Improved Attacks on sLiSCP Permutation and Tight Bound of Limited Birthday Distinguishers

Akinori Hosoyamada[1,3], María Naya-Plasencia[2] and Yu Sasaki[1]

[1] NTT Secure Platform Laboratories, Tokyo, Japan,
{akinori.hosoyamada.bh,yu.sasaki.sk}@hco.ntt.co.jp
[2] Inria, Paris, France, maria.naya_plasencia@inria.fr
[3] Nagoya University, Nagoya, Japan hosoyamada.akinori@nagoya-u.jp

Abstract. Limited birthday distinguishers (LBDs) are widely used tools for the cryptanalysis of cryptographic permutations. In this paper we propose LBDs on several variants of the `sLiSCP` permutation family that are building blocks of two round 2 candidates of the NIST lightweight standardization process: Spix and `SpoC`. We improve the number of steps with respect to the previously known best results, that used rebound attack. We improve the techniques used for solving the middle part, called inbound, and we relax the external conditions in order to extend the previous attacks.
The lower bound of the complexity of LBDs has been proved only against functions. In this paper, we prove for the first time the bound against permutations, which shows that the known upper bounds are tight.

Keywords: limited birthday distinguisher · sLiSCP · permutation · NIST Lightweight cryptography · rebound attack

1 Introduction

Lightweight cryptography aims at providing an efficient cryptographic primitive on highly-constrained devices such as sensor networks, distributed control systems, the Internet of Things, and so on. Recently, the National Institute of Standards and Technology (NIST) initiated a lightweight cryptography standardization process [Nat19] to select and standardize several lightweight cryptographic algorithms. In April 2019, 56 algorithms were announced as round 1 candidates and in August 2019, 32 algorithms were selected as round 2 candidates. NIST had originally planned to announce about 8 round 3 candidates in September 2020, but this announcement has been delayed a few months. Given the situation, improved security analysis of round 2 candidates is very important.

Design of a cryptographic algorithm that simultaneously achieves high security and lightweight implementation properties is a challenging task. A recent trend [1] is to design a cryptographic permutation as an underlying primitive, and to build an authenticated encryption with associated data (AEAD) with the duplex construction [BDPA11]. This approach is also advantageous to additionally implement a cryptographic hash function only with a small overhead. In fact, NIST reported that 49% of the round 1 candidates and 50% of the round 2 candidates are based on a permutation [TMÇ+19].

A cryptographic permutation is expected to behave as a uniformly random permutation. From an attacker's position, the goal is to find a specific behavior that differs between the target permutation and a random permutation. The attacker first specifies a certain relationship for a set of inputs and the corresponding outputs, and then compares the

[1] See for instance [Dae17]

Figure 1: Step Function of `sLiSCP` (left) and `sLiSCP-light` (right) Permutations. Simeck^r_w denotes r-rounds of w-bit block Simeck.

complexity, i.e. computational cost and memory amount, to find such a set for the target algorithm and a randomly chosen permutation. A limited birthday distinguisher (LBD) [GP10] is a natural application of differential cryptanalysis to permutations. The attacker specifies a set of input differences and a set of output differences. The attacker's goal is to find a pair of texts that confirm both of the input and output differences.

In this paper, we provide the cryptanalysis for `sLiSCP` [ARH+17] and `sLiSCP-light` permutations [ARH+18]. `sLiSCP` is a cryptographic permutation based on Simeck [YZS+15]. `sLiSCP` was designed to be used in their sponge hash function and duplex AEAD mode. `sLiSCP` consists of 18 iterations of the step function that adopts a 4-branch type-2 generalized Feistel network (GFN) in which the size of each branch w is $w \in \{48, 64\}$. The whole permutation size is 192 bits or 256 bits, which is called `sLiSCP-192` and `sLiSCP-256`. The step function is illustrated in the left-hand side of Fig. 1. The step function of `sLiSCP-192` (resp. `sLiSCP-256`) computes two unkeyed 6-round Simeck48 (resp. 8-round Simeck64).

The same designers later presented a tweaked version called `sLiSCP-light`. The major difference from `sLiSCP` is that the GFN is replaced with the partial substitution permutation network (PSPN) [ARH+18] illustrated in the right-hand side of Fig. 1. The recommended number of steps of `sLiSCP-light` was also reduced from 18 to 12.

`sLiSCP-light` is used as an underlying primitive of two round 2 candidates in NIST's standardization process. `SpoC` [AGH+19a] builds an AEAD scheme with the duplex-like framework using 18-step `sLiSCP-light-192` and 18-step `sLiSCP-light-256` as underlying permutations. Spix [AGH+19b] also builds an AEAD scheme with the duplex framework and 18-step `sLiSCP-light-256` is used to process the key material while 9-step `sLiSCP-light-256` is used to process associated data and message/ciphertext. The active usage of `sLiSCP-light` shows the importance of third-party security analysis.

To the best of our knowledge, there exists only a single third-party security analysis against `sLiSCP` [LSSW18] and no third-party analysis exists against `sLiSCP-light`. Liu et al. [LSSW18] provided a forgery attack and a collision attack against 6-step `sLiSCP` in the AEAD mode and the hash mode. In addition, a LBD was presented against 15-step `sLiSCP` permutation. The designers of `sLiSCP`, `sLiSCP-light`, `SpoC`, and Spix also provided some cryptanalysis for the permutation, which includes impossible differential, zero-correlation and integral distinguishers against 9-step `sLiSCP`. [2]

After the submission of this work, Kraleva, Posteuca and Rijmen uploaded the analysis on `SpoC` to Cryptology ePrint Archive [KPR20], which was later presented at the Fourth Lightweight Cryptography Workshop organized by NIST. Soon after, the SpoC team posted a message to the NIST LWC forum [Tea20] to report that the attacks and observations in [KPR20] do not pose any threats to the security of `SpoC` and its underlying permutation `sLiSCP-light`.

In the design document of `SpoC` [AGH+19a] and Spix [AGH+19b], the designers claim that they aim to provide the evidence that 18-step `sLiSCP-light` is secure against various

[2] The authors of [ARH+17] reported a 17-step zero-sum distinguisher for `sLiSCP-192` and `sLiSCP-256` with very high complexities (2^{190} and 2^{255}) without discussing the generic attack complexity to satisfy the same property. Thanks to an ongoing discussion with the authors, we believe now that the generic complexity would be better, so we are not convinced of the validity of this distinguisher.

distinguishing attacks to prove that its behavior is as close as possible to that of an ideal permutation. Hence we believe that improving the previous permutation distinguishers for `sLiSCP` and providing a new analysis on `sLiSCP-light` is of great interest.

Remarks on the Permutation Distinguisher Framework. As explained in [IPS13], one may argue that the LBD can trivially be solved for any permutation Π by choosing any input pair (X, Y) and computing $X \oplus Y$ as the set of input difference and $\Pi(X) \oplus \Pi(Y)$ as the set of the output difference. In general, those meaningless attacks are avoided by considering that a hash function is part of a family indexed by a key input (e.g. IV is replaced with a key). In our case, the attack works even if a constant in Simeck boxes is given right before the attack starts. Hence, our attacks are not meaningless.

Our Contributions. The contribution of this paper is twofold. First, we improve the best known attacks against `sLiSCP` and present the first third-party cryptanalysis against `sLiSCP-light`. Second, we prove the lower bound of the complexity to solve LBDs for a random permutation, showing that the current best known generic attack is actually tight.

Limited-Birthday Distinguishers against `sLiSCP` and `sLiSCP-light`. We first reduce the complexity of the 15-step LBDs for `sLiSCP`, which is computed in the rebound-like procedure [MRST09]. By carefully analyzing the computation order, we extend the number of steps that can be covered by the inbound phase, which reduces the complexity from $2^{122.7}$ to $2^{111.4}$ for `sLiSCP-192` and from $2^{168.3}$ to $2^{149.6}$ for `sLiSCP-256`.

Even with this complexity improvement, the attack cannot be extended to 16 steps easily because the remaining degrees of freedom are insufficient to satisfy the differential propagation for another step. Here, we look into the differential characteristics for Simeck48 and Simeck64 and try to make many bits inactive by spending a small amount of degrees of freedom. This allows us to attack 16-step `sLiSCP-256` with $2^{154.6}$ cost.

For `sLiSCP-light`, the designers of Spix and `SpoC` argued that the best known distinguisher is a zero-sum distinguisher with a start-from-the-middle approach, which works up to 14 steps but requires data and time complexities equal to that of the exhaustive search. Although the designers of Spix and `SpoC` cited the work by Liu et al. [LSSW18], no word is given on the possibility of applying the LBDs on `sLiSCP` to `sLiSCP-light`. In this paper we formally claim, for the first time, that 16-step `sLiSCP-light` can be attacked, using a similar procedure to the one on `sLiSCP`. Besides, the replacement of the Feistel network of `sLiSCP` with the partial SPN allows us to attack 16 steps of `sLiSCP-light-192` with a complexity of $2^{113.0}$. The comparison of the attack complexities is given in Table 1.

Tight Lower Bound for the Limited-Birthday Problem. We show that the upper bound given by the known best algorithm on the limited birthday problem for a random permutation is asymptotically tight.

As we mentioned before, the goal of a LBD is to find a pair of texts that confirm both of the input and output differences that are specified by the attacker. More precisely, the *limited birthday problem* on an n-bit permutation P and closed subsets Δ_{in}, Δ_{out} is the problem of finding a tuple (X, X', Y, Y') such that $P(X) = Y$, $P(X') = Y'$, $X \oplus X' \in \Delta_{\text{in}}$, and $Y \oplus Y' \in \Delta_{\text{out}}$. Here, a non-empty subset $\Delta \subset \{0,1\}^n$ is *closed* if and only if $X \oplus Y \in \Delta$ for all $X, Y \in \Delta$. An algorithm to solve the problem is called a *limited-birthday distinguisher* (LBD), or simply *distinguisher*.

