consists of an AddConstant (AC) operation and two rounds of AES while omitting the AddRoundKey (ARK) operation as illustratedin Figure 1. Otherwise, the operations such as SubBytes (SB), ShiftRows (SR), MixColumns (MC) are identical as AES. Every subblock can be divided into a 4×4 matrix of bytes as follows:

$$S = \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{bmatrix}.$$

Thus, the definition of the round function is:

$$F(S) = \mathtt{MC} \circ \mathtt{SR} \circ \mathtt{SB} \circ \mathtt{AC} \circ \mathtt{MC} \circ \mathtt{SR} \circ \mathtt{SB}(S).$$

To refer to intermediate states of F for an input S, we use the following notations:

$$S \xrightarrow{\mathtt{SB}} S^{\mathtt{SB1}} \xrightarrow{\mathtt{SR}} S^{\mathtt{SR1}} \xrightarrow{\mathtt{MC}} S^{\mathtt{MC1}} \xrightarrow{\mathtt{AC}} S^{\mathtt{AC1}} \xrightarrow{\mathtt{SB}} S^{\mathtt{SB2}} \xrightarrow{\mathtt{SR}} S^{\mathtt{SR2}} \xrightarrow{\mathtt{MC}} S^{\mathtt{MC2}} = F(S).$$

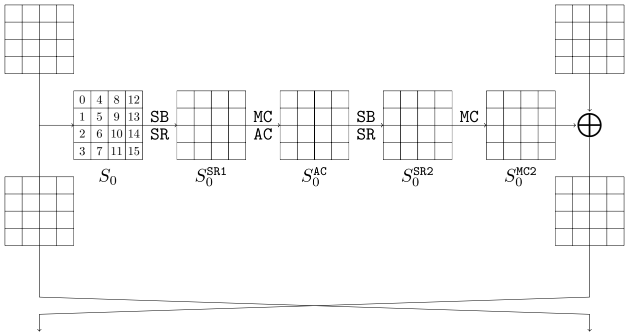

Figure 1: Round function of Simpira-2

2.2 The Rebound Attack

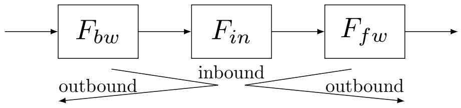

Figure 2: The Rebound-Attack Technique

The rebound attack was first introduced by Mendel et al. in [MRST09], it consists of an inbound phase and an outbound phase as shown in Figure 2, where F is an internal block cipher or permutation which is split into three subparts, then $F = F_{fw} \circ F_{in} \circ F_{bw}$.

- **Inbound phase**. In the inbound phase, the attackers efficiently fulfill the low probability part in the middle of the differential trail with a meet-in-the-middle technique. The degree of freedom is the number of matched pairs in the inbound phase, which will act as the starting points for the outbound phase.

- **Outbound phase**. In the outbound phase, the matched values of the inbound phase, i.e., starting points, are computed backward and forward through F_{bw} and F_{fw} to obtain a pair of values which satisfy the differential trail.

Suppose the probability of the inbound phase is p, then we have to prepare $1/p$ starting points in the inbound phase to expect one pair conforming to the differential trail of the outbound phase. Hence, the degree of freedom should be larger than $1/p$.

2.3 Super-Sbox Technique

The Super-Sbox technique was introduced by Gilbert et al. [GP10] and Lamberger et al. [LMR+09]. We use Figure 3 to briefly recall the details. We want to search for a pair of values whose difference satisfies the truncated differential $\Delta Z_0 \rightarrow \Delta W_2$.

For each of the 2^{32} differences at state Z_0, since the operation MC is linear, we compute the corresponding difference at state X_1. For each of the 2^{32} differences at state W_2, we compute the corresponding difference at state Y_2 and store them in a list L. Then, given ΔX_1, we search for a pair of values whose difference satisfies the differential trail $\Delta X_1 \rightarrow \Delta Y_2$, where $\Delta Y_2 \in L$. Here, we divide the search process into 4 parts that can all be computed independently. Each part contains 4 bytes, and we call the operation of changing the values of these 4 bytes from the state X_1 to state Y_2 as a Super-Sbox. We refer the reader to the example in Figure 3, one of the Super-Sboxes is highlighted. For each of the 2^{32} pairs of input values to a Super-Sbox at X_1, we compute the corresponding

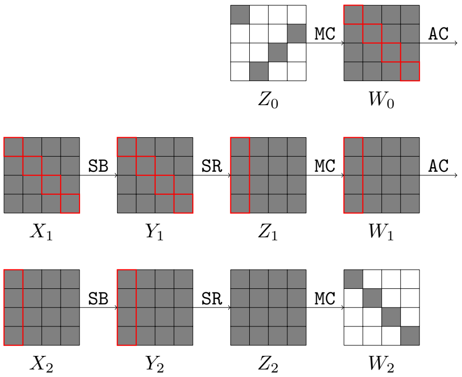

Figure 3: Super-Sbox technique

output difference ΔY_2. Since the list L contains 2^{32} values, the Super-Sbox position of each value has one collision on average with the output difference. Similarly, we perform the same analysis for the remaining 3 Super-Sboxes. After that, each difference in list L can be obtained once for the 4 Super-Sboxes. Then we get 2^{32} pairs $(X_1, X_1 \oplus \Delta X_1)$ satisfying the differential $\Delta X_1 \to \Delta Y_2$, $\Delta Y_2 \in L$, and the values and difference at Z_0 can be computed from X_1.

Complexity Analysis. In the above program, every Super-Sbox requires to compute 2^{32} input values, the length of which is 32 bits, and list L contains 2^{32} differences. Thus, both the time complexity and the memory complexity are 2^{32}, and we can obtain 2^{32} pairs of values that satisfy the requirement. So the amortized complexity to get one pair of values that satisfies the differential trail $\Delta Z_0 \to \Delta W_2$ is 1.

In a quantum setting, Hosoyamada and Sasaki [HS20] introduced two methods to apply the Super-Sbox technique. The first method is to use quantum random access memory (qRAM) to store the Super-Sbox. The second method is to apply the nested Grover's algorithm [Gro96] to compute the pair that conforms to $\Delta Z_0 \to \Delta W_2$ with an additional time complexity of about 2^{16} without qRAM.

2.4 Permutation-based Hashing with Davies-Meyer Construction

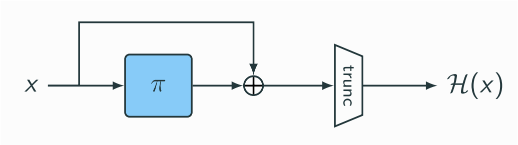

Figure 4: Davies-Meyer Construction

The designers of Simpira proposed several applications to the Simpira permutation, such as a block cipher using Even-Mansour construction, a wide-block encryption scheme and so on. One particular suggested application is a hash function using Davies-Meyer construction. The proposal is to use a single-block, keyless Davies-Meyer-like construction with a feed-forward, and compute the hash function $H(x)$ of x as shown in Figure 4. We can represent it as follows:

$$H(x) = \texttt{trunc}(\pi(x) \oplus x),$$

where the function π is the permutation Simpira-b. This approach provides an efficient construction for hashing inputs of limited length, which is required by many applications.

2.5 Quantum Computation and Quantum RAM

The states of an n-qubit quantum system can be described as unit vectors in C^{2^n} under the orthonormal base $\{|0\dots00\rangle, |0\dots01\rangle, \dots, |1\dots11\rangle\}$, alternatively written as $\{|i\rangle : 0 \leq i < 2^n\}$. The quantum algorithms can be described as series of unitary transformations and measurements acting on the state of an n-qubit system, where all unitary transformations can be implemented as a sequence of single-qubit and two-qubit quantum gates in the standard quantum circuit model. Then we introduce the standard quantum circuit model and adopt the basic gate set $\{H, CNOT, T\}$ (Clifford+T gates). Here, H is the single qubit Hadamard gate, $CNOT$ is the two-qubit $CNOT$ gate denote as $CNOT : |a\rangle|b\rangle \mapsto |a\rangle|b \oplus a\rangle$, and T is the gate defined as $T : |0\rangle \mapsto |0\rangle$ and $T : |1\rangle \mapsto e^{i\pi/4}|1\rangle$. The identity operator on n-qubit states is denoted by $I^{\otimes n}$.

Superposition Oracles for Classical Circuit. When f is a Boolean function, the quantum oracle of a function $f : \{0,1\}^n \mapsto \{0,1\}$ is modeled as the unitary operator $\mathcal{U}_f$ defined as:

$$\mathcal{U}_f : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle,$$

where $x \in \{0,1\}^n$ and $y \in \{0,1\}$. As a linear operator, $\mathcal{U}_f$ acts on superposition states as:

$$\mathcal{U}_f\left(\sum_{x\in\{0,1\}^n} a_i|x\rangle|0\rangle\right) = \sum_{x\in\{0,1\}^n} a_i|x\rangle|f(x)\rangle.$$

We notice that if there is an efficient classical circuit that computes f, the corresponding quantum circuit of $\mathcal{U}_f$ can be implemented efficiently in the quantum circuit model.

Grover's Algorithm. First, we describe the problem which can be solved by the Grover's algorithm.
Problem. Given a search space of 2^n elements, and a Boolean function or predicate $f : \{0,1\}^n \mapsto \{0,1\}$, where f is given as a black-box, find x such that $f(x) = 1$. (For the sake of simplicity, we assume that there is only one such x).

In classical setting, the best algorithm with a black-box access to f requires about 2^n evaluations of the black-box oracle to identify x such that $f(x) = 1$ with probability one. In the quantum setting, Grover search provides a quadratic speedup to solve the same problem, which needs about $O(\sqrt{2^n})$ calls to a quantum oracle $\mathcal{U}_f$ that outputs $\sum\limits_x a_i|x\rangle|y \oplus f(x)\rangle$ upon input of $\sum\limits_x a_i|x\rangle|y\rangle$. Firstly, we construct a uniform superposition of states

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle$$

by applying the Hadamard transformation $H^{\otimes n}$ to $|0\rangle^{\otimes n}$. Then Grover's algorithm iteratively applies the unitary transformation $(2|\phi\rangle\langle\phi| - I_n)\mathcal{U}_f$ to $|\phi\rangle$ such that the amplitudes of those values x with $f(x) = 1$ are amplified. And then we measure the final state x which has an overwhelming probability such that satisfies $f(x) = 1$.
Remark. It is important to understand what resources are required to implement the Oracle. Because there may be a lot of complexity when constructing Oracle circuits using Grover algorithm, Oracle circuits need to be implemented efficiently, or the acceleration of the search will be illusory.