The known best distinguisher to solve the problem when P is a random permutation is

$$\max\left\{\min\left\{\sqrt{\frac{2^n}{|\Delta_{\text{out}}|}}, \sqrt{\frac{2^n}{|\Delta_{\text{in}}|}}\right\}, \frac{2^{n+1}}{|\Delta_{\text{out}}| \cdot |\Delta_{\text{in}}|}\right\}, \tag{1}$$

Table 1: Comparison of the Attacks against `sLiSCP` and `sLiSCP-light`

Target	Attack	Steps	Time	Memory	Reference
`sLiSCP-192`	ID/ZC	9	N/A	N/A	[ARH+17]
	LBD	15	$2^{122.7}$	$2^{37.7}$	[LSSW18]
	LBD	15	$2^{111.4}$	$2^{37.7}$	Sect. 3.2
`sLiSCP-256`	ID/ZC	9	N/A	N/A	[ARH+17]
	LBD	15	$2^{168.3}$	$2^{47.3}$	[LSSW18]
	LBD	15	$2^{149.6}$	$2^{47.3}$	Sect. 3.2
	LBD	16	$2^{154.6}$	$2^{47.3}$	Sect. 3.4
`sLiSCP-light-192`	integral	8	2^{191}	N/A	[AGH+19a]
	zero-sum	14	2^{192}	N/A	[AGH+19a]
	LBD	15	$2^{111.4}$	$2^{37.7}$	Sect. 3.3
	LBD	16	$2^{113.0}$	$2^{37.7}$	Sect. 3.4
`sLiSCP-light-256`	integral	8	2^{255}	N/A	[AGH+19a, AGH+19b]
	zero-sum	14	2^{256}	N/A	[AGH+19a, AGH+19b]
	LBD	15	$2^{149.6}$	$2^{47.3}$	Sect. 3.3
	LBD	16	$2^{154.6}$	$2^{47.3}$	Sect. 3.4

which was shown by Gilbert and Peyrin [GP10].

LBDs on permutations are claimed to be valid attacks when their complexity is less than (1). Although the lower bound for a random *function* was proven by Iwamoto et al [IPS13][3], no such a formal proof is known for a random *permutation*. In this paper, we for the first time give a formal proof that (1) is actually the (asymptotically) tight bound to solve the limited-birthday problem on a random permutation. More precisely, we prove the following theorem.

Theorem 1 (Lower bound for the limited-birthday problem, informal)**.** *When P is a random permutation, to solve the limited-birthday problem with a probability greater than p_s,*

$$\frac{1}{2p_s} \cdot \max\left\{ \min\left\{ \sqrt{\frac{2^n}{|\Delta_{\mathrm{out}}|}}, \sqrt{\frac{2^n}{|\Delta_{\mathrm{in}}|}} \right\}, \frac{2^n}{|\Delta_{\mathrm{out}}| \cdot |\Delta_{\mathrm{in}}|} \right\} \tag{2}$$

queries to P or P^{-1} are required[4]*.*

This theorem strengthens the rationale of validity of various LBDs including those in previous works such as [GP10], and our attacks on `sLiSCP` and `sLiSCP-light` (The complexities of all of our new attacks are smaller than the lower bound for a random permutation in (2)).

The proof of Theorem 1 is more complex than the proof for the lower bound on a random *function* by Iwamoto et al. since we have to deal with queries to both of P and P^{-1}. To achieve the lower bound that is the complex combination of "max" and "min", and is quite close to the upper bound (1), we introduce a technical parameter in our proof.

Paper Outline. `sLiSCP` permutation family and LBD will be introduced in section 2. New LBDs for `sLiSCP` permutation family will be shown in section 3. A proof of the lower bound of LBD will be shown in section 4. We will conclude this paper in section 5.

[3]The lower bound for a random function is $O\left(\max\left\{\sqrt{\frac{2^{n+1}}{|\Delta_{out}|}}, \frac{2^{n+1}}{|\Delta_{\mathrm{out}}|\cdot|\Delta_{\mathrm{in}}|}\right\}\right)$.

[4]The statement is valid as long as $2^{-n+4} \leq p_s \leq 1/2$.

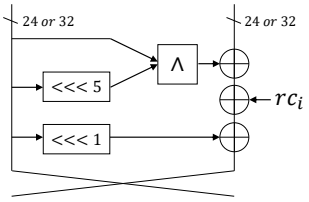

Figure 2: Round Function of Simeck.

2 Preliminaries

2.1 Specification of sLiSCP

An input to the permutation is first divided into four w-bit words, where $w = 48$ for sLiSCP-192 and $w = 64$ for sLiSCP-256. Let $(X_0^0, X_0^1, X_0^2, X_0^3)$ be the input to the permutation. This value is updated by iteratively computing the following step function shown in the left-hand side of Fig. 1 for $i = 0, 1, \ldots, 17$.

$$x_0^{i+1} = x_1^i, \quad x_1^{i+1} = x_2^i \oplus \mathrm{Simeck}_w^r(x_3^i) \oplus sc_i, \quad x_2^{i+1} = x_3^i, \quad x_3^{i+1} = x_0^i \oplus \mathrm{Simeck}_w^r(x_1^i) \oplus sc_i',$$

where Simeck_w^r is an r rounds of w-bit block Simeck, called Simeck box, described in the following paragraph and sc_i and sc_i' are step-dependent constants.

Specification of Simeck Box. Simeck [YZS+15] is a block cipher based on Simon [BSS+13]. Although Simeck supports various block sizes, only 48-bit block version called Simeck48 and 64-bit version called Simeck64 are adopted in sLiSCP and sLiSCP-light. Moreover, a key is replaced with a constant to convert Simeck into a keyless permutation.

Let $m \in \{24, 32\}$ be the word size such that $2m$ is a block size. The $2m$-bit input value is divided into two m-bit values $L_0 \| R_0$. Then the following is computed for $i = 1, 2, \ldots, r$ where r is 6 and 8 for Simeck48 and Simeck64, respectively.

$$L_i = R_{i-1} \oplus (L_{i-1} \wedge L_{i-1} \lll 5) \oplus L_{i-1} \lll 1 \oplus rc_i, \qquad R_i = L_{i-1},$$

where rc_i is the round constant generated by an LFSR that is initialized with sc_i or sc_i'. Since the constant does not impact to our analysis, we omit the details. The diagram of the round function of Simeck is illustrated in Fig. 2.

2.2 Specification of sLiSCP-light

sLiSCP-light is a tweaked design of sLiSCP. The major difference from sLiSCP is a step function and the number of steps to be computed. Let $(X_0^0, X_0^1, X_0^2, X_0^3)$ be the input to the permutation. This value is updated by iteratively computing the following step function shown in the right-hand side of Fig. 1 for $i = 0, 1, \ldots, 11$.

$$x_0^{i+1} = \mathrm{Simeck}_w^r(x_1^i), \qquad x_1^{i+1} = x_2^i \oplus \mathrm{Simeck}_w^r(x_3^i) \oplus sc_{2i+1},$$
$$x_2^{i+1} = \mathrm{Simeck}_w^r(x_3^i), \qquad x_3^{i+1} = x_0^i \oplus \mathrm{Simeck}_w^r(x_1^i) \oplus sc_{2i}.$$

sLiSCP-light is used as an underlying primitive of two NIST second-round candidates SpoC [AGH+19a] and Spix [AGH+19b]. Though the original number of steps is 12, the number of steps for the instantiations in those two designs is either 9 or 18.

2.3 Limited-Birthday Problem

The limited-birthday problem on a permutation P is the problem defined as follows[5].

Definition 1 (The limited-birthday problem on permutation)**.** Let P be an n-bit permutation, and Δ_{in}, Δ_{out} be (non-empty) closed subsets of $\{0,1\}^n$. For the limited-birthday problem on the permutation P, the goal of the adversary is to generate a quadruple of n-bit strings (X, X', Y, Y') such that $P(X) = Y$ and $P(X') = Y'$ and $X \oplus X' \in \Delta_{\text{in}}$ and $Y \oplus Y' \in \Delta_{\text{out}}$.

The complexity of the best known attack to solve the limited-birthday problem on a random permutation [GP10] is

$$\max\left\{\min\left\{\sqrt{\frac{2^n}{|\Delta_{\text{out}}|}}, \sqrt{\frac{2^n}{|\Delta_{\text{in}}|}}\right\}, \frac{2^{n+1}}{|\Delta_{\text{out}}| \cdot |\Delta_{\text{in}}|}\right\}. \tag{3}$$

3 Improved LBD against sLiSCP

We revisit the previous 15-step attack on sLiSCP in Sect. 3.1. We show how to improve its complexity for sLiSCP in Sect. 3.2. Applications to sLiSCP-light are then discussed in Sect. 3.3. We finally present the attacks on 16 steps in Sect. 3.4 and Sect. 3.5.

3.1 Previous Analysis on sLiSCP

Liu et al. [LSSW18] analyzed the differential properties of the sLiSCP permutation and presented a 15-step distinguisher for sLiSCP-192 and sLiSCP-256. They first focused on a 6-step iterative differential characteristic that maps a difference $(0,0,0,\alpha)$ to a difference $(0,0,0,\alpha)$. This includes four differential propagations of $\alpha \to \beta$ and two differential propagations of $\beta \to \alpha$ through 6-round (resp. 8-round) Simeck48 (resp. Simeck64). The 6-round iterative characteristic is shown in the left-hand side of Fig. 3.

Liu et al. then searched for the choice of α and β that has the maximum characteristic probability for $\alpha \to \beta$ and $\beta \to \alpha$ taking into account the weight that $\alpha \to \beta$ occurs twice more frequently than $\beta \to \alpha$. Such differential properties are summarized in Table 2.

Table 2: Differential Property of Simeck for 6-Round Iterative Characteristic [LSSW18].

Target	Differential Mask	Probability	
		Characteristic	Differential
Simeck^6_{48}	014000 || 020000 → 014000 || 008000	2^{-12}	$2^{-11.3}$
	014000 || 008000 → 014000 || 020000	2^{-26}	$2^{-21.8}$
Simeck^8_{64}	08800000 || 00000000 → 00800000 || 00000000	2^{-22}	$2^{-18.7}$
	00800000 || 00000000 → 08800000 || 00000000	2^{-22}	$2^{-18.7}$
Simeck^8_{64}	00000000 || 80000000 → 00000000 || 80000008	2^{-22}	$2^{-18.7}$
	00000000 || 80000008 → 00000000 || 80000000	2^{-22}	$2^{-18.7}$

Note that the probabilities in Table 2 were originally for the situation where each subkey of the keyed Simeck is chosen uniformly randomly, while what we need here is the probability that a randomly chosen plaintext (and the counterpart of the differential pair) would follow the specified differential transition for a fixed constant. We assume that the

[5] This definition follows the one for a (random) function by Iwamoto et al. [IPS13].