Quantum Random Access Memories (qRAM). For a list of classical data $L = \{x_0, \cdots, x_{2^n-1}\}$ with $x \in \{0,1\}^m$, we introduce the definition of the qRAM for L. Quantum random access memory (QRAM) handle the quantum superposition of 2^n memory cells by using n-qubit, it is a quantum analogue of a classical random access memory, so the qRAM for L can be defined as an unitary transformation $\mathcal{U}^L_{QRAM}$:

$$\mathcal{U}^L_{qRAM} : |i\rangle_{Addr} \otimes |y\rangle_{Out} \mapsto |i\rangle_{Addr} \otimes |y \oplus x_i\rangle_{Out},$$

where $i \in \{0,1\}^n$ and $y \in \{0,1\}^m$, and $|\cdot\rangle_{Addr}$ and $|\cdot\rangle_{Out}$ may be regarded as the address and output registers respectively.

Therefore, we can access any quantum superposition of the data cells by using the corresponding superposition of addresses:

$$\mathcal{U}^L_{qRAM}\left(\sum_i a_i|i\rangle|y\rangle\right) = \sum_i a_i|i\rangle|y \oplus x_i\rangle.$$

3 Minimal Number of Active Sboxes for Simpira v2

3.1 Incompatibilities in the Truncated Trails of Simpira v2

When searching differential or linear trail on AES-like ciphers, it is usually to first determine a truncated differential trail with low number of active Sboxes. To build such searching algorithm, one has to deal with the incompatibilities, where the truncated trail within a small number of rounds holds but may bring in internal contradictions when considering more rounds. Similar incompatibilities [BN10, FJP13, GLMS18, ENP19] in the truncated differential or linear trail of AES-like ciphers have been discovered before. Here we find an incompatibility for a 3-round truncated linear trail for Simpira-2 as shown in Figure 5, where a one-round truncated linear trail is iterated 3 times. Now we prove that the 3-round trail is incompatible.

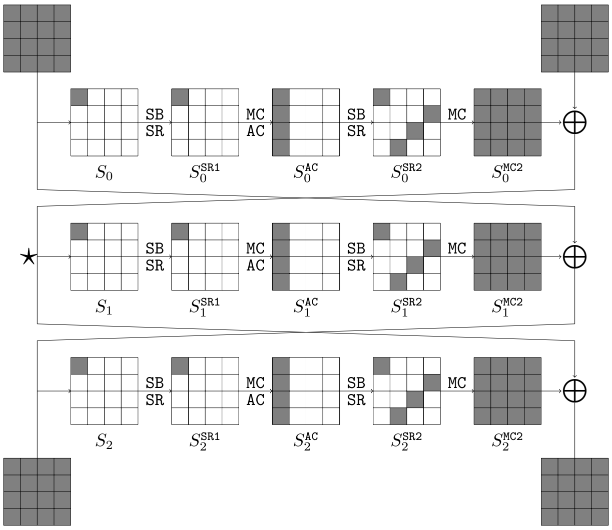

Figure 5: The 3-round linear trail on Simpira-2

Proof. Let us focus on the state S_1 marked with a star, we have the following equation:

$$S_1 = S_0^{\texttt{MC2}} \oplus S_2^{\texttt{MC2}} = MC(S_0^{\texttt{SR2}}) \oplus MC(S_2^{\texttt{SR2}}) = MC(S_0^{\texttt{SR2}} \oplus S_2^{\texttt{SR2}}).$$

As shown in Figure 5, follow the linear iterative trail of the least active Sboxes, each column of $S_0^{\mathtt{SR2}}$ and $S_2^{\mathtt{SR2}}$ has only one active Sbox, and there are at most two active Sboxes after XOR, namely each column of $S_0^{SR2} \oplus S_2^{SR2}$ with two active Sboxes at most.

According to the constraint of MC in the AES, that is, if the input to one column of MC has active Sboxes, then the total number of active Sboxes of input and output on this column is greater than or equal to 5. Thus, each column of S_1 should have at least 3 active Sboxes or no active Sboxes. In the given iterative trail, there is only one active Sbox in the first column of S_1. It is contradictory. Hence, there is no such linear trail. Based on this contradiction, we present an improved MILP model, which can get a tighter bound of the estimation of number of active Sboxes. □

3.2 Improved MILP Model for Counting Active Sboxes in the Differential Trails

For Simpira-2, we introduce a new MILP model and use it to search the linear and differential trail that has a minimum number of active Sboxes. The MILP models of the linear and differential trail are similar. Here we first focus on the MILP model of differential trails. The MILP model can also be simply translated into models for other versions of Simpira-b.

For the differential trail of r-round Simpira-2, we define some notations. First, for the convenience of expression, we directly denote the difference of the state S_i as S_i here, and the same for other states.

Then, we divide each difference of these states into 16 bytes, and use 1 and 0 to indicate its activity. For example, for the bytes in the row k $(k = 0, 1, 2, 3)$ and column l $(l = 0, 1, 2, 3)$ of the state ΔS_i, we call it the $(k + 4l)$-th byte of the state, marked as $S_i[k + 4l]$. When the Sbox is active, set $S_i[k + 4l] = 1$, otherwise, $S_i[k + 4l] = 0$. Next, we show the constraints of our MILP model for searching differential trails with the minimum number of active Sboxes.

Constraints of AES Round Function. The round of AES includes SubBytes (SB), ShiftRows (SR), MixColumns (MC). For these operations, SB does not change the activity of the state, and SR only changes the activity position in the state. The main constraint is in the MC operation, that the branch number is greater than or equal to 5. Therefore, each column and its corresponding output state meets the following conditions:

$$\begin{aligned} \sum_{k=0}^{3} (S_i^{\mathtt{SR1}}[k+4l] + S_i^{\mathtt{MC1}}[k+4l]) - 5d \geq 0, \\ d - S_i^{\mathtt{SR1}}[4l] \geq 0, \\ d - S_i^{\mathtt{SR1}}[1+4l] \geq 0, \\ d - S_i^{\mathtt{SR1}}[2+4l] \geq 0, \\ d - S_i^{\mathtt{SR1}}[3+4l] \geq 0, \\ d - S_i^{\mathtt{MC1}}[4l] \geq 0, \\ d - S_i^{\mathtt{MC1}}[1+4l] \geq 0, \\ d - S_i^{\mathtt{MC1}}[2+4l] \geq 0, \\ d - S_i^{\mathtt{MC1}}[3+4l] \geq 0, \end{aligned} \tag{1}$$

where auxiliary variable d is binary, and $0 \leq i \leq r$, $l = 0, 1, 2, 3$.

Constraints on the Feistel Structure. For the differential trail of the Feistel structure, the difference must be satisfied at the branch XOR conditions, that is, $S_i^{\mathtt{MC2}} = S_{i-1} \oplus S_{i+1}$, so we get the following XOR constraint model [ENP19]:

$$\begin{array}{c} S_i^{\texttt{MC2}}[t] + S_{i-1}[t] \geq S_{i+1}[t], \\ S_i^{\texttt{MC2}}[t] + S_{i+1}[t] \geq S_{i-1}[t], \\ S_{i-1}[t] + S_{i+1}[t] \geq S_i^{\texttt{MC2}}[t], \end{array}$$

where $1 \leq i \leq (r-1)$, $0 \leq t \leq 15$.

Due to the particularity of round function F of `Simpira-2` structure, we give a more accurate constraint, that is, the input of F_i function satisfies the following relationship:

$$S_i = S_{i-1}^{\texttt{MC2}} \oplus S_{i-2} = \texttt{MC}(S_{i-1}^{\texttt{SR2}}) \oplus S_{i-2},$$

$$S_i = S_{i+1}^{\texttt{MC2}} \oplus S_{i+2} = \texttt{MC}(S_{i+1}^{\texttt{SR2}}) \oplus S_{i+2}.$$

Thus, we get an equation:

$$\texttt{MC}(S_{i-1}^{\texttt{SR2}} \oplus S_{i+1}^{\texttt{SR2}}) = S_{i-2} \oplus S_{i+2}.$$

Define two new state variables A_i and B_i such that they satisfy the following relationship:

$$\begin{cases} A_i = S_{i-1}^{\texttt{SR2}} \oplus S_{i+1}^{\texttt{SR2}}, \\ B_i = S_{i-2} \oplus S_{i+2}. \end{cases} \tag{2}$$

According the Equation (2), we give the XOR constraint [ENP19]:

$$\begin{array}{c} A_i[t] + S_{i-1}^{\texttt{SR2}}[t] \geq S_{i+1}^{\texttt{SR2}}[t], \\ A_i[t] + S_{i+1}^{\texttt{SR2}}[t] \geq S_{i-1}^{\texttt{SR2}}[t], \\ S_{i-1}^{\texttt{SR2}}[t] + S_{i+1}^{\texttt{SR2}}[t] \geq A_i[t], \\ B_i[t] + S_{i-2}[t] \geq S_{i+2}[t], \\ B_i[t] + S_{i+2}[t] \geq S_{i-2}[t], \\ S_{i-2}[t] + S_{i+2}[t] \geq B_i[t], \end{array}$$

where $2 \leq i \leq (r-3)$, $0 \leq t \leq 15$. Meanwhile, A_i and B_i are needed to satisfy `MC` constraint model as Eq. (4):

$$\begin{array}{r} \sum_{k=0}^{3} (A_i[k+4l] + B_i[k+4l]) - 5d \geq 0, \\ d - A_i[4l] \geq 0, \\ d - A_i[1+4l] \geq 0, \\ d - A_i[2+4l] \geq 0, \\ d - A_i[3+4l] \geq 0, \\ d - B_i[4l] \geq 0, \\ d - B_i[1+4l] \geq 0, \\ d - B_i[2+4l] \geq 0, \\ d - B_i[3+4l] \geq 0, \end{array} \tag{3}$$

where auxiliary variable d is binary, and $2 \leq i \leq (r-3)$, $l = 0, 1, 2, 3$.

Through the above modeling, we have complete constraints for the differential trail propagation of `Simpira`-2. The objective function is to minimize the number of the active Sboxes for given r-round `Simpira`-2. With the same way, we can also make the MILP model for other values of b, such as $b = 3$, $b = 4$, $b = 6$, $b = 8$. We summarize all results in Table 2 and get tighter bounds for different versions of `Simpira`-b.

3.3 Improved MILP Model for Counting Active Sboxes in the Linear Trails

Now we briefly introduce the linear MILP model. For the linear trail of r-round `Simpira-2`, we denote the linear mask of the state S_i as S_i here, and the same for other states. Then we have the model as follows:

Table 2: Lower bounds on the number of active Sboxes of the differential trail of `Simpira v2` with b branches

b	Rounds															Ref.
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
2	-	-	-	-	25	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	25	30	35	40	55	60	65	70	85	90	95	100	Ours
3	-	-	-	-	-	25	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	5	5	10	30	35	35	40	55	61	70	75	80	92	Ours
4	-	-	-	-	40	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	60	65	70	90	100	120	120	125	130	Ours
6	-	-	-	25	-	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	70	115	120	130	140	160	-	-	-	Ours
8	-	-	-	25	-	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	70	120	145	-	-	-	-	-	-	Ours

Constraints of `AES` Round Function. The main constraint is in the `MC` operation, that the branch number is greater than or equal to 5. Therefore, each column and its corresponding output state meets the following conditions:

$$\begin{aligned} \sum_{k=0}^{3}(S_i^{\texttt{SR1}}[k+4l]+S_i^{\texttt{MC1}}[k+4l])-5d &\geq 0,\\ d-S_i^{\texttt{SR1}}[4l] &\geq 0,\\ d-S_i^{\texttt{SR1}}[1+4l] &\geq 0,\\ d-S_i^{\texttt{SR1}}[2+4l] &\geq 0,\\ d-S_i^{\texttt{SR1}}[3+4l] &\geq 0,\\ d-S_i^{\texttt{MC1}}[4l] &\geq 0,\\ d-S_i^{\texttt{MC1}}[1+4l] &\geq 0,\\ d-S_i^{\texttt{MC1}}[2+4l] &\geq 0,\\ d-S_i^{\texttt{MC1}}[3+4l] &\geq 0, \end{aligned} \tag{4}$$

where auxiliary variable d is binary, and $0 \leq i < r$, $l = 0, 1, 2, 3$.

Constraints on the Feistel Structure. For the linear trail of the Feistel structure, the linear mask must be satisfied at the branch XOR conditions, that is, $S_i = S_{i-1}^{\texttt{MC2}} \oplus S_{i+1}^{\texttt{MC2}}$, so we get the following XOR constraint model [ENP19]:

$$\begin{aligned} S_i[t]+S_{i-1}^{\texttt{MC2}}[t] &\geq S_{i+1}^{\texttt{MC2}}[t],\\ S_i[t]+S_{i+1}^{\texttt{MC2}}[t] &\geq S_{i-1}^{\texttt{MC2}}[t],\\ S_{i-1}^{\texttt{MC2}}[t]+S_{i+1}^{\texttt{MC2}}[t] &\geq S_i[t], \end{aligned}$$

where $1 \leq i < (r-1)$, $0 \leq t \leq 15$.

Due to the particularity of round function F of `Simpira-2` structure, we give a more accurate constraint, that is, the input of F_i function satisfies the following relationship:

$$\begin{aligned} S_i &= S_{i-1}^{\texttt{MC2}} \oplus S_{i+1}^{\texttt{MC2}}\\ &= \texttt{MC}(S_{i-1}^{\texttt{SR2}}) \oplus \texttt{MC}(S_{i+1}^{\texttt{SR2}})\\ &= \texttt{MC}(S_{i-1}^{\texttt{SR2}} \oplus S_{i+1}^{\texttt{SR2}}). \end{aligned}$$

Thus, we get an equation:

$$S_i = \texttt{MC}(S_{i-1}^{\texttt{SR2}} \oplus S_{i+1}^{\texttt{SR2}}).$$

Define a new state variables A_i and such that they satisfy the following relationship:

$$A_i = S_{i-1}^{\texttt{SR2}} \oplus S_{i+1}^{\texttt{SR2}}. \tag{5}$$

According the Equation (5), we give the XOR constraint [ENP19]:

$$\begin{gathered}
A_i[t] + S_{i-1}^{\mathtt{SR2}}[t] \geq S_{i+1}^{\mathtt{SR2}}[t], \\
A_i[t] + S_{i+1}^{\mathtt{SR2}}[t] \geq S_{i-1}^{\mathtt{SR2}}[t], \\
S_{i-1}^{\mathtt{SR2}}[t] + S_{i+1}^{\mathtt{SR2}}[t] \geq A_i[t],
\end{gathered}$$

where $1 \leq i < (r-1)$, $0 \leq t \leq 15$. Meanwhile, A_i and S_i are needed to satisfy MC constraint model as Eq. (4):

$$\begin{aligned}
\sum_{k=0}^{3} (A_i[k+4l] + S_i[k+4l]) - 5d &\geq 0, \\
d - A_i[4l] &\geq 0, \\
d - A_i[1+4l] &\geq 0, \\
d - A_i[2+4l] &\geq 0, \\
d - A_i[3+4l] &\geq 0, \\
d - S_i[4l] &\geq 0, \\
d - S_i[1+4l] &\geq 0, \\
d - S_i[2+4l] &\geq 0, \\
d - S_i[3+4l] &\geq 0,
\end{aligned} \tag{6}$$

where auxiliary variable d is binary, and $1 \leq i < (r-1)$, $l = 0, 1, 2, 3$.

Through the above modeling, we have complete constraints for the linear trail propagation of Simpira-2. The objective function is to minimize the number of the active Sboxes for given r-round Simpira-2. With the same way, we can also make the MILP model for other values of b, such as $b = 3$, $b = 4$, $b = 6$, $b = 8$. We summarize all results in Table 3 and get tighter bounds for different versions of Simpira-b. Comparing Tables 2 and 3, there are some differences. In the differential trail, the minimal numbers of active Sboxes are 80 for 14-round Simpira-3, 120 for 8-round Simpira-8, 145 for 9-round Simpira-8. In the linear trail, the minimal numbers of active Sboxes are 85 for 14-round Simpira-3, 110 for 8-round Simpira-8, 120 for 9-round Simpira-8. We think the difference may come from the different modeling methods of the differential and linear trail, where one model may be tighter than the other, rather than a difference in Simpira's properties.

Table 3: Lower bounds on the number of active Sboxes of the linear trail of Simpira v2 with b branches

b	Rounds															Ref.
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
2	-	-	-	-	25	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	25	30	35	40	55	60	65	70	85	90	95	100	Ours
3	-	-	-	-	-	25	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	5	5	10	30	35	35	40	55	61	70	75	85	92	Ours
4	-	-	-	-	40	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	60	65	70	90	100	120	120	125	130	Ours
6	-	-	-	25	-	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	70	115	120	130	140	160	-	-	-	Ours
8	-	-	-	25	-	-	-	-	-	-	-	-	-	-	-	[GM16]
	5	5	10	30	40	60	70	110	120	145	-	-	-	-	-	Ours

4 Collision Attacks on Reduced Simpira v2

With the MILP model introduced in Sect. 3.2, we derive several truncated differentials for Simpira-2 and Simpira-4. Based on them, we launch collision attacks via rebound attacks [MRST09]. Collision attacks on generic Feistel ciphers have been investigated in quite a few papers [SY11, SEHK12, DW16].

4.1 Collision Attack on 9-round Simpira-2

We introduce a new 9-round truncated differential of Simpira-2, which is applied in the collision attack, as shown in Figure 6. The inbound phase happen in round 3 and round 5. The process is as follows:

1. For each of the 2^{32} possible differences of $S_3^{\texttt{MC2}}$, set the difference of state $S_5^{\texttt{MC2}}$ and $S_3^{\texttt{MC2}}$ be equal, i.e. $\Delta S_5^{\texttt{MC2}} = \Delta S_3^{\texttt{MC2}}$.
2. For each of the 2^{32} possible differences of $S_2^{\texttt{SR2}}$, since $\Delta S_2^{\texttt{MC2}} = \Delta S_3$, we can get one pair of values on average that satisfy this trail $\Delta S_2^{\texttt{SR2}} \to \Delta S_3^{\texttt{MC2}}$ ($\Delta S_2^{\texttt{SR2}} \to \Delta S_3 \to \Delta S_3^{\texttt{SB1}} \to \Delta S_3^{\texttt{SR1}} \to \Delta S_3^{\texttt{SB2}} \to \Delta S_3^{\texttt{SR2}} \to \Delta S_3^{\texttt{MC2}}$), by applying the Super Sbox technique in Sect. 2.3. In the same way, for each of the 2^{32} possible differences $\Delta S_4^{\texttt{SR2}}$, we obtain one pair of values on average that satisfy this differential trail $\Delta S_4^{\texttt{SR2}} \to \Delta S_5^{\texttt{MC2}}$ ($\Delta S_4^{\texttt{SR2}} \to \Delta S_5 \to \Delta S_5^{\texttt{SB1}} \to \Delta S_5^{\texttt{SR1}} \to \Delta S_5^{\texttt{SB2}} \to \Delta S_5^{\texttt{SR2}} \to \Delta S_5^{\texttt{MC2}}$).
3. From the values at state S_3 and S_5, we can get the values at $S_4^{\texttt{MC2}}$, and then know all the intermediate state values of round 5. Thus, combined with the values at $S_3^{\texttt{MC2}}$ and $S_5^{\texttt{MC2}}$, we compute the values at S_2 and S_6. Then check the following equations:

$$\Delta[\texttt{SR} \circ \texttt{SB} \circ \texttt{AC} \circ \texttt{MC} \circ \texttt{SR} \circ \texttt{SB}(S_2)] \overset{?}{=} \Delta S_2^{\texttt{SR2}}. \tag{7}$$

$$\Delta[\texttt{SR} \circ \texttt{SB} \circ \texttt{AC} \circ \texttt{MC} \circ \texttt{SR} \circ \texttt{SB}(S_6)] \overset{?}{=} \Delta S_6^{\texttt{SR2}}. \tag{8}$$

   If we find a solution for the above two equations, it means that we have found a pair of the values that satisfies the truncated differential from round 2 to 8. For an arbitrary triple $(\Delta S_2^{\texttt{SR2}}, \Delta S_3^{\texttt{MC2}}, \Delta S_6^{\texttt{SR2}})$, the probability that the above two equations hold is 2^{-64}.