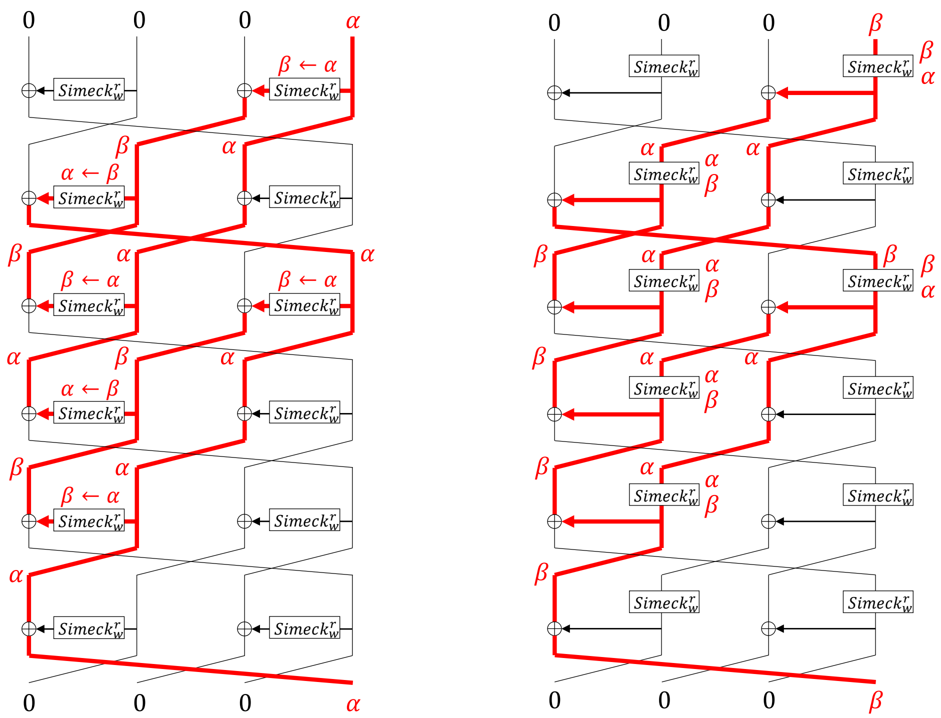

Figure 3: Left: 6-Step Iterative Differential Trail for Original `sLiSCP`. Right: 6-Step Iterative Differential Trail for `sLiSCP-light`.

probability of the differential characteristic of Simeck is almost the same for all keys, thus is also true for the fixed constant in `sLiSCP`.

Finally, Liu et al. built a 15-step differential characteristic and proposed to find a pair satisfying the characteristic using the rebound attack framework [MRST09, LMS+15], in which the attacker first efficiently finds paired values to satisfy the propagation for the middle steps (inbound phase), and then propagate the pairs backwards and forwards to probabilistically satisfy the characteristic (outbound phase).

Previous Inbound Phase for Three Steps. To explain our improvement, the previous procedure of the inbound phase needs to be explained more precisely. Liu et al. focused on four active Simeck boxes in three middle steps, which is shown in the left-hand side of Fig. 4. For each active Simeck box, paired values satisfying the differential propagation are exhaustively searched, which is performed with 4×2^{48} or 4×2^{64} computations. Any combination of the solutions from four Simeck boxes will fix the entire 192-bit or 256-bit state. In Fig. 4, paired values for red lines are fully determined by fixing paired values of four active Simeck boxes. The black lines can be directly computed from the red ones.

The combined number of solutions for four active Simeck boxes is $2^{125.8}$ and $2^{182.8}$ for `sLiSCP-192` and `sLiSCP-256`, respectively. Then the characteristic were extended so that the probability of the outbound phase is higher than this number of solutions. As shown in the left-hand side of Fig. 9 in the supplementary material A, 12 steps were added for the outbound phase that can be satisfied with probability $2^{-122.7}$ and $2^{-168.3}$.

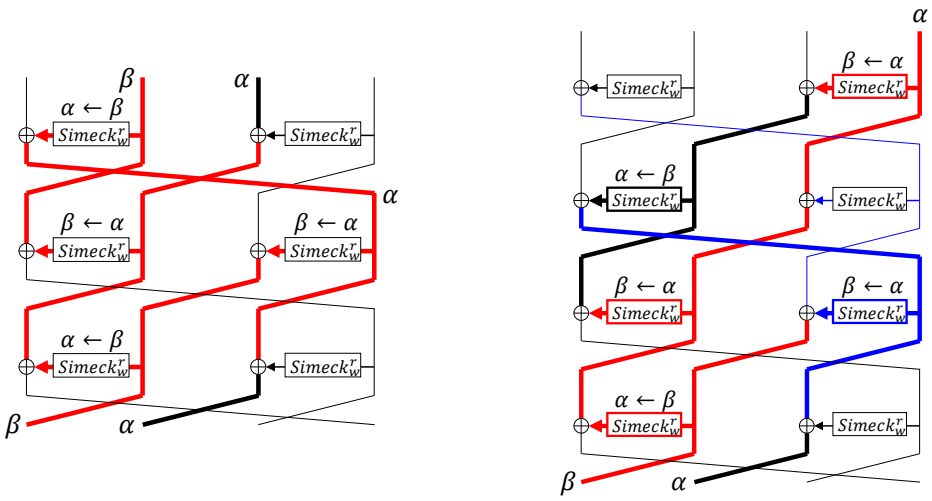

Figure 4: Left: Previous 3-Step Inbound Procedure for `sLiSCP`. Right: Improved 4-Step Inbound Procedure for `sLiSCP`. In the previous procedure, four active Simeck boxes are fixed independently, while in the new procedure, fixing three active Simeck boxes (red) will fix another one (blue), and an additional one (black) can be fixed independently.

3.2 Improving Complexity of 15-Step Attacks on `sLiSCP`

We present an improved procedure for the inbound phase, which covers four middle steps. Intuitively, this improvement on its own does not increase the number of paired values that satisfy the full characteristic, hence the number of attacked steps does not increase in a straight-forward analysis. Instead, the procedure to find the paired values will be more sophisticated (another stage is introduced for the divide-and-conquer approach), which improves the attack complexity as it allows to reduce the outbound part. The new inbound procedure is illustrated in the right-hand side of Fig. 4, which is explained as follows.

1. We exhaustively search for the paired values satisfying the differential propagation for three Simeck boxes; the right-hand side of the first step, the left-hand side of the third step, and the left-hand side of the fourth step (in red). As preparation for Step 3, we also search for the paired values for the left-hand side of the second step (in black).

2. We take any combination of the solutions from those three Simeck boxes, which fixes the paired values for another active Simeck box in the right-hand side of the third step (in blue). Then we check whether the differential propagation of this Simeck box is satisfied or not, which filters out wrong pairs at this stage.

3. Any combination of the solutions for the first four Simeck boxes (red and blue) and the precomputed Simeck box (black) fixes the entire state. Then, the state is propagated to the outbound part to satisfy the 15-step trail shown in the left-hand side of Fig. 9 in the supplementary material A.

Complexity Analysis for `sLiSCP-256`. Step 1 requires to test 2^{64} inputs for each Simeck box. Because the probability is $2^{-18.7}$ for both of $\alpha \to \beta$ and $\beta \to \alpha$, the number of obtained solutions will be $2^{64-18.7} = 2^{45.3}$ for each Simeck box. Time complexity of this step is $4 \times 2^{64} = 2^{66}$ and a memory to store $4 \times 2^{45.3} = 2^{47.3}$ values are required.

Step 2 requires to test $2^{45.3\times 3} = 2^{135.9}$ values. The power of the filter is $2^{-18.7}$, hence we will obtain $2^{117.2}$ solutions. The time complexity of this step is $2^{135.9}$ and a memory to store $2^{117.2}$ values would be required a priori.

We have $2^{45.3}$ for this Simeck box (black) and $2^{117.2}$ solutions from the previous step (red and blue). Hence, we can generate up to $2^{117.2+45.3} = 2^{162.5}$ solutions from four inbound steps.

In the outbound phase in Fig. 9, we need to control 1 active Simeck box for the first 2 steps and 6 active Simeck boxes from steps 7 - 14, which is satisfied with probability $2^{-18.7\times 8} = 2^{-149.6}$. The last step contains 1 active Simeck box, but we do not control it. In the end, after trying $2^{149.6}$ solutions of the inbound phase, we will have a pair whose input difference is of the form $(\beta, \alpha, 0, 0)$ and output difference is of the form $(0, *, \alpha, 0)$, where α and β are shown in Table 2 and $*$ denotes any difference.

The complexity to find such a pair for a random permutation is given by the limited birthday problem. The size of the input and output differences are 1 and 2^{64}, respectively, the generic attack complexity in Eq. (1) is $2^{256+1}/(1 \cdot 2^{64}) = 2^{193}$ and the lower bound in Eq. (2) is 2^{190}. Thus the distinguisher finds a non-ideal property of 15-step `sLiSCP-256`.

The first remark is that the previous attack complexity is $2^{168.3}$ and our new attack complexity is $2^{149.6}$. The improved attack factor is $2^{18.7}$. The improvement clearly comes from the inclusion of one more active Simeck box in the inbound phase.

The second remark is that the required memory of $2^{117.2}$ for Step 2 can be omitted by performing Step 3 (the exhaustive search of the active Simeck box should be finished in advance) as soon as a solution is generated in Step 2 from the tables of the red values. Hence the required memory is $2^{47.3}$.

Complexity Analysis for `sLiSCP-192`. The analysis is almost the same as `sLiSCP-256` but it is a bit more complicated because the probabilities for $\alpha \to \beta$ and $\beta \to \alpha$ are different; the former is $2^{-11.3}$ and the latter is $2^{-21.8}$ as shown in Table 2.

Step 1 requires to test 2^{48} inputs for each Simeck box. For two of them with $\alpha \to \beta$, we will obtain $2^{48-11.3} = 2^{36.7}$ solutions. For one of them with $\beta \to \alpha$, we will obtain $2^{48-21.8} = 2^{26.2}$ solutions. Time complexity of this step is $4 \times 2^{48} = 2^{50}$ and a memory to store $2 \times 2^{36.7}$ values are required.