4. As shown in Figure 6, we have:

$$\Delta P_L = \Delta S_0 = \Delta S_2 = \Delta S_3^{\texttt{MC2}} \oplus \Delta S_4 = \Delta S_4 \oplus \Delta S_5^{\texttt{MC2}} = \Delta S_6 = \Delta S_8 = \Delta C_L.$$

$$\Delta P_R = \Delta S_0^{\texttt{MC2}} = \texttt{MC}(\Delta S_0^{\texttt{SR2}}). \tag{9}$$

$$\Delta C_R = \Delta S_8^{\texttt{MC2}} = \texttt{MC}(\Delta S_8^{\texttt{SR2}}). \tag{10}$$

   According to the Equation (9) and (10), we can get a equivalence relation as follows:

$$\Delta S_0^{\texttt{SR2}} = \Delta S_8^{\texttt{SR2}} \Leftrightarrow \Delta P_R = \Delta C_R. \tag{11}$$

   The probability that the equation $\Delta S_0^{\texttt{SR2}} = \Delta S_8^{\texttt{SR2}}$ holds is 2^{-32}. Thus, the probability for a collision between input difference and output difference in this truncated differential is $2^{-64} \times 2^{-32} = 2^{-96}$.

Degrees of Freedom. The number of values of the triple $(\Delta S_2^{\texttt{SR2}}, \Delta S_3^{\texttt{MC2}}, \Delta S_6^{\texttt{SR2}})$ is $(2^{32})^3 = 2^{96}$. Thus, we have enough degrees of freedom to obtain a collision, and the time complexity is 2^{96}.

4.2 Collision Attacks on 11-round Simpira-4

A new 11-round truncated differential of Simpira-4 is introduced in Figure 7, in view of which, we build a collision attack. Here, we consider the output of the hash $H(x)$ based on Simpira-4 is the leftmost 256 bits.

We denote the input at round i from left to right as S_i^A, S_i^B, S_i^C, S_i^D, seen in Figure 7. The 11-round collision attack process is as follows:

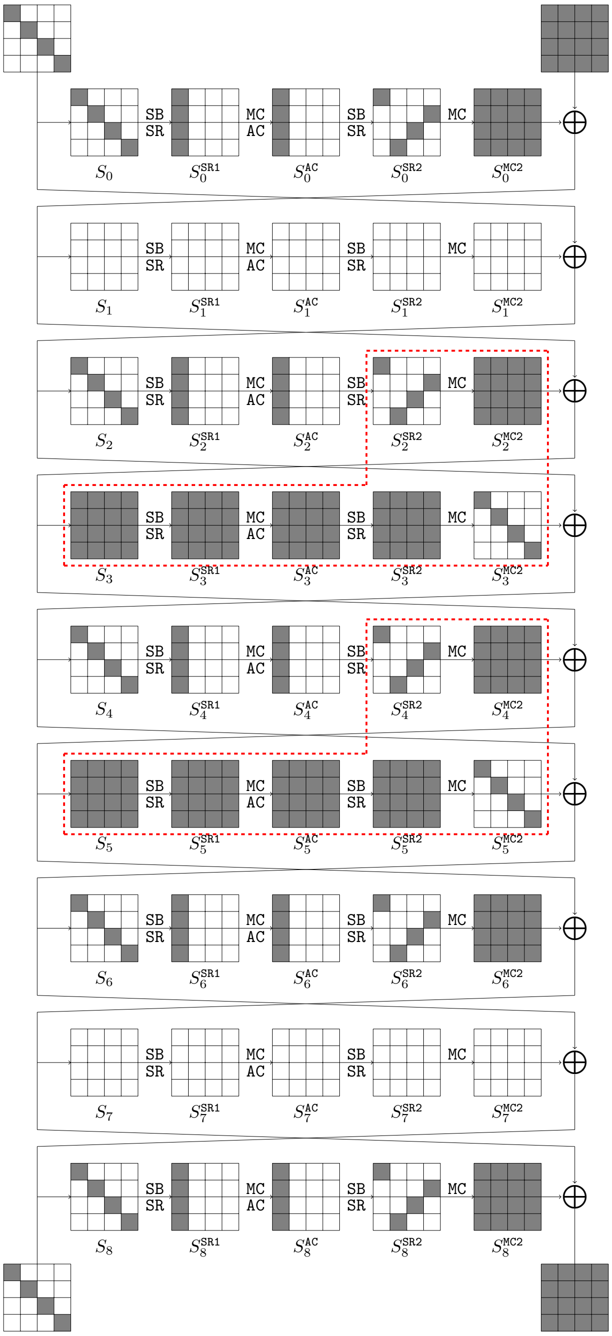

Figure 6: The 9-round collision attack on Simpira-2

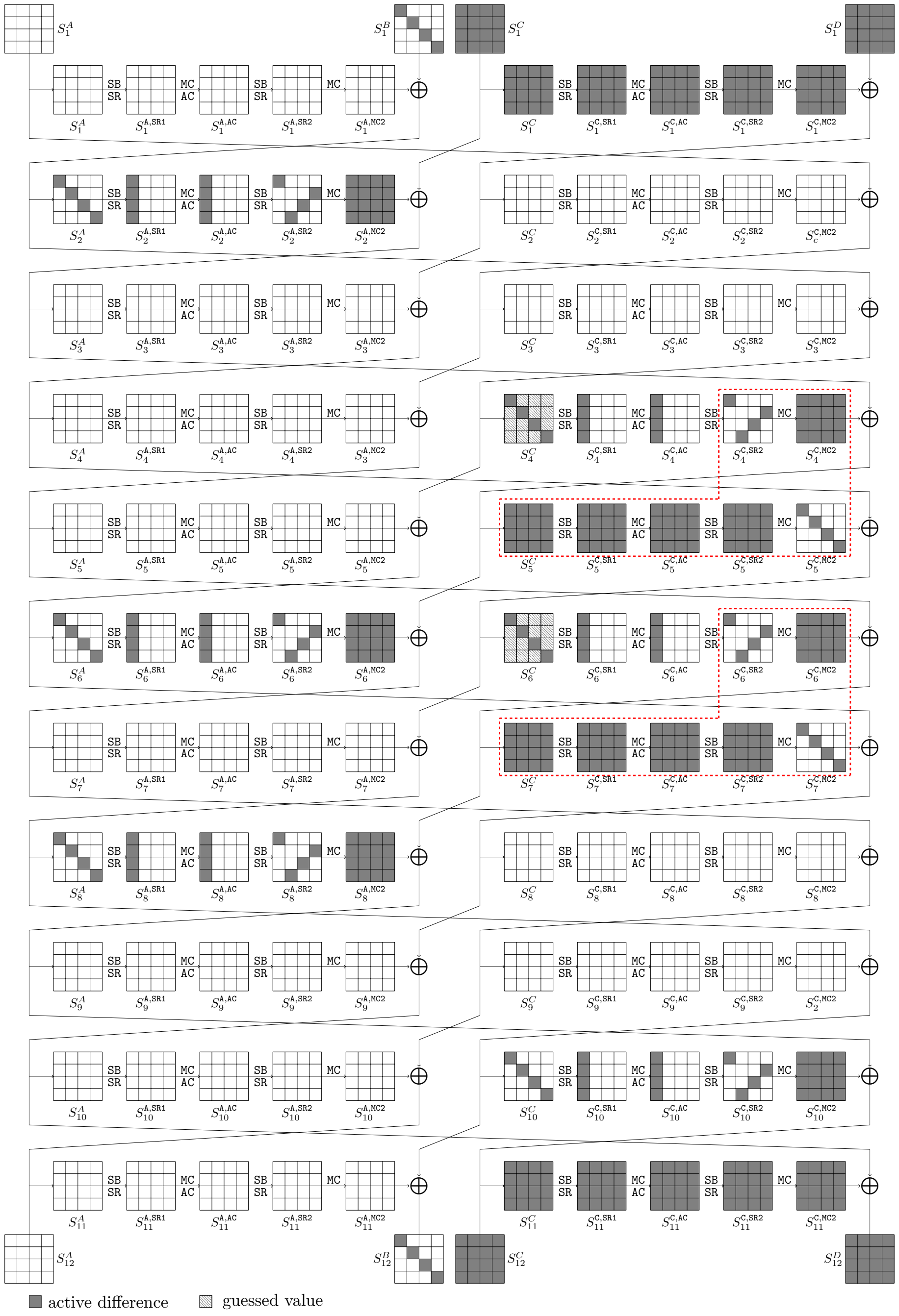

Figure 7: The 11-round collision attack on `Simpira-4`

1. For an arbitrary value of the difference ΔS_4^C, let the difference $\Delta S_5^{C,\texttt{MC2}} = \Delta S_4^C = \Delta S_7^{C,\texttt{MC2}}$.

2. For an arbitrary value of the difference $\Delta S_4^{C,\texttt{SR2}}$, since the difference $\Delta S_4^{C,\texttt{MC2}} = \Delta S_5^C$, we apply the Super-Sbox technique in Sect. 2.3, to obtain one pair of values on average that satisfies the differential trail $\Delta S_4^{C,\texttt{SR2}} \to \Delta S_5^{C,\texttt{MC2}}$ ($\Delta S_4^{C,\texttt{SR2}} \to \Delta S_5^C \to \Delta S_5^{C,\texttt{SB1}} \to \Delta S_5^{C,\texttt{SR1}} \to \Delta S_5^{C,\texttt{SB2}} \to \Delta S_5^{C,\texttt{SR2}} \to \Delta S_5^{C,\texttt{MC2}}$). In the same way, for an arbitrary difference of $S_6^{C,\texttt{SR2}}$, we can achieve one pair of values on average that is in content with the differential trail $\Delta S_6^{C,\texttt{SR2}} \to \Delta S_7^{C,\texttt{MC2}}$ ($\Delta S_6^{C,\texttt{SR2}} \to \Delta S_7^C \to \Delta S_7^{C,\texttt{SB1}} \to \Delta S_7^{C,\texttt{SR1}} \to \Delta S_7^{C,\texttt{SB2}} \to \Delta S_7^{C,\texttt{SR2}} \to \Delta S_7^{C,\texttt{MC2}}$).

3. Let Δ_0, Δ_1, Δ_2 denote the difference value of $S_4^C, S_4^{C,\texttt{SR2}}, S_6^{C,\texttt{SR2}}$, respectively. According to the property of truncated differential, we have:

$$\begin{aligned}\Delta_0 &= \Delta S_1^B = \Delta S_2^A = \Delta S_4^C = \Delta S_5^{C,\texttt{MC2}} = \Delta S_6^A \\ &= \Delta S_6^C = \Delta S_7^{C,\texttt{MC2}} = \Delta S_8^A = \Delta S_{10}^C = \Delta S_{12}^B, \\ \Delta_1 &= \Delta S_4^{C,\texttt{SR2}} = \Delta S_6^{A,\texttt{SR2}}, \\ \Delta_2 &= \Delta S_6^{C,\texttt{SR2}} = \Delta S_8^{A,\texttt{SR2}}.\end{aligned}$$

   Thus, for the difference triple $(\Delta_0, \Delta_1, \Delta_2)$, since we know that $\Delta S_4^C = \Delta_0$ and $\Delta S_4^{C,\texttt{SR2}} = \Delta_1$, we deduce a pair of 32-bit values of state S_4^C on average with the differential trail $\Delta S_4^C \to \Delta S_4^{C,\texttt{SR2}}$ (The position of the 32-bit values at state S_4^C is on the active bytes of ΔS_4^C). In the same way, we also achieve a pair of 32-bit values of S_6^A, S_6^C and S_8^A with the differential trails $\Delta S_6^A \to \Delta S_6^{A,\texttt{SR2}}$, $\Delta S_6^C \to \Delta S_6^{C,\texttt{SR2}}$ and $\Delta S_8^A \to \Delta S_8^{A,\texttt{SR2}}$, respectively.

4. There are two steps in the outbound phase.

   (a) We guess the value of the unknown bytes at state S_6^C at random, then get a pair of values at state S_6^C. According to the values at state S_6^C and S_7^C, we can get the values at state S_5^A, and deduce the value $S_5^{A,\texttt{MC2}} = S_6^A \oplus S_4^C$. Let $S[j]$ denote the j-th byte of state S, then check the following equation:

$$\texttt{MC} \circ \texttt{SR} \circ \texttt{SB} \circ \texttt{AC} \circ \texttt{MC} \circ \texttt{SR} \circ \texttt{SB}(S_5^A)[i] \overset{?}{=} S_6^A[i] \oplus S_4^C[i], \text{ where } i = 0, 5, 10, 15. \quad (12)$$

   (b) When the Equation (12) holds, we obtain a pair of values at state $S_5^{A,\texttt{MC2}}$, then we guess the value of the unknown bytes at state S_4^C randomly. According to the property of truncated differential trail, we can obtain the value at state S_7^A, then check the following equation:

$$\texttt{MC} \circ \texttt{SR} \circ \texttt{SB} \circ \texttt{AC} \circ \texttt{MC} \circ \texttt{SR} \circ \texttt{SB}(S_7^A)[i] \overset{?}{=} S_8^A[i] \oplus S_6^C[i], \text{ where } i = 0, 5, 10, 15. \quad (13)$$

   If the Equations (12) and (13) hold, we have $\Delta S_1^B = \Delta S_{12}^B$, which means that we have found a pair of the values in content with the 11-round truncated differential.

The probability that Equations (12) and (13) hold is 2^{-64}. Thus, if we guess 2^{32} values of S_4^C and S_6^C, respectively, we would obtain a collision on average, and the time complexity is 2^{64}.

Degrees of Freedom. The number of values of the triple $(\Delta S_4^C, \Delta S_4^{C,\texttt{SR2}}, \Delta S_6^{C,\texttt{SR2}})$ is $(2^{32})^3 = 2^{96}$. In addition, we can make arbitrary guessing about $2 \times 96 = 192$ bits values of S_4^C and S_6^C. Thus, we have the total degrees of freedom $(2^{96})^3 = 2^{288}$.

5 Quantum Collision Attack on 9-round `Simpira-2` without qRAM

This quantum attack consists of two phases, called inbound and outbound phases. The inbound phase is like an efficient meet-in-the-middle phase, which uses the truncated differential to satisfy the low-probability part of the middle of the differential. In the outbound phase, the matches of the inbound phase are computed backward and forward to obtain the collision attack on the hash function. In addition, our attack has a special property that the inbound phases consist of more than one inbound phase.

The core of the quantum collision attack is to apply Grover's algorithm to the search space, where the interesting elements are marked by an efficiently computable Boolean function F. Let us define the function F. We denote the instantiated input-output difference pair as $(\Delta_{in}, \Delta_{out})$ with the inbound differential. The goal of the inbound phase of a rebound attack is to generate data pairs respecting the two inbound differentials. For this two inbound phases, let the input-output difference pair $(\Delta_{in}, \Delta_{out}) = (\Delta^1_{in}, \Delta^2_{in}, \Delta_{out})$, where $(\Delta^1_{in}, \Delta_{out})$ be the input-output difference pair for the first inbound phase, and $(\Delta^2_{in}, \Delta_{out})$ be the input-output difference pair for the second inbound phase.

Given a pair $(\Delta^1_{in}, \Delta_{out})$, if there exists one input-output pair (x, x') and (y, y') that satisfies the differential trail $(\Delta^{1(i)}_{in}, \Delta^{(i)}_{out})$ for $\mathtt{SSB}^{(i)}$ for each $0 \leq i \leq 3$, there exist 8 choices for starting points for each $(\Delta^1_{in}, \Delta_{out})$. We add additional 3-bit input $\alpha = (\alpha_0, \alpha_1, \alpha_2)$ as inputs of F so that it will take that specify which starting point we choose among 8 choices. Similarly, for a pair $(\Delta^2_{in}, \Delta_{out})$, We also add additional 3-bit input $\beta = (\beta_0, \beta_1, \beta_2)$ as inputs of F. Thus, we define:

$$F : \mathbb{F}_2^{32} \times \mathbb{F}_2^{32} \times \mathbb{F}_2^{32} \times \mathbb{F}_2^{3} \times \mathbb{F}_2^{3} \rightarrow \mathbb{F}_2$$

F fulfils the backward and forward outbound differentials if and only if the starting point is computed with $(\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta)$.

Given $(\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta)$, $F(\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta)$ is computed in the classical setting by the following steps:

1. Compute the differential $(\Delta S_3^{(i)}, \Delta S_3^{\mathtt{SR2}(i)})$ for each Super Sbox $\mathtt{SSB}^{(i)}(0 \leq i < 4)$ from $(\Delta S_2^{\mathtt{SR2}(i)}, \Delta S_3^{\mathtt{MC}(i)})$, where $\Delta S_2^{\mathtt{SR2}} = \Delta^1_{in}$ and $\Delta S_3^{\mathtt{SR2}} = \Delta_{out}$.

2. Solve the active Sbox $\mathtt{SSB}^{(0)}$ to obtain $\Delta S_3^{(0)}$, such that

   $$\mathtt{SSB}^{(0)}(S_3^{(0)} \oplus \Delta S_3^{(0)}) \oplus \mathtt{SSB}^{(0)}(S_3^{(0)}) = \Delta S_3^{\mathtt{SR2}(0)}$$

   If $\alpha_0 = 0$, pick the $\min(S_3^{(0)}, S_3^{(0)} \oplus \Delta S_3^{(0)})$ as the new value for $S_3^{(0)}$. Else, pick the $\max(S_3^{(0)}, S_3^{(0)} \oplus \Delta S_3^{(0)})$ as the new value for $S_3^{(0)}$. And in the same way, we can get $S_3^{(1)}, S_3^{(2)}$. For the pair $(S_3^{(3)}, S_3^{(3)} \oplus \Delta S_3^{(3)})$, we always pick the bigger one as $S_3^{(3)}$. Thus, we obtain a starting point as

   $$S_3 = (S_3^{(0)}, S_3^{(1)}, S_3^{(2)}, S_3^{(3)}).$$

3. Compute the differential $(\Delta S_5^{(i)}, \Delta S_5^{\mathtt{SR2}(i)})$ for each Super Sbox $\mathtt{SSB}^{(i)}(0 \leq i < 4)$ from $(\Delta S_4^{\mathtt{SR2}(i)}, \Delta S_5^{\mathtt{MC}(i)})$, where $\Delta S_4^{\mathtt{SR2}} = \Delta^2_{in}$ and $\Delta S_5^{\mathtt{SR2}} = \Delta_{out} = \Delta S_3^{\mathtt{SR2}}$.

4. Solve the active Sbox $\mathtt{SSB}^{(0)}$ to obtain $\Delta S_5^{(0)}$, such that

   $$\mathtt{SSB}^{(0)}(S_5^{(0)} \oplus \triangle S_5^{(0)}) \oplus \mathtt{SSB}^{(0)}(S_5^{(0)}) = \triangle S_5^{\mathtt{SR2}(0)}$$

If $\beta_0 = 0$, pick the $\min(S_5^{(0)}, S_5^{(0)} \oplus \Delta S_5^{(0)})$ as the new value for $S_5^{(0)}$. Else, pick the $\max(S_5^{(0)}, S_5^{(0)} \oplus \Delta S_5^{(0)})$ as the new value for $S_5^{(0)}$. And in the same way, we can get $S_5^{(1)}, S_5^{(2)}$. For the pair $(S_5^{(3)}, S_5^{(3)} \oplus \Delta S_5^{(3)})$, we always pick the bigger one as $S_5^{(3)}$. Thus, we obtain a starting point as

$$S_5 = (S_5^{(0)}, S_5^{(1)}, S_5^{(2)}, S_5^{(3)}).$$

5. According to the values S_3 and S_5, we can compute S_3' and $S_5'(S_3' = S_3 \oplus \Delta S_3$, $S_5' = S_5 \oplus \Delta S_5)$, then calculate all the values of round 5. Thus, combined with the values $(S_5^{\texttt{MC2}}, S_5'^{\texttt{MC2}})$, we compute the values at (S_6, S_6') and $(S_6^{\texttt{SR2}}, S_6'^{\texttt{SR2}})$. Then check whether the equation $S_6^{SR2} \oplus S_6'^{SR2} = \Delta_{out}$ holds. If not, return to Step 3.

6. Similarly, we compute the values at (S_2, S_2') and $(S_2^{\texttt{SR2}}, S_2'^{\texttt{SR2}})$. And then check whether the equation $S_2^{\texttt{SR2}} \oplus S_2'^{\texttt{SR2}} = \Delta_{in}^1$ holds. If not, return to Step 1.