Step 2 requires to test $2^{36.7+36.7+26.2} = 2^{99.6}$ values. The power of the filter is $2^{-11.3}$, hence we will obtain $2^{88.3}$ solutions. Time complexity of this step is $2^{99.6}$. Similarly to the case of `sLiSCP-256`, a memory to store $2^{88.3}$ values can be omitted by testing the filtered solutions on the fly thanks to the smaller previous lists.

We have $2^{48-21.8} = 2^{26.2}$ solutions for the precomputed Simeck box (black) and $2^{88.3}$ solutions from the previous step (red and blue). Hence, we can generate up to $2^{88.3+26.2} = 2^{114.5}$ solutions from the inbound phase.

The probability of the outbound phase in Fig. 9 is $2^{6\times -11.3+2\times -21.8} = 2^{-111.4}$. After trying $2^{111.4}$ pairs, we will have a pair whose input difference is of the form $(\beta, \alpha, 0, 0)$ and whose output difference is of the form $(0, *, \alpha, 0)$.

The generic complexity for a random permutation is $2^{192+1}/(1 \cdot 2^{48}) = 2^{145}$ and the lower bound is 2^{145}. Hence the distinguisher finds a non-ideal property of 15-step `sLiSCP-192`.

3.3 Application to 15-step LBD for `sLiSCP-light`

The distinguishers in the previous subsection apply to `sLiSCP`, while two NIST second round candidates Spix [AGH+19b] and `SpoC` [AGH+19a] are based on `sLiSCP-light`. According to the designers, the best distinguisher for `sLiSCP-light` is a zero-sum distinguisher for 14 steps. The designers of Spix and `SpoC` cited the work by Liu et al. [LSSW18] but did not mention the possibility of applying the rebound attack on `sLiSCP` to `sLiSCP-light`. Here we formally claim for the first time, that 15-step `sLiSCP-light` can be attacked by using a similar procedure to the one on `sLiSCP`.

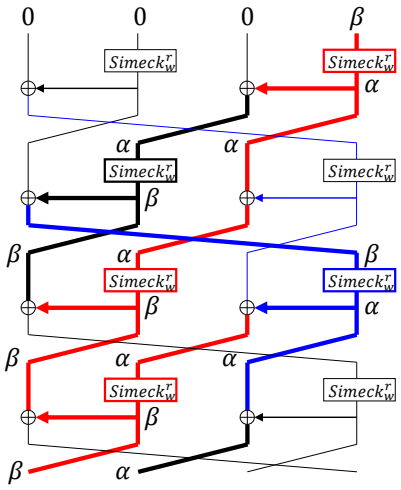

Figure 5: 4-Step Inbound Procedure for `sLiSCP-light`.

Our analysis starts from the 6-step iterative differential characteristic for the partial SPN in `sLiSCP-light`. The diagram of the differential propagation is illustrated in the right-hand of Fig. 3. The difference $(0, 0, 0, \beta)$ will be mapped to itself after 6 steps by going through four active Simeck boxes with the differential propagation $\alpha \to \beta$ and two active Simeck boxes with $\beta \to \alpha$. Hence, the efficiency of the 6-step trail as well as the best choice of α and β are the same as the ones for `sLiSCP`.

Our inbound phase that covers four steps of `sLiSCP` can also be applied to `sLiSCP-light`, which is illustrated in Fig. 5. We will omit the details that were already explained in the previous distinguishers. Intuitively, we first precompute all the solutions for three active Simeck boxes highlighted in red. Any combination uniquely determines the paired values for another active Simeck box highlighted in blue. Finally, we can freely choose the solution for another Simeck box highlighted in black.

The extension for the outbound phase is straightforward. The 15-step trail is given in Fig. 9 in supplementary material A. It includes six active Simeck box with differential propagation $\alpha \to \beta$, two active Simeck box with $\beta \to \alpha$, and one uncontrolled active Simeck box in the last step. The only difference is the form of the output difference, that is $(0, *, *, 0)$. Note that $*$ is unknown but must be identical for two branches, hence the size of the possible output differences remains the same as the size for `sLiSCP`.

As a result, 15-step `sLiSCP-light` can be attacked in a similar way as `sLiSCP`. The complexity is $2^{111.4}$ computational cost and $2^{38.7}$ memory amount for `sLiSCP-light-192`, and $2^{168.3}$ computational cost and $2^{47.3}$ memory amount for `sLiSCP-light-256`.

3.4 16 Steps Attacks against `sLiSCP-256`

The attack in the previous section cannot be extended to 16 steps easily from the following reason. The inbound phase can generate up to $2^{114.5}$ (resp. $2^{162.5}$) solutions for `sLiSCP-192` (resp. for `sLiSCP-256`), while the probability of the outbound phase of the 15-step attack is $2^{-111.4}$ (resp. $2^{-149.6}$). The remaining degrees of freedom is only $2^{3.1}$ (resp. $2^{12.9}$), which is not sufficient to satisfy one more active Simeck box.

Overall Idea. To extend the trail to 16 steps, we will have one more active Simeck box. We exploit the remaining degrees of freedom to control the differential propagation only partially. The input difference for the 16-step `sLiSCP` distinguisher is unchanged,

$(\beta, \alpha, 0, 0)$, while the output differences becomes $(\gamma, \alpha, 0, *)$, where γ is partially controlled, i.e. a few bits of γ are inactive. Depending on the number of inactive bits in γ, the dedicated attack can be faster than the generic attack. Fig. 6 illustrates how to extend by one step the 15-step distinguisher. For the sake of completeness, the entire 16-step trail is shown in Fig. 10 in supplementary material B.

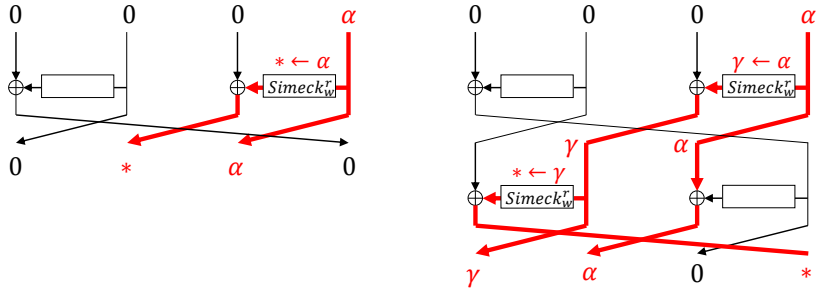

Figure 6: Left: the Last Step of the 15-Step Attack for `sLiSCP`. Right: the Last Two Steps of the 16-Step Attack for `sLiSCP`. Several inactive bit positions are specified for γ.

3.4.1 Analysis for `sLiSCP-256`

The differential probability for $\alpha \to \beta$ is $2^{-18.7}$, but we only have $2^{12.9}$ degrees of freedom left. To evaluate the impact of partially controlling the propagation, we look into the details of the differential characteristic with the highest probability.

For `sLiSCP-256`, as shown in Table 2, there are two kinds of the differential masks: $(\mathrm{wt}(\alpha), \mathrm{wt}(\beta)) = (2, 1)$ or $(1, 2)$, where wt denotes the Hamming weight. If the input differential mask has the heavier weight of 2, the first a few rounds have relatively low probability. Hence most of the remaining degrees of freedom are consumed for those rounds, and the number of uncontrolled rounds (following the full diffusion) becomes long, which is disadvantageous. Thus we set $\alpha \leftarrow$ `0080000` $||$ `0000000` and $\beta \leftarrow$ `0880000` $||$ `0000000`. This characteristic can be satisfied with probability 2^{-22}. The breakdown for each round is given in the left-hand side of Table 3.

Partially Controlled Differential Characteristic. The analysis of the propagation where we partially control the differential propagation of Simeck64 is shown in the right-hand side of Table 3. We have $2^{12.9}$ degrees of freedom remaining. The analysis here assumes that the differential propagation follows the characteristic up to 2^{-13} (this is a temporary assumption, we will later discuss its validity), and the analysis for the full diffusion is applied to the remaining rounds. Such words start to appear from the computation of round 5, which are denoted by L_5, L_6, L_7, and L_8 in the right-hand side of Table 3. The bitwise representation of those words are also given in Table 3, where '`0`' denotes the inactive bits, '`1`' denotes the active bits, and '`2`' denotes that the bit may or may not have a difference. As can be seen in the table, we have 33 inactive bits after 8 rounds; 14 bits in the left-hand side L_8 and 19 bits in the right-hand side L_7.

Differential Probability / Validity of the Assumption. The analysis in the previous paragraph is based on two assumptions. The first assumption is too optimistic for the attacker in which the propagation follows the characteristic up to probability 2^{-13}, while we only have $2^{12.9}$ degrees of freedom. The second assumption is too pessimistic for the attacker in which the analysis from round 5 follows the full diffusion. Note that the full diffusion analysis is the worst-case scenario for the attacker because it usually

Table 3: Differential Propagation for 8-Round Simeck64

Step	Differential Mask	probability	Step	Differential Mask		probability
	00800000 00000000			00800000 00000000		
1	01000000 00800000	2^{-2}	1	01000000 00800000		2^{-2}
2	02800000 01000000	2^{-2}	2	02800000 01000000		2^{-2}
3	04000000 02800000	2^{-4}	3	04000000 02800000		2^{-4}
4	0a800000 04000000	2^{-2}	4	0a800000 04000000		2^{-2}
5	01000000 0a800000	2^{-6}	5	L_5	0a800000	2^{-3}
6	08800000 01000000	2^{-2}	6	L_6	L_5	1
7	00000000 08800000	2^{-4}	7	L_7	L_6	1
8	08800000 00000000	1	8	L_8	L_7	1

L_5 : 0001 2021 0000 0000 0000 0000 0000 0000

L_6 : 0222 2222 1000 0000 0000 0000 0000 0000

L_7 : 2222 2222 2000 0000 0000 0000 0000 2222

L_8 : 2222 2222 2000 0000 0000 0002 2222 2222

corresponds to the situation where active AND gates always produce the difference. Indeed, this probability is the same as the situation where active AND gates always stop the difference. We also need to consider the differential probability, namely, even if the differential propagation does not follow the characteristic up to 2^{-13}, the target 33 bits can be inactive. Namely, the bit-wise differential form of $\gamma = \gamma_L \,||\, \gamma_R$ can be as follows.