7. Note that now we have a corrected path for the starting points. If the starting point $(S_3, S_3 \oplus \Delta S_3)$ obtained in step 2 respects the forward outbound differential, and the starting point $(S_3, S_3 \oplus \Delta S_3)$ obtained in step 4 respects the backward outbound differential, then $F(\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}; \alpha, \beta)$ returns 1; otherwise it returns 0.

Therefore, by applying Grover's search with the quantum oracle U_F which maps $|\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}; \alpha, \beta\rangle|y\rangle$ to $|\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}; \alpha, \beta\rangle|y \oplus F(\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}; \alpha, \beta)\rangle$, we can find a collision with around $\frac{\pi}{4} \cdot \sqrt{2^{102}}$ queries. Next, in order to estimate the overall complexity, we need to be clear on the complexity caused by U_F.

5.1 Implementation of the Quantum Oracle U_F

Similar to [HS20, DSS+20], we need some additional functions to implement U_F. As shown in Figure 3, we define a function $G^{(i)}$ which marks the values of $X_1^{(i)}$ to solutions (compatible data pairs) for the given differential $(\Delta X_1^{(i)}, \Delta Y_2^{(i)})$ of the Super Sbox $\texttt{SSB}^{(i)}$. The implementation of the quantum oracle $U_{G^{(i)}}$ of $G^{(i)} (i = 0, 1, 2)$ is presented in Algorithm 1. And the implementation of the quantum oracle $U_{G^{(3)}}$ of $G^{(3)}$ is presented in Algorithm 2. Finally, by using $U_{G^{(i)}}$, the oracle U_F can be constructed which is presented in Algorithm 3.

5.2 Complexity Analysis.

First of all, we make several assumptions on our quantum collision attack.

- The complexity of the computation of 9-round Simpira-2 is approximated by $16 \times 2 \times 9 = 288$ Sbox computations.
- The complexity of one access to the qRAM storing a table is equivalent to one Sbox computation.
- The implementation of one inverse Sbox is about one Sbox.
- Uncomputing is taken into account.

Complexity of $G^{(i)}$. Applying Grover algorithm to $G^{(i)}$ given $(\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i)$ to find a 32-bit value $X_1^{(i)}$ requires $\frac{\pi}{4} \times \sqrt{2^{32}} \approx 2^{15.65}$ queries to the oracle $U_{G^{(i)}}$. And the $U_{G^{(i)}}$ need 16 Sbox evaluations. Thus, the overall complexity is $2 \times 2^{15.65} \times 16 \times \frac{1}{288} \approx 2^{12.45}$ 9-round Simpira-2 computations.

Complexity of U_F. In Algorithm 3, the first for loop needs 4 calls of $G^{(i)}$, the second for loop also needs 4 calls of $G^{(i)}$. we need to compute backward for 2 rounds and forward

Algorithm 1: Implementation of $U_{G^{(i)}}$

```
Input: $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i; X_1^{(i)}\rangle|y\rangle$
Output: $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i; X_1^{(i)}\rangle|y \oplus G^{(i)}(\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i; X_1^{(i)})\rangle$
if $\alpha_i = 0$ then
    Set $X_1^{(i)} \leftarrow \min(X_1^{(i)}, X_1^{(i)} \oplus \Delta X_1^{(i)})$.
    Compute $Y_2^{(i)}$ and $Y_2^{'(i)}$ from $X_1^{(i)}$ and $X_1^{(i)} \oplus \Delta X_1^{(i)}$.
    if $\mathit{SSB}^{(i)}(X_1^{(i)}) \oplus \mathit{SSB}^{(i)}(X_1^{(i)} \oplus \Delta X_1^{(i)}) = \Delta Y_2^{(i)}$ then
        return $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, X_1^{(i)}, \alpha_i\rangle|y \oplus 1\rangle$
    else
        return $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, X_1^{(i)}, \alpha_i\rangle|y\rangle$
    end
else
    Set $X_1^{(i)} \leftarrow \max(X_1^{(i)}, X_1^{(i)} \oplus \Delta X_1^{(i)})$. Compute $Y_2^{(i)}$ and $Y_2^{'(i)}$ from $X_1^{(i)}$ and
      $X_1^{(i)} \oplus \Delta X_1^{(i)}$
    if $\mathit{SSB}^{(i)}(X_1^{(i)}) \oplus \mathit{SSB}^{(i)}(X_1^{(i)} \oplus \Delta X_1^{(i)}) = \Delta Y_2^{(i)}$ then
        return $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i; X_1^{(i)}\rangle|y \oplus 1\rangle$
    else
        return $|\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i; X_1^{(i)}\rangle|y\rangle$
    end
end
```

Algorithm 2: Implementation of $U_{G^{(3)}}$

```
Input: $|\Delta X_1^{(3)}, \Delta Y_2^{(3)}; X_1^{(3)}\rangle|y\rangle$
Output: $|\Delta X_1^{(3)}, \Delta Y_2^{(3)}; X_1^{(3)}\rangle|y \oplus G^{(3)}(\Delta X_1^{(3)}, \Delta Y_2^{(3)}; X_1^{(3)})\rangle$
Set $X_1^{(3)} \leftarrow \max(X_1^{(3)}, X_1^{(3)} \oplus \Delta X_1^{(3)})$.
Compute $Y_2^{(3)}$ and $Y_2^{'(3)}$ from $X_1^{(3)}$ and $X_1^{(3)} \oplus \Delta X_1^{(3)}$.
if $\mathit{SSB}^{(3)}(X_1^{(3)}) \oplus \mathit{SSB}^{(3)}(X_1^{(3)} \oplus \Delta X_1^{(3)}) = \Delta Y_2^{(3)}$ then
    return $|\Delta X_1^{(3)}, \Delta Y_2^{(3)}; X_1^{(3)}\rangle|y \oplus 1\rangle$
else
    return $|\Delta X_1^{(3)}, \Delta Y_2^{(3)}; X_1^{(3)}\rangle|y\rangle$
end
```

Algorithm 3: Implementation of U_F

```
Input: $|\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta\rangle|y\rangle$, with $\alpha = (\alpha_0, \alpha_1, \alpha_2), \beta = (\beta_0, \beta_1, \beta_2) \in \mathbb{F}_2^3$
Output: $|\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta\rangle|y \oplus F(\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta)\rangle$
for $i \in (0, 1, 2)$ do
    Compute the differential $(\Delta S_3^{(i)}, \Delta S_3^{\texttt{SR2}(i)})$ for each Super Sbox $\texttt{SSB}^{(i)}$ from
      $(\Delta S_2^{\texttt{SR2}(i)}, \Delta S_3^{\texttt{MC}(i)})$, where $\Delta S_2^{\texttt{SR2}(i)} = \Delta^1_{in}$ and $\Delta S_3^{\texttt{SR2}(i)} = \Delta_{out}$.
    Run Grover search on the function $G^{(i)}(\Delta S_3^{(i)}, \triangle S_3^{\texttt{SR2}(i)}, \alpha_i; \cdot) : F_2^{32} \to F_2$
    Let $S_3^{(i)}$be the output.
end
Compute the differential $(\Delta S_3^{(3)}, \Delta S_3^{\texttt{SR2}(3)})$ for each Super Sbox $\texttt{SSB}^{(3)}$ from
  $(\Delta S_2^{\texttt{SR2}(3)}, \Delta S_3^{\texttt{MC}(3)})$, where $\Delta S_2^{\texttt{SR2}(3)} = \Delta^1_{in}$ and $\Delta S_3^{\texttt{SR2}(3)} = \Delta_{out}$.
Run Grover search on the function $G^{(3)}(\Delta S_3^{(3)}, \triangle S_3^{\texttt{SR2}(3)}; \cdot) : F_2^{32} \to F_2$
Let $S_3^{(3)}$be the output.
for $i \in (0, 1, 2)$ do
    Compute the differential $(\Delta S_5^{(i)}, \Delta S_5^{\texttt{SR2}(i)})$ for each Super Sbox $\texttt{SSB}^{(i)}$ from
      $(\Delta S_4^{\texttt{SR2}(i)}, \Delta S_5^{\texttt{MC}(i)})$, where $\Delta S_4^{\texttt{SR2}(i)} = \Delta^1_{in}$ and $\Delta S_5^{\texttt{SR2}(i)} = \Delta_{out}$.
    Run Grover search on the function $G^{(i)}(\Delta S_5^{(i)}, \Delta S_5^{\texttt{SR2}(i)}, \beta_i; \cdot) : F_2^{32} \to F_2$
    Let $S_5^{(i)}$be the output.
end
Compute the differential $(\Delta S_5^{(3)}, \Delta S_5^{\texttt{SR2}(3)})$ for each Super Sbox $\texttt{SSB}^{(3)}$ from
  $(\Delta S_4^{\texttt{SR2}(3)}, \Delta S_5^{\texttt{MC}(3)})$, where $\Delta S_4^{\texttt{SR2}(3)} = \Delta^1_{in}$ and $\Delta S_5^{\texttt{SR2}(3)} = \Delta_{out}$.
Run Grover search on the function $G^{(3)}(\Delta S_5^{(3)}, \Delta S_5^{\texttt{SR2}(3)}; \cdot) : F_2^{32} \to F_2$
Let $S_5^{(3)}$be the output. Set $S_3 \leftarrow (S_3^{(0)}, S_3^{(1)}, S_3^{(2)}, S_3^{(3)})$ and
  $S_5 \leftarrow (S_5^{(0)}, S_5^{(1)}, S_5^{(2)}, S_5^{(3)})$
Set $S'_3 \leftarrow S_3 \oplus \Delta S_3$ and $S'_5 \leftarrow S_5 \oplus \Delta S_5$
Compute $(S_4, S'_4)$ from values $(S_3, S'_3)$ and $(S_5, S'_5)$
Compute $(S_2, S'_2)$ from values $(S_3, S'_3)$ and $(S_4, S'_4)$
Compute $(S_6, S'_6)$ from values $(S_4, S'_4)$ and $(S_5, S'_5)$
Compute $(S_2^{\texttt{SR2}}, S_2'^{\texttt{SR2}})$ from values $(S_2, S'_2)$
Compute $(S_6^{\texttt{SR2}}, S_6'^{\texttt{SR2}})$ from values $(S_6, S'_6)$
if $S_2^{\texttt{SR2}} \oplus S_2'^{\texttt{SR2}} = \Delta^1_{in}$ and $S_6^{\texttt{SR2}} \oplus S_6'^{\texttt{SR2}} = \Delta_{out}$ then
    set 1-bit sign $s_0 = 1$
else
    $s_0 = 0$
end
if $(S_2'^{\texttt{SR2}}, S_2'^{\texttt{SR2}}, S_6^{\texttt{SR2}}, S_6'^{\texttt{SR2}})$ fulfills the outbound differential and $s_0 = 1$ then
    return $|\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta\rangle|y \oplus 1\rangle$
else
    return $|\Delta^1_{in}, \Delta^2_{in}, \Delta_{out}; \alpha, \beta\rangle|y\rangle$
end
```

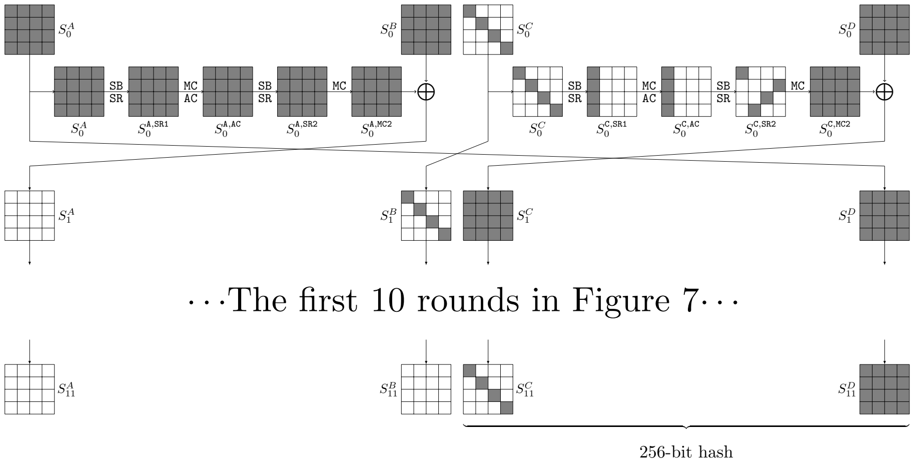

Figure 8: Differential for collision on the last two branches without the last network twist

for 2 rounds to obtain $(S_2^{\texttt{SR2}}, S_6^{\texttt{SR2}})$, and we need to compute backward for 2 rounds and forward for 2 rounds to fulfills the outbound differential. Therefore, we need $\frac{16\times4\times2\times6}{288} = \frac{8}{3}$ 9-round `Simpira`-2 computations. The overall complexity of U_F is $\frac{8}{3} + 8 \times 2^{12.45} \approx 2^{15.45}$ 9-round `Simpira`-2 computations.

Complexity to find a collision. In order to identify a 102-bit value $(\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}, \alpha, \beta) \in F_2^{32} \times F_2^{32} \times F_2^{32} \times F_2^3 \times F_2^3$ with Grover's search such that $F(\Delta_{in}^1, \Delta_{in}^2, \Delta_{out}; \alpha, \beta) = 1$, it requires $\frac{\pi}{4} \cdot \sqrt{2^{102}}$ queries to U_F. Therefore, the complexity to find a collision is $\frac{\pi}{4} \cdot \sqrt{2^{102}} \times 2^{15.45} = 2^{66.1}$ 9-round `Simpira`-2 computations.

6 Quantum Collision Attack on 11-round `Simpira-4` without qRAM

The quantum attack on `Simpira-4` is based on the differential trails shown in Figure 7. We also define a similar quantum oracle U'_F as defined in Algorithm 3 and the same $G(i)$ as Algorithm 1. Then, we apply Grover algorithm to the quantum oracle U'_F which is constructed without qRAM with similar techniques shown in previous sections.

Complexity Analysis. Here the complexity of the computation of 11-round `Simpira`-4 is approximated by $16 \times 2 \times 2 \times 11 = 704$ Sbox computations.

Complexity of $G^{(i)}$. Applying Grover algorithm to $G^{(i)}$ given $(\Delta X_1^{(i)}, \Delta Y_2^{(i)}, \alpha_i)$ to find a 32-bit value $X_1^{(i)}$ requires $\frac{\pi}{4} \times \sqrt{2^{32}} \approx 2^{15.65}$ queries to the oracle $U_{G^{(i)}}$, and the $U_{G^{(i)}}$ need 16 Sbox evaluations. Thus, the overall complexity is $2 \times 2^{15.65} \times 16 \times \frac{1}{704} \approx 2^{11.2}$ 11-round Simpira-4 computations.

Complexity of U'_F. Obviously, in the Algorithm U'_F, the complexity of the analysis is similar to U_F in Section 5. The overall complexity of U_F depends largely on the complexity of $G^{(i)}$. Since the Algorithm U'_F needs 16 calls to $G^{(i)}$, the total complexity is about $16 \times 2^{11.2} = 2^{15.2}$ 11-round `Simpira`-4 computations.

Complexity to find a collision. By applying Grover's search with the quantum oracle U'_F, we can find a collision with around $\frac{\pi}{4} \cdot \sqrt{2^{70}}$ queries to U'_F. Therefore, the complexity to find a collision is $\frac{\pi}{4} \cdot \sqrt{2^{70}} \times 2^{15.2} \approx 2^{49.85}$ 11-round `Simpira`-4 computations.

7 Collision Attacks on Other Variants of Simpira-4

The designers of Simpira commented that "To match the intended application, padding of the input and/or truncation of the output of Simpira may be required". Thus, it makes sense to consider different truncated versions of the hash construction. For Simpira-4, the four branches are denoted as (A, B, C, D). "Collision $[A, B]$" means the collision happens in branch A and B. In Section 4.2, we introduce a differential trail as shown in Figure 7. In this section, we will discuss the collision attacks on different truncations of the output of Simpira-4 plugged in the Davies-Meyer construction. The details of each case are as follows:

- **Collision on branches $[B, C]$.** First the equation $\Delta S_1^B = \Delta S_{12}^B$ holds with probability 2^{-64}, according to the property of truncated differential trail, we have the following two equations:

$$\Delta S_1^C = \Delta S_2^{A,\mathtt{MC2}} = \mathtt{MC}(\Delta S_2^{A,\mathtt{SR2}}),$$
$$\Delta S_{12}^C = \Delta S_{10}^{C,\mathtt{MC2}} = \mathtt{MC}(\Delta S_{10}^{C,\mathtt{SR2}}).$$

  Thus, the equation $\Delta S_1^C = \Delta S_{11}^C$ holds with probability 2^{-32}, and we get a 256-bit collision on branches $[B, C]$ with probability $2^{-64} \times 2^{-32} = 2^{-96}$. Then we get the complexity of this collision attack: In the classical setting, the time complexity is 2^{96}; In the quantum setting, the time complexity is $2^{65.85}$.

- **Collision on branches $[A, C]$.** First the equation $\Delta S_1^A = \Delta S_{12}^A$ holds with probability 2^{-64}. Similar to the collision on branches $[B, C]$, We obtain a 256-bit collision on branches $[A, C]$ with probability $2^{-64} \times 2^{-32} = 2^{-96}$. The complexity of this collision attack, in the classical setting, is 2^{96}, and in the quantum setting, is $2^{65.85}$.

- **Collision on branches $[A, D]$.** First the equation $\Delta S_1^A = \Delta S_{12}^A$ holds with probability 2^{-64}. Since the Sboxes of the states ΔS_1^D and ΔS_{12}^D are fully active, the equation $\Delta S_1^D = \Delta S_{12}^D$ holds with 2^{-128}. Thus, we can obtain a 256-bit collision on branches $[A, D]$ probability $2^{-64} \times 2^{-128} = 2^{-192}$. In classical setting, the collision attack will need 2^{192} which is weaker than birthday attack. However, in quantum setting, the time complexity is about $2^{113.85}$ without qRAM and classical memory, which is better than the quantum version parallel rho algorithm [HS20]. For CNS algorithm [CNS17], it needs $2^{102.4}$ time and $2^{51.2}$ classical memory.

- **Collision on branches $[B, D]$.** This case is similar to the collision on branches $[A, D]$. We can obtain a 256-bit Collision $[B, D]$ with probability $2^{-64} \times 2^{-128} = 2^{-192}$. The classical collision attack is invalid. The quantum collision attack needs a time complexity of $2^{113.85}$ without qRAM.

- **Collision on branches $[C, D]$.** In this case, the probability to obtain a 256-bit collision is $2^{-64} \times 2^{-32} \times 2^{-128} = 2^{-224}$. Both the classical collision attack and quantum collision are weaker than generic attacks.

In order to obtain a better differential trail for the collision attack on the last two branches, we make a slight change to Figure 7 by adding one round before first round and peeling off its last round. Then, we obtain a new 11-round differential trail in Figure 8. Obviously, the equation $\Delta S_0^C = \Delta S_{11}^C$ holds with 2^{-64} following the analysis in Section 4.2. According to the property of truncated differential trail, we have the following two equations:

$$\begin{aligned}\Delta S_{11}^{D} &= \Delta S_{10}^{C,\texttt{MC2}} = \texttt{MC}(\Delta S_{10}^{C,\texttt{SR2}}),\\ \Delta S_{0}^{D} &= \Delta S_{0}^{C,\texttt{MC2}} \oplus \Delta S_{1}^{C}\\ &= \Delta S_{0}^{C,\texttt{MC2}} \oplus \Delta S_{2}^{A,\texttt{MC2}}\\ &= \texttt{MC}(\Delta S_{0}^{C,\texttt{SR2}}) \oplus \texttt{MC}(\Delta S_{2}^{A,\texttt{SR2}})\\ &= \texttt{MC}(\Delta S_{0}^{C,\texttt{SR2}} \oplus \Delta S_{2}^{A,\texttt{SR2}}).\end{aligned}$$

Thus, the equation $\Delta S_0^D = \Delta S_{11}^D$ holds with 2^{-32}, and we get a 256-bit hash collision of the output $[C, D]$ branches with the probability of $2^{-64} \times 2^{-32} = 2^{-96}$. The probability is $2^{-64} \times 2^{-32} = 2^{-96}$. The time complexity to find the collision is 2^{96} in the classical setting and $2^{65.85}$ in the quantum setting (without qRAM).