γ_L = **** **** *000 0000 0000 000* **** ****

γ_R = **** **** *000 0000 0000 0000 0000 ****

The simplest way to evaluate the precise probability to satisfy γ is to perform an experiment, i.e. we choose many 64-bit values x and process x and $x \oplus \alpha$ with 8-round Simeck64 to calculate the probability that the target 33 bits are inactive. In our experiment, we took 1 million choices of x uniformly at random and 33 target bits became inactive for 32,501 choices. Hence, the probability is $2^{-4.94}$. This implies that we do not need to use 2^{13} degrees of freedom to make those 33 bits inactive, but only 2^5 degrees are sufficient.

Complexity Evaluation. The attack will spend 2^5 more degrees of freedom than the 15-step attack. Hence, the computational cost of the 16-step attack is $2^{149.6+5.0} = 2^{154.6}$. The required memory amount is $2^{47.3}$.

The generic attack complexity for a random permutation in Eq. (1) is $2^{256+1}/(1 \cdot 2^{64+(64-33)}) = 2^{162}$ because the input difference is fixed and the output difference $(\gamma, \alpha, 0, *)$ can be chosen from 2^{95} choices. The lower bound in Eq. (2) is 2^{159}. Hence our attack finds the non-ideal property of 16-step `sLiSCP-256`.

3.4.2 Remarks for `sLiSCP-192`

For `sLiSCP-192`, we only have $2^{3.1}$ degrees of freedom remaining and to control the differential characteristic for 6-round Simeck48 is difficult. Currently we have not found any valid distinguishers on 16-steps `sLiSCP-192`. We experimentally confirmed that the partial control of Simeck48 could work up to 5 rounds, but could not work for 6 rounds.

3.5 16 Steps Attacks against `sLiSCP-light-192` and `sLiSCP-light-256`

The extension to 16 steps can be similarly applied to `sLiSCP-light`, however there is a certain difference due to the usage of the different network. As shown in the right-hand side of Fig. 9, the last step of the 15-step attack includes a differential propagation with $\beta \to \alpha$. If the same strategy as `sLiSCP` is applied, we need to partially control this propagation. For Simeck48, the probability for $\beta \to \alpha$ is much smaller than one for $\alpha \to \beta$, hence it is not a good strategy to partially control the last step of the 15-step attack. To avoid this problem, we extend the 15-step trail in backwards, and partially control the difference in the first step of the 15-step attack. The diagram of the extended steps is given in Fig. 7.

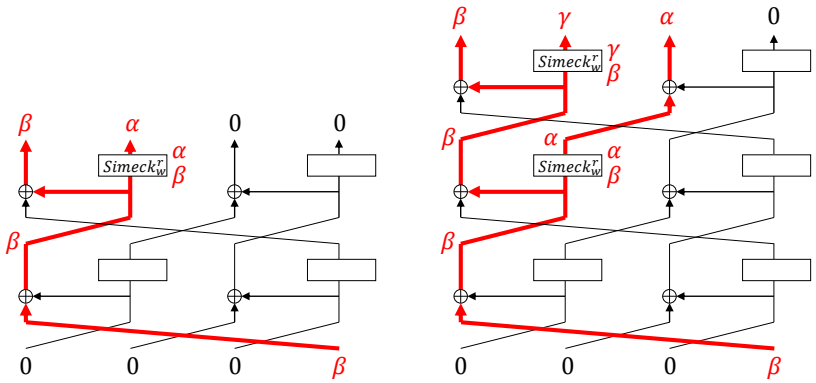

Figure 7: Left: the First Step of the 15-Step Attack for `sLiSCP-light`. Right: the First Two Steps of the 16-Step Attack for `sLiSCP-light`.

3.5.1 Peeling off the Last Round

We point out that Simeck round function has the property that the difference after the first round or before the last round can be computed from the input or the output without knowing the key value. This is because the key is directly added to the Feistel network, which is different from traditional designs in which the key is added inside a so-called F-function. To be more precise, let $I_l \,||\, I_r$ be the input to the Simeck permutation. Then an attacker can compute the difference after the first round by $i_r \oplus (i_l \lll 1) \oplus (i_l \wedge (i_l \lll 5)) \,||\, i_l$, which is independent of the secret-key/fixed-constant value.

Moreover, the network of `sLiSCP-light` helps an attacker to exploit this property. As shown in Fig. 7, the partially controlled Simeck box is located in the first step, and the input to the entire permutation (with difference γ) is directly used to compute this Simeck box. Hence, the attacker who only has an access to the input and the output of the entire permutation can actually compute the difference after 1 Simeck round. Note that it is not the case with `sLiSCP`. As shown in Fig. 6, the output of the partially controlled Simeck box is masked by an internal state due to the Feistel network. Hence, an attacker who only has an access to the output of the entire construction cannot compute 1 Simeck round.

3.5.2 Analysis for `sLiSCP-light-192`

We only have $2^{3.1}$ degrees of freedom remaining. The differential characteristic probability for 6-round Simeck48 is 2^{-12} and the propagation where the characteristic is satisfied up to 2^{-4} are given in Table 4.

As a result of our experiments, we fixed the inactive bit positions as bit positions 4 to 13 in both sides of the network, namely 20 inactive bits in total. In other words, we

Table 4: Differential Propagation for 6-Round or 5-Round Simeck48 Inverse

Step	Differential Mask	probability	Step	Differential Mask	probability
6	014000 020000	2^{-4}	6	****** ******	1
5	008000 014000	2^{-2}	5	R_4 R_5	1
4	004000 008000	2^{-2}	4	004000 R_4	1
3	000000 004000	1	3	000000 004000	1
2	004000 000000	2^{-2}	2	004000 000000	2^{-2}
1	008000 004000	2^{-2}	1	008000 004000	2^{-2}
	014000 008000			014000 008000	

R_5 : 0002 2001 2200 0000 0000 0002

R_4 : 0000 2000 1200 0000 0000 0000

expect to get zero-difference after masking the output value with 003ff0 in both sides. To evaluate the differential probability by the experiment, we took 1 million values and 20 target bits became inactive for 346,403 times. Hence, the probability is $2^{-1.53}$. This implies that we only need $2^{1.6}$ degrees of freedom compared to the 15-step attack. The computational cost of the 16-step attack is $2^{111.4+1.6} = 2^{113.0}$. The required memory amount is $2^{37.7}$.

The generic attack complexity for a random permutation in Eq. (1) is $2^{192+1}/(1 \cdot 2^{48+(48-20)}) = 2^{117}$ and the lower bound in Eq. (2) is 2^{114}. Hence our attack finds a non-ideal property for 16-step sLiSCP-light-192.

Towards More than Experimental Verification. The above attacks used the differential probability obtained by the experiments. Here, we discuss if there exist other methods to validate the obtained probability. Note that what we want to evaluate is different from the standard differential probability, i.e. summing up probabilities of all the trails that map a fixed input difference to a fixed output difference. In our setting, the input difference to the inverse of Simeck is fixed, but for the output difference, only several inactive bit positions are fixed. In case of sLiSCP-192, inactive 20-bit positions are fixed, in other words, the output differences can be any of $2^{48-20} = 2^{28}$ choices. Evaluating differential probability for such a case is not an easy task.

We approach this probability with MILP-based evaluation. For the input, we gave a condition for each bit to specify whether each bit is active or not. For the output, we gave a condition only for the target bits to set them inactive. The MILP solver found that the maximum characteristic probability to make the target 20 bits inactive is 2^{-8}. Hence, we added the condition that the probability of the trail is X, where $X = 2^{-8}, 2^{-9}, 2^{-10}, \cdots$ and counted the number of characteristics for each X. The results are given in Table 5.

Table 5 shows that Y distinct characteristics with probability X were found by MILP. As X becomes smaller, Y becomes bigger. We stopped this evaluation when the number of characteristics reached 10 million. The sum of the probability of all the characteristics up to 2^{-29} is $2^{-1.916} \approx 2^{-2.0}$. We still have some gap to reach $2^{-1.6}$, however we believe that this analysis provides better understanding of the differential probability obtained by our experiments.

3.5.3 Analysis for sLiSCP-light-256

16 steps of sLiSCP-light-256 can be attacked in the same procedure but the analysis can be much simpler because the probabilities for $\alpha \to \beta$ and $\beta \to \alpha$ are identical in

Table 5: Evaluation of Differential Probability with MILP. MILP found Y distinct characteristics with probability X, and its contribution is calculated by XY.

X	Y	$\log XY$	X	Y	$\log XY$	X	Y	$\log XY$	X	Y	$\log XY$
2^{-8}	4	-6.000	2^{-14}	224	-6.193	2^{-20}	13128	-6.320	2^{-26}	767468	-6.450
2^{-9}	0	0	2^{-15}	352	-6.541	2^{-21}	19096	-6.780	2^{-27}	1825772	-6.200
2^{-10}	32	-5.000	2^{-16}	860	-6.252	2^{-22}	36780	-6.833	2^{-28}	4013000	-6.064
2^{-11}	32	-6.000	2^{-17}	2144	-5.934	2^{-23}	64240	-7.029	2^{-29}	8144700	-6.043
2^{-12}	20	-7.678	2^{-18}	4796	-5.772	2^{-24}	138384	-6.922			
2^{-13}	4	-11.000	2^{-19}	6988	-6.229	2^{-25}	340008	-6.625			

Simeck64 (Table 2). Hence 16-step `sLiSCP-light-256` can be attacked in the same way as `sLiSCP-256`, which requires a computational cost of $2^{153.6}$ and memory amount of $2^{47.3}$.

4 Tight Lower Bound for the Limited-Birthday Problem

In previous works, LBDs on permutations are claimed to be valid distinguishers when their complexity is less than the one given by (3). For a random permutation, the complexity (3) has been considered to be the best because

1. we don't know of any algorithm that solves the limited-birthday problem on a random permutation with a complexity smaller than (3), and
2. on the limited-birthday problem on a random *function* F, the complexity $\max\left\{\sqrt{\frac{2^{n+1}}{|\Delta_{out}|}}, \frac{2^{n+1}}{|\Delta_{\mathrm{out}}|\cdot|\Delta_{\mathrm{in}}|}\right\}$ is proven to be tight [IPS13].

The above two evidences strongly indicate that (3) will be the tight bound to solve the limited-birthday problem on a random permutation. However, to strengthen the rationale of validity of LBDs on concrete permutations, it is highly desirable to give a formal proof that (3) is the tight bound for a random permutation. Although many works have been done on LBDs on permutations, there does not exist any previous work that gives such a formal proof.

We for the first time provide a formal proof showing that (3) is actually the (asymptotically) tight bound to solve the limited-birthday problem on a random permutation.

When the attack target is a random permutation, the limited-birthday problem can be reformalized as the game $G^{\mathcal{A}}$ such that an adversary $\mathcal{A}$ has access to the oracles P and P^{-1}, where P is a random permutation, and $\mathcal{A}$ wins if it outputs a quadruple of n-bit strings $(X_{\mathsf{fin}}, X'_{\mathsf{fin}}, Y_{\mathsf{fin}}, Y'_{\mathsf{fin}})$ such that $X_{\mathsf{fin}} \oplus X'_{\mathsf{fin}} \in \Delta_{\mathrm{in}}$ and $Y_{\mathsf{fin}} \oplus Y'_{\mathsf{fin}} \in \Delta_{\mathrm{out}}$ and $P(X_{\mathsf{fin}}) = Y_{\mathsf{fin}}$ and $P(X'_{\mathsf{fin}}) = Y'_{\mathsf{fin}}$. Let $G^{\mathcal{A}} \Rightarrow 1$ denote the event that $\mathcal{A}$ wins the game $G^{\mathcal{A}}$. The formal description of $G^{\mathcal{A}}$ is given in Fig. 8.

The following theorem shows that (3) is asymptotically the tight bound of the number of queries to solve the limited-birthday problem on a random permutation.

Theorem 2. *If $\mathcal{A}$ makes at most q queries,*

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq \frac{2^n}{2^n - q + 1} \cdot \frac{q}{\max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\}} + \frac{3}{2^n - q} \quad (4)$$

holds[6]*. In addition, to win the game $G^{\mathcal{A}}$ with a probability that is greater than p_s ($2^{-n+4} \leq$*

[6]Note that this implies $\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq O\left(\frac{q}{\max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\}}\right)$.

Game $G^{\mathcal{A}}$
$(X_{\mathsf{fin}}, X'_{\mathsf{fin}}, Y_{\mathsf{fin}}, Y'_{\mathsf{fin}}) \leftarrow \mathcal{A}^{P,P^{-1}}$
if $P(X_{\mathsf{fin}}) = Y_{\mathsf{fin}}$, $P(X'_{\mathsf{fin}}) = Y'_{\mathsf{fin}}$, $X_{\mathsf{fin}} \oplus X'_{\mathsf{fin}} \in \Delta_{\mathrm{in}}$, and $Y_{\mathsf{fin}} \oplus Y'_{\mathsf{fin}} \in \Delta_{\mathrm{out}}$
  return 1
else
  return 0

Procedure $P(X)$
if $(X, Y) \in L_P$ for some Y
  return Y
else
  $Y \xleftarrow{\$} \{0,1\}^n \setminus L_Y$
  $L_X \leftarrow L_X \cup \{X\}$, $L_Y \leftarrow L_Y \cup \{Y\}$, $L_P \leftarrow L_P \cup \{(X, Y)\}$
  return Y

Procedure $P^{-1}(Y)$
if $(X, Y) \in L_P$ for some X
  return X
else
  $X \xleftarrow{\$} \{0,1\}^n \setminus L_X$
  $L_X \leftarrow L_X \cup \{X\}$, $L_Y \leftarrow L_Y \cup \{Y\}$, $L_P \leftarrow L_P \cup \{(X, Y)\}$
  return X

Figure 8: The game $G^{\mathcal{A}}$ that defines the limited birthday problem on a random permutation P. The lists L_X, L_Y, and L_P are set to be empty at the beginning of the game.

$p_s \leq 1/2$), $\mathcal{A}$ has to make at least

$$\frac{p_s}{2} \cdot \max\left\{\min\left\{\sqrt{\frac{2^n}{|\Delta_{\mathrm{out}}|}}, \sqrt{\frac{2^n}{|\Delta_{\mathrm{in}}|}}\right\}, \frac{2^n}{|\Delta_{\mathrm{out}}| \cdot |\Delta_{\mathrm{in}}|}\right\} \tag{5}$$

queries to P and P^{-1}.

We first provide intuition of our proof for the theorem, and then describe the formal proof. Let I and O denote the integers such that $|\Delta_{\mathrm{in}}| = 2^I$ and $|\Delta_{\mathrm{out}}| = 2^O$.

Proof intuition. We provide proof intuition on the latter statement of the theorem (the number of queries should be larger than (5)) when $p_s = 1/2$.

For simplicity, we assume that $\mathcal{A}$ stores a pair (X, Y) into a list L at each query to P or P^{-1}. When $\mathcal{A}$ queries X to P, $\mathcal{A}$ stores $(X, P(X))$ into L. When $\mathcal{A}$ queries Y to P^{-1}, $\mathcal{A}$ stores $(P^{-1}(Y), Y)$ into L. Intuitively, at each query, $\mathcal{A}$ tries its best to obtain a new pair (X, Y) such that $X \oplus X' \in \Delta_{\mathrm{in}}$ and $Y \oplus Y' \in \Delta_{\mathrm{out}}$ for some existing pair (X', Y') in L.

Without loss of generality we can assume that $I \leq O$. Then, $\min\left\{\sqrt{\frac{2^n}{|\Delta_{\mathrm{out}}|}}, \sqrt{\frac{2^n}{|\Delta_{\mathrm{in}}|}}\right\} = \sqrt{\frac{2^n}{|\Delta_{\mathrm{out}}|}} = 2^{(n-O)/2}$ holds. Roughly speaking, the best strategy for $\mathcal{A}$ at the i-th query is to perform the following procedure:

1. Choose the largest possible subset $S \subset L$ such that $X' \oplus X'' \in \Delta_{\mathrm{in}}$ for all $(X', Y'), (X'', Y'') \in S$.

2. Choose a (fresh) X such that $X \oplus X' \in \Delta_{\text{in}}$ holds for all $(X', Y') \in S$.[7]
3. Query X to P, expecting that $Y = P(X)$ happens to satisfy the condition $Y \oplus Y'' \in \Delta_{\text{out}}$ for some $(X'', Y'') \in S$.

Intuitively, we can assume that $\mathcal{A}$ performs the above procedure at every query. Let p_i be the probability that "$Y = P(X)$ happens to satisfy the condition $Y \oplus Y'' \in \Delta_{\text{out}}$ for some $(X'', Y'') \in S$" in Step 3 of the above procedure is satisfied at the i-th query (i.e., p_i is the probability that $\mathcal{A}$ wins the game at the i-th query). For each pair (X'', Y'') in S, the probability that $Y = P(X)$ happens to satisfy the condition $Y \oplus Y'' \in \Delta_{\text{out}}$ is about $|\Delta_{\text{out}}|/2^n = 2^{n-O}$. Hence, roughly speaking,

$$\begin{aligned} p_i &\le (\text{the number of possible values of } Y'')/2^{n-O} \\ &= |S|/2^{n-O} \le \min\{|L|, |\Delta_{\text{in}}|\}/2^{n-O} = \min\left\{i, 2^I\right\}/2^{n-O} \end{aligned}$$

holds. In particular, $p_{\mathcal{A}\text{wins}} := \Pr\left[G^{\mathcal{A}} \Rightarrow 1\right]$ is upper bounded as

$$p_{\mathcal{A}\text{wins}} \le \sum_{1 \le i \le q} p_i \le \sum_{1 \le i \le q} p_q \le q \cdot \min\left\{q, 2^I\right\}/2^{n-O}.$$

In what follows, we show the contrapositive statement. That is, we show that, if q is smaller than (5) with $p_s = 1/2$, then $p_{\mathcal{A}\text{wins}} \le 1/2$ holds. We consider two separate cases depending on whether $2I + O \ge n$.

Suppose $2I+O \ge n$. In this case, $q < \frac{1}{4} \cdot \max\left\{\min\left\{\sqrt{\frac{2^n}{|\Delta_{\text{in}}|}}, \sqrt{\frac{2^n}{|\Delta_{\text{out}}|}}\right\}, \frac{2^n}{|\Delta_{\text{out}}| \cdot |\Delta_{\text{in}}|}\right\} = \frac{1}{4} \cdot \sqrt{\frac{2^n}{|\Delta_{\text{out}}|}} = \frac{1}{4} \cdot 2^{(n-O)/2}$ holds (recall that we are assuming $I \le O$). In addition, $2^{(n-O)/2} \le 2^I$ holds, which implies that $\min\left\{q, 2^I\right\} = q$ holds. Therefore $p_{\mathcal{A}\text{wins}}$ is upper bounded as $p_{\mathcal{A}\text{wins}} \le q \cdot \min\left\{q, 2^I\right\}/2^{n-O} = q^2/2^{n-O} \le \left(\frac{1}{4}2^{(n-O)/2}\right)^2/2^{n-O} = 1/16$. Hence $p_{\mathcal{A}\text{wins}} \le 1/2$ holds and the contrapositive statement holds when $2I + O \ge n$.

Next, suppose that $2I + O \le n - 1$ holds. In this case, $q < \frac{1}{4} \cdot \max\left\{\min\left\{\sqrt{\frac{2^n}{|\Delta_{\text{in}}|}}, \sqrt{\frac{2^n}{|\Delta_{\text{out}}|}}\right\}, \frac{2^n}{|\Delta_{\text{out}}| \cdot |\Delta_{\text{in}}|}\right\} = \frac{1}{4} \cdot \frac{2^n}{|\Delta_{\text{out}}| \cdot |\Delta_{\text{in}}|} = \frac{1}{4} \cdot 2^{n-I-O}$ holds (recall that we are assuming $I \le O$). If q is relatively small (i.e., $q \le 2^I$), $p_{\mathcal{A}\text{wins}}$ is upper bounded as $p_{\mathcal{A}\text{wins}} \le q \cdot \min\left\{q, 2^I\right\}/2^{n-O} = q^2/2^{n-O} \le 2^{2I}/2^{n-O} \le \frac{1}{2}$ since now we are assuming $2I + O \le n - 1$. In addition, if q is relatively large ($q > 2^I$), $p_{\mathcal{A}\text{wins}}$ is upper bounded as $p_{\mathcal{A}\text{wins}} \le q \cdot \min\left\{q, 2^I\right\}/2^{n-O} = q \cdot 2^I/2^{n-O} \le \left(\frac{1}{4} \cdot 2^{n-I-O}\right) \cdot 2^I/2^{n-O} = 1/4$. Therefore $p_{\mathcal{A}\text{wins}} \le 1/2$ holds and the contrapositive statement also holds when $2I + O \le n - 1$.