8 Discussion and Conclusion

In this paper, we provided cryptanalytic results on `Simpira v2`, which is an `AES`-based permutation proposed by Gueron and Mouha at ASIACRYPT 2016. We study the MILP model to search the minimum of the active Sboxes for differential and linear trail, and give new constraints on the Feistel structure of `Simpira v2`, which eliminate some impossible truncated trails due to internal contradiction. With the updated MILP model, we renovate the low bound of the number of active Sboxes for `Simpira v2`. Furthermore, some new truncated differentials for `Simpira`-2 and `Simpira`-4 are presented, which are adopted to construct a series collision attacks on the both reduced versions of `Simpira v2` with `Davies-Meyer` hashing mode in both classical and quantum setting.

Recently, Gueron and Mouha suggested a post-quantum public key algorithm based on SPHINCS [BHH+15] and `Simpira v2`, named as SPHINCS-`Simpira` [GM17]. The security of SPHINCS [BHH+15] is based on the preimage-resistance of its internal hash function. Hence, our collision attacks do not affect the security of SPHINCS-`Simpira`.

Acknowledgments

We would like to thank Xavier Bonnetain and the anonymous reviewers from ToSC for their detailed comments. This paper was supported by the Major Program of Guangdong Basic and Applied Research (Grant No. 2019B030302008), the National Key Research and Development Program of China (Grant Nos. 2018YFA0704701 and 2017YFA0303903), the National Natural Science Foundation of China (Grant Nos. 62072270, 61902207 and 62072207), Shandong Province Key Research and Development Project(Grant Nos. 2020ZLYS09 and 2019JZZY010133), National Cryptography Development Fund (Grant Nos. MMJJ20180101 and MMJJ20170121), Natural Science Foundation of Shanghai (Grant No. 19ZR1420000), and Open Foundation of Network and Data Security Key Laboratory of Sichuan Province (University of Electronic Science and Technology of China).

References

[Ber] Daniel J. Bernstein. Cost analysis of hash collisions: will quantum computers make sharcs obsolete? *SHARCS (2009).*

[BHH+15] Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. SPHINCS: practical stateless hash-based signatures. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 368–397. Springer, 2015.

[BHN+19] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum attacks without superposition queries: The offline simon's algorithm. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 552–583. Springer, 2019.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum cryptanalysis of hash and claw-free functions. In *LATIN '98, Campinas, Brazil, April, 20-24, 1998, Proceedings*, pages 163–169, 1998.

[BN10] Alex Biryukov and Ivica Nikolic. Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to aes, camellia, khazad and others. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 322–344. Springer, 2010.

[BNS19a] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. On quantum slide attacks. In Kenneth G. Paterson and Douglas Stebila, editors, *Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers*, volume 11959 of *Lecture Notes in Computer Science*, pages 492–519. Springer, 2019.

[BNS19b] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. Quantum security analysis of AES. *IACR Trans. Symmetric Cryptol.*, 2019(2):55–93, 2019.

[CNS17] André Chailloux, María Naya-Plasencia, and André Schrottenloher. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *Lecture Notes in Computer Science*, pages 211–240. Springer, 2017.

[DEM16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Cryptanalysis of Simpira v1. In Roberto Avanzi and Howard M. Heys, editors, *Selected Areas in Cryptography - SAC 2016 - 23rd International Conference, St. John's, NL, Canada, August 10-12, 2016, Revised Selected Papers*, volume 10532 of *Lecture Notes in Computer Science*, pages 284–298. Springer, 2016.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned rebound attack: Application to Keccak. In *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, pages 402–421, 2012.

[DSS+20] Xiaoyang Dong, Siwei Sun, Danping Shi, Fei Gao, Xiaoyun Wang, and Lei Hu. Quantum collision attacks on AES-like hashing with low quantum random access memories. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 727–757. Springer, 2020.

[DW16] Xiaoyang Dong and Xiaoyun Wang. Chosen-key distinguishers on 12-round Feistel-SP and 11-round collision attacks on its hashing modes. *IACR Trans. Symmetric Cryptol.*, 2016(1):13–32, 2016.

[ENP19] Maria Eichlseder, Marcel Nageler, and Robert Primas. Analyzing the linear keystream biases in AEGIS. *IACR Trans. Symmetric Cryptol.*, 2019(4):348–368, 2019.

[FJP13] Pierre-Alain Fouque, Jérémy Jean, and Thomas Peyrin. Structural evaluation of AES and chosen-key distinguisher of 9-round AES-128. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 183–203. Springer, 2013.

[GLMS18] David Gérault, Pascal Lafourcade, Marine Minier, and Christine Solnon. Revisiting AES related-key differential attacks with constraint programming. *Inf. Process. Lett.*, 139:24–29, 2018.

[GM16] Shay Gueron and Nicky Mouha. Simpira v2: A family of efficient permutations using the AES round function. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 95–125, 2016.

[GM17] Shay Gueron and Nicky Mouha. SPHINCS-Simpira: Fast stateless hash-based signatures with post-quantum security. *IACR Cryptol. ePrint Arch.*, 2017:645, 2017.

[GNS18] Lorenzo Grassi, María Naya-Plasencia, and André Schrottenloher. Quantum algorithms for the k-xor problem. In *ASIACRYPT 2018, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, pages 527–559, 2018.

[GP10] Henri Gilbert and Thomas Peyrin. Super-Sbox cryptanalysis: Improved attacks for AES-like permutations. In *FSE 2010, Seoul, Korea, February 7-10, 2010*, pages 365–383, 2010.

[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, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219, 1996.

[HS18a] Akinori Hosoyamada and Yu Sasaki. Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In Nigel P. Smart, editor, *Topics in Cryptology - CT-RSA 2018 - The Cryptographers' Track at the RSA Conference 2018, San Francisco, CA, USA, April 16-20, 2018, Proceedings*, volume 10808 of *Lecture Notes in Computer Science*, pages 198–218. Springer, 2018.

[HS18b] Akinori Hosoyamada and Yu Sasaki. Quantum Demiric-Selçuk meet-in-the-middle attacks: Applications to 6-round generic Feistel constructions. In *SCN 2018, Amalfi, Italy, September 5-7, 2018*, pages 386–403, 2018.

[HS20] Akinori Hosoyamada and Yu Sasaki. Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 249–279. Springer, 2020.

[JNP12] Jérémy Jean, María Naya-Plasencia, and Thomas Peyrin. Improved rebound attack on the finalist Grøstl. In *FSE 2012, Washington, DC, USA, March 19-21, 2012*, pages 110–126, 2012.

[JNP13] Jérémy Jean, María Naya-Plasencia, and Thomas Peyrin. Multiple limited-birthday distinguishers and applications. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 533–550. Springer, 2013.

[JNS11] Jérémy Jean, María Naya-Plasencia, and Martin Schläffer. Improved analysis of ECHO-256. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 2011.

[KLLN16a] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 207–237. Springer, 2016.

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

[KLMR16] Stefan Kölbl, Martin M. Lauridsen, Florian Mendel, and Christian Rechberger. Haraka v2 - efficient short-input hashing for post-quantum applications. *IACR Trans. Symmetric Cryptol.*, 2016(2):1–29, 2016.

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

[KM12] Hidenori Kuwakado and Masakatu Morii. Security on the quantum-type Even-Mansour cipher. In *ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*, pages 312–316, 2012.

[Köl18] Stefan Kölbl. Putting wings on SPHINCS. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, volume 10786 of *Lecture Notes in Computer Science*, pages 205–226. Springer, 2018.

[LM17] Gregor Leander and Alexander May. Grover meets simon - quantumly attacking the FX-construction. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *Lecture Notes in Computer Science*, pages 161–178. Springer, 2017.

[LMR+09] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. Rebound distinguishers: Results on the full Whirlpool compression function. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2009.

[MNN+09] Krystian Matusiewicz, María Naya-Plasencia, Ivica Nikolic, Yu Sasaki, and Martin Schläffer. Rebound attack on the full Lane compression function. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, pages 106–125, 2009.

[MPRS09] Florian Mendel, Thomas Peyrin, Christian Rechberger, and Martin Schläffer. Improved cryptanalysis of the reduced Grøstl compression function, ECHO permutation and AES block cipher. In *Selected Areas in Cryptography, 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009, Revised Selected Papers*, pages 16–35, 2009.

[MRS14] Florian Mendel, Vincent Rijmen, and Martin Schläffer. Collision attack on 5 rounds of Grøstl. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *Lecture Notes in Computer Science*, pages 509–521. Springer, 2014.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In *FSE 2009, Leuven, Belgium, February 22-25, 2009*, pages 260–276, 2009.

[Nay11] María Naya-Plasencia. How to improve rebound attacks. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 188–205, 2011.

[NS20] María Naya-Plasencia and André Schrottenloher. Optimal merging in quantum k-xor and k-xor-sum algorithms. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 311–340. Springer, 2020.

[Røn16] Sondre Rønjom. Invariant subspaces in Simpira. *IACR Cryptol. ePrint Arch.*, 2016:248, 2016.

[RZW18] Xiaoyang Dong Rui Zong and Xiaoyun Wang. Impossible differential attack on Simpira v2. *Science China Information Sciences*, 61:032106:1–032106:13, 2018.

[SEHK12] Yu Sasaki, Sareh Emami, Deukjo Hong, and Ashish Kumar. Improved known-key distinguishers on Feistel-SP ciphers and application to Camellia. In *Information Security and Privacy - 17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, July 9-11, 2012. Proceedings*, pages 87–100, 2012.

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

[SLW$^+$10] Yu Sasaki, Yang Li, Lei Wang, Kazuo Sakiyama, and Kazuo Ohta. Non-full-active Super-Sbox analysis: Applications to ECHO and Grøstl. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, pages 38–55, 2010.

[SY11] Yu Sasaki and Kan Yasuda. Known-key distinguishers on 11-round Feistel and collision attacks on its hashing modes. In *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, pages 397–415, 2011.

[THW17] Ivan Tjuawinata, Tao Huang, and Hongjun Wu. Cryptanalysis of Simpira v2. In Josef Pieprzyk and Suriadi Suriadi, editors, *Information Security and Privacy - 22nd Australasian Conference, ACISP 2017, Auckland, New Zealand, July 3-5, 2017, Proceedings, Part I*, volume 10342 of *Lecture Notes in Computer Science*, pages 384–401. Springer, 2017.

[vOW94] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with application to hash functions and discrete logarithms. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, and Ravi S. Sandhu, editors, *CCS '94, Proceedings of the 2nd ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 2-4, 1994*, pages 210–218. ACM, 1994.