Formal proof. Based on the above intuition, we provide a formal proof of the theorem. Let $G^{\mathcal{A}} \Rightarrow 1$ denote the event that $\mathcal{A}$ wins the game $G^{\mathcal{A}}$. To show the theorem, we first show the following lemma.

Lemma 1. *Let $\mathcal{A}$ be an adversary that makes at most q queries to P and P^{-1} in total. If $I \le O$, then*

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \le \begin{cases} \frac{2^n}{2^n - q + 1} \cdot \frac{q^2}{2^{n-O}} + \frac{3}{2^n - q} & \text{if } q \le 2^I, \\ \frac{2^n}{2^n - q + 1} \cdot \frac{q}{2^{n-I-O}} + \frac{3}{2^n - q} & \text{if } q \ge 2^I, \end{cases}$$

holds. If $O \le I$, then

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \le \begin{cases} \frac{2^n}{2^n - q + 1} \cdot \frac{q^2}{2^{n-I}} + \frac{3}{2^n - q} & \text{if } q \le 2^O, \\ \frac{2^n}{2^n - q + 1} \cdot \frac{q}{2^{n-I-O}} + \frac{3}{2^n - q} & \text{if } q \ge 2^O, \end{cases}$$

holds.

[7] If it is impossible to find such an X, $\mathcal{A}$ has to choose another subset for S, or choose X just randomly.

Proof. We show the claim in the case $I \leq O$. The claim in the case $O \leq I$ can be shown in the same way.

First, without loss of generality we can assume the followings:

1. $\mathcal{A}$ is a deterministic algorithm.
2. $\Delta_{\text{in}} = \{\alpha \,||\, 0^{n-I} | \alpha \in \{0,1\}^I\}$ and $\Delta_{\text{out}} = \{\beta \,||\, 0^{n-O} | \beta \in \{0,1\}^O\}$.[8]
3. $\mathcal{A}$ does not query X to P if
   (a) $\mathcal{A}$ has already queried X to P before, or
   (b) $\mathcal{A}$ queried Y to P^{-1} for some Y before and got $P^{-1}(Y) = X$ as the response.
4. $\mathcal{A}$ does not query Y to P^{-1} if
   (a) $\mathcal{A}$ has already queried Y to P^{-1} before, or
   (b) $\mathcal{A}$ queried X to P for some X before and got $P(X) = Y$ as the response.

Let $(X_{\mathsf{fin}}, X'_{\mathsf{fin}}, Y_{\mathsf{fin}}, Y'_{\mathsf{fin}})$ denote the final output of $\mathcal{A}$, and (X_i, Y_i) denote the i-th element in L_P (*i.e.* (X_i, Y_i) is added to L_P when $\mathcal{A}$ makes the i-th query). Let X_i^L and X_i^R be the most significant I bits and the least significant $(n-I)$ bits of X_i, respectively. Let Y_i^L and Y_i^R be the most significant O bits and the least significant $(n-O)$ bits of Y_i, respectively. Let coll_i be the event that $X_j \oplus X_k \in \Delta_{\text{in}}$ and $Y_j \oplus Y_k \in \Delta_{\text{out}}$ for some $j < k \leq i$.

Then we have that

$$\begin{aligned}
\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] &\leq \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \wedge \neg\mathsf{coll}_q\right] + \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{coll}_q\right] \\
&= \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] \cdot \Pr\left[\neg\mathsf{coll}_q\right] + \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \mathsf{coll}_q\right] \cdot \Pr\left[\mathsf{coll}_q\right] \\
&\leq \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] + \Pr\left[\mathsf{coll}_q\right] \\
&\leq \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] + \left(\Pr\left[\mathsf{coll}_q \wedge \neg\mathsf{coll}_{q-1}\right] + \Pr\left[\mathsf{coll}_q \wedge \mathsf{coll}_{q-1}\right]\right) \\
&= \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] + \Pr\left[\mathsf{coll}_q | \neg\mathsf{coll}_{q-1}\right] \cdot \Pr\left[\neg\mathsf{coll}_{q-1}\right] + \Pr\left[\mathsf{coll}_{q-1}\right] \\
&\leq \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] + \Pr\left[\mathsf{coll}_q | \neg\mathsf{coll}_{q-1}\right] + \Pr\left[\mathsf{coll}_{q-1}\right] \\
&\leq \cdots \leq \Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] + \sum_{1 \leq i \leq q} \Pr\left[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}\right] \qquad (6)
\end{aligned}$$

holds, where we denote $\Pr[\mathsf{coll}_1]$ by "$\Pr[\mathsf{coll}_1 | \neg\mathsf{coll}_0]$", by abuse of notation.

Upper bounding the term $\sum_{1 \leq i \leq q} \Pr[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}]$ in (6).

Suppose that $\mathcal{A}$'s i-th query is made to P (but not to P^{-1}). If $X_i^R \neq X_j^R$ for all $j < i$, $\Pr[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}] = 0$. If there exist indices $1 \leq j_1 < \cdots < j_s < i$ such that

[8] We can assume this due to the following reason. Suppose that an adversary $\mathcal{B}$ solves the problem by making q queries if elements of Δ_{in}, Δ_{out} have the form $\alpha||0^{n-I}$ and $\beta||0^{n-O}$, respectively. Assume that $\mathcal{B}$ does not query to P^{-1} for simplicity. Then, we can make an adversary $\mathcal{A}$ to solve the problem for arbitrary closed subsets $\Delta'_{\text{in}}, \Delta'_{\text{out}} \subset \{0,1\}^n$ such that $|\Delta'_{\text{in}}| = 2^I$ and $|\Delta'_{\text{out}}| = 2^O$ by making q queries, as follows: (1) $\mathcal{A}$ chooses linear isomorphisms $L, \tilde{L} : \{0,1\}^n \to \{0,1\}^n$ such that $L(\Delta_{\text{in}}) = \Delta'_{\text{in}}$ and $\tilde{L}(\Delta_{\text{out}}) = \Delta'_{\text{out}}$. (2) $\mathcal{A}$ runs $\mathcal{B}$. When $\mathcal{B}$ queries X, $\mathcal{A}$ queries $L(X)$ to P, and returns $(\tilde{L}^{-1} \circ P \circ L)(X)$ to $\mathcal{B}$. (3) When $\mathcal{B}$ returns an output (X, X', Y, Y'), $\mathcal{A}$ returns $(L(X), L(X'), \tilde{L}(Y), \tilde{L}(Y'))$ as its own output. Then $\mathcal{A}$ simulates a random permutation perfectly, and $\Pr[G^{\mathcal{A}} \Rightarrow 1] = \Pr[G^{\mathcal{B}} \Rightarrow 1]$ holds. (Here we considered the special case that $\mathcal{B}$ does not make queries to P^{-1}, but the general case can be shown in the same way.)

$X_i^R = X_{j_1}^R = X_{j_2}^R = \cdots = X_{j_s}^R$ and $X_i^R \neq X_{j'}^R$ holds for all $j' \in \{1, \ldots, i-1\} \setminus \{j_1, \ldots, j_s\}$,

$$\begin{aligned}
&\Pr\left[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}\right] \\
&= \Pr\left[Y_i^R = Y_{j_k}^R \text{ for some } 1 \le k \le s \middle| \neg\mathsf{coll}_{i-1}\right] \\
&= \sum_{1\le k\le s} \Pr\left[Y_i^R = Y_{j_k}^R \middle| \neg\mathsf{coll}_{i-1}\right] \\
&\le \sum_{1\le k\le s} \frac{\left(\text{The number of possible values for } (Y_i)^L \text{ under the condition } (Y_i)^R = Y_{j_k}^R\right)}{(\text{The number of possible values for } Y_i)} \\
&\le \sum_{1\le k\le s} \frac{2^O - 1}{2^n - (i-1)} \le \frac{\min\{2^I - 1, i-1\}(2^O - 1)}{2^n - (i-1)}
\end{aligned}$$

holds (we used the property $s \le \min\{2^I - 1, i-1\}$ for the last inequality). Similarly, if $\mathcal{A}$'s i-th query is made to P^{-1}, we can show that

$$\Pr\left[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}\right] \le \frac{\min\{2^O - 1, i-1\}(2^I - 1)}{2^n - (i-1)}$$

holds. Therefore we have

$$\begin{aligned}
\Pr\left[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}\right] &\le \frac{\max\left\{\min\{2^I - 1, i-1\}(2^O - 1), \min\{2^O - 1, i-1\}(2^I - 1)\right\}}{2^n - (i-1)} \\
&= \begin{cases} \frac{(i-1)(2^O-1)}{2^n-(i-1)} & \text{if } i \le 2^I, \\ \frac{(2^I-1)(2^O-1)}{2^n-(i-1)} & \text{if } i \ge 2^I. \end{cases}
\end{aligned}$$

(Recall that now we are assuming that $2^I \le 2^O$ holds.) Hence

$$\sum_{1\le i\le q} \Pr\left[\mathsf{coll}_i | \neg\mathsf{coll}_{i-1}\right] \le \begin{cases} \frac{2^n}{2^n-q+1} \cdot \frac{q^2}{2^{n-O}} & \text{if } q \le 2^I, \\ \frac{2^n}{2^n-q+1} \cdot \frac{q}{2^{n-I-O}} & \text{if } q \ge 2^I, \end{cases} \tag{7}$$

follows.

Upper bounding the term $\Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right]$ *in* (6).

Let bad_1, bad_2, and bad_3 be the events that $G^{\mathcal{A}} \Rightarrow 1$ holds ($\mathcal{A}$ wins the game $G^{\mathcal{A}}$) and

1. $(X_{\mathsf{fin}}, Y_{\mathsf{fin}}) \notin L_P$, $(X'_{\mathsf{fin}}, Y'_{\mathsf{fin}}) \in L_P$,
2. $(X_{\mathsf{fin}}, Y_{\mathsf{fin}}) \in L_P$, $(X'_{\mathsf{fin}}, Y'_{\mathsf{fin}}) \notin L_P$, and
3. $(X_{\mathsf{fin}}, Y_{\mathsf{fin}}) \notin L_P$, $(X'_{\mathsf{fin}}, Y'_{\mathsf{fin}}) \notin L_P$

hold just after $\mathcal{A}$ outputs the final output $(X_{\mathsf{fin}}, X'_{\mathsf{fin}}, Y_{\mathsf{fin}}, Y'_{\mathsf{fin}})$, respectively. Then

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg\mathsf{coll}_q\right] = \sum_{1\le i\le 3} \Pr\left[\mathsf{bad}_i | \neg\mathsf{coll}_q\right] \tag{8}$$

holds.

Now we have that

$$\Pr\left[\mathsf{bad}_1 | \neg\mathsf{coll}_q\right] \le \Pr\left[P(X_{\mathsf{fin}}) = Y_{\mathsf{fin}} | (X_{\mathsf{fin}}, Y_{\mathsf{fin}}) \notin L_P\right] \le \frac{1}{2^n - q} \tag{9}$$

holds. Similarly,

$$\Pr\left[\mathsf{bad}_2 | \neg\mathsf{coll}_q\right] \le \frac{1}{2^n - q}, \quad \text{and} \quad \Pr\left[\mathsf{bad}_3 | \neg\mathsf{coll}_q\right] \le \frac{1}{2^n - q} \tag{10}$$

hold.

From (8), (9), and (10),

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1 \middle| \neg \mathsf{coll}_q\right] \leq \frac{3}{2^n - q} \tag{11}$$

follows.

The claim of the proposition (in the case $I \leq O$) follows from (6), (7), and (11). □

Next, we show the following proposition. To achieve the lower bound that is the complex combination of "max" and "min" and quite close to the known best upper bound, we introduce a technical parameter c.

Proposition 1. *Let c be a positive number such that $c \leq n$. If $\mathcal{A}$ makes at most*

$$q \leq \max\left\{\min\left\{2^{(n-O)/2-c/2}, 2^{(n-I)/2-c/2}\right\}, 2^{n-(I+O)-c}\right\},$$

queries,

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^c} + \frac{3}{2^n - q}$$

holds.

Proof. We show the claim of the proposition when $I \leq O$. The claim for $I \geq O$ can be shown in the same way. Let $Q(n, c, I, O) := \max\left\{\min\left\{2^{(n-O)/2-c/2}, 2^{(n-I)/2-c/2}\right\}, 2^{n-(I+O)-c}\right\}$.

We consider two cases depending on whether $2I + O \geq n - c$ or $2I + O \leq n - c$. Note that $I \leq O$ is equivalent to

$$\min\left\{2^{(n-O)/2-c/2}, 2^{(n-I)/2-c/2}\right\} = 2^{(n-O)/2-c/2},$$

and $2I + O \geq n - c$ is equivalent to

$$\max\left\{2^{(n-O)/2-c/2}, 2^{n-(I+O)-c}\right\} = 2^{(n-O)/2-c/2}.$$

Case I: $I \leq O$ and $2I + O \geq n - c$.
In this case, $q \leq Q(n, c, I, O) = 2^{(n-O)/2-c/2} \leq 2^I$ holds. Hence

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq \frac{2^n}{2^n - q + 1} \cdot \frac{q^2}{2^{n-O}} + \frac{3}{2^n - q} \leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^c} + \frac{3}{2^n - q}$$

follows from Lemma 1.

Case II: $I \leq O$ and $2I + O \leq n - c$.
In this case, $q \leq Q(n, c, I, O) = 2^{n-(I+O)-c}$ holds. If $q \leq 2^I$ holds,

$$\begin{aligned}\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] &\leq \frac{2^n}{2^n - q + 1} \cdot \frac{q^2}{2^{n-O}} + \frac{3}{2^n - q} \leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^{n-(2I+O)}} + \frac{3}{2^n - q} \\ &\leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^c} + \frac{3}{2^n - q}\end{aligned}$$

follows from Lemma 1. In addition, if $2^I \leq q$ holds,

$$\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq \frac{2^n}{2^n - q + 1} \cdot \frac{q}{2^{n-(I+O)}} + \frac{3}{2^n - q} \leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^c} + \frac{3}{2^n - q} \tag{12}$$

follows from Lemma 1. Therefore the claim of the proposition also holds when $I \leq O$ and $2I + O \leq n - c$. □

Proof of Theorem 2. First, we show (4). Let c be the positive value such that

$$q = \frac{1}{2^c} \cdot \max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\}.$$

Then, $q \leq \max\left\{\min\left\{2^{(n-O)/2-c/2}, 2^{(n-I)/2-c/2}\right\}, 2^{n-(I+O)-c}\right\}$ holds. Hence, from Proposition 1,

$$\begin{aligned}\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] &\leq \frac{2^n}{2^n - q + 1} \cdot \frac{1}{2^c} + \frac{3}{2^n - q} \\ &= \frac{2^n}{2^n - q + 1} \cdot \frac{q}{\max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\}} + \frac{3}{2^n - q}\end{aligned}$$

follows. Therefore (4) holds.

Second, we show the latter statement of the theorem. Suppose

$$q < (p_s/2) \cdot \max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\},$$

where $2^{-n+4} \leq p_s \leq 1/2$. Let $m := -\log_2(p_s)$. Then, $q < (p_s/2) \cdot 2^n = 2^{n-m-1}$ and $1 \leq m \leq n-4$ hold. Hence, from (4), it follows that

$$\begin{aligned}\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] &< \frac{2^n}{2^n - q + 1} \cdot \frac{p_s}{2} + \frac{3}{2^n - q} \leq \frac{2^n}{2^n - 2^{n-m-1} + 1} \cdot \frac{1}{2^{m+1}} + \frac{3}{2^n - 2^{n-m-1}} \\ &= \frac{1}{2^m} \cdot \left(\frac{1}{1 - 2^{-m-1} + 2^{-n}} \cdot \frac{1}{2} + \frac{3}{2^{n-m}(1 - 2^{-m-1})}\right) \\ &\leq \frac{1}{2^m} \cdot \left(\frac{1}{1 - 2^{-2}} \cdot \frac{1}{2} + \frac{3}{2^4(1 - 2^{-2})}\right) = \frac{1}{2^m}\left(\frac{2}{3} + \frac{1}{4}\right) < \frac{1}{2^m} = p_s \qquad (13)\end{aligned}$$

holds. Therefore, if $q < (p_s/2) \cdot \max\left\{\min\left\{2^{(n-O)/2}, 2^{(n-I)/2}\right\}, 2^{n-(I+O)}\right\}$ (and $2^{-n+4} \leq p_s \leq 1/2$), then $\Pr\left[G^{\mathcal{A}} \Rightarrow 1\right] \leq p_s$ holds. This is the contraposition of the latter statement of the theorem. □

5 Conclusion

We can identify three main results in this paper:

1. Improved cryptanalysis results: First, we have showed improved attacks against `sLiSCP-192`, `sLiSCP-256`, `sLiSCP-light-192` and `sLiSCP-light-256`. We could increase the highest number of attack steps in the three latter ones. Though no attack on the full step primitives is presented, our results allow to better determine and understand the security margin of the related primitives, in particular of Spix and `SpoC`, two 2-round candidates of the NIST lightweight standardization process.

2. From a generalized point of view: To achieve the previous results, we improved the part solving the inbound phase from the previous applied rebound attacks, showing once again (as was done for instance in [Nay11]) that this phase is very technical. Our new results provide some hints on how to improve the inbound phase in previous attacks: the order of building the lists to merge needs to be carefully chosen so that the highest possible number of steps can be included in the inbound phase. For that, it seems a good idea to merge first the parts that have a lower number of solutions. Also, on-the-fly computations often allow to reduce the memory needs. The outbound can sometimes also be extended by relaxing the input-output conditions studying the technical properties of the round functions. An interesting

further work would be to study if an automatic tool could provide the best LBD, like see for instance [STW+14]. For example, we evaluated the differential probability of the partially controlled trails by the experiments, but we also tried to evaluate it with MILP. An efficient method to solve this problem would be an interesting direction.

3. Finally, we were able to prove for the first time that the bound of limited birthday distinguishers on random permutations is asymptotically tight, which is a very important result in order to asses the actual impact of the permutation distinguishers published in the literature.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 714294 - acronym QUASYModo).

References

[AGH+19a] Riham AlTawy, Guang Gong, Morgan He, Ashwin Jha, Kalikinkar Mandal, Mridul Nandi, and Raghvendra Rohit. SpoC: An Authenticated Cipher Submission to the NIST LWC Competition. Submitted to NIST Lightweight Standardization Process, 2019.

[AGH+19b] Riham AlTawy, Guang Gong, Morgan He, Kalikinkar Mandal, and Raghvendra Rohit. Spix: An Authenticated Cipher Submission to the NIST LWC Competition. Submitted to NIST Lightweight Standardization Process, 2019.

[ARH+17] Riham AlTawy, Raghvendra Rohit, Morgan He, Kalikinkar Mandal, Gangqiang Yang, and Guang Gong. sLiSCP: Simeck-Based Permutations for Lightweight Sponge Cryptographic Primitives. In Carlisle Adams and Jan Camenisch, editors, *Selected Areas in Cryptography - SAC 2017 - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, volume 10719 of *Lecture Notes in Computer Science*, pages 129–150. Springer, 2017.

[ARH+18] Riham AlTawy, Raghvendra Rohit, Morgan He, Kalikinkar Mandal, Gangqiang Yang, and Guang Gong. SLISCP-light: Towards Hardware Optimized Sponge-specific Cryptographic Permutations. *ACM Trans. Embedded Comput. Syst.*, 17(4):81:1–81:26, 2018.

[BDPA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. 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 320–337. Springer, 2011.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013.

[Dae17] Joan Daemen. Innovations in permutation-based encryption and/or authentication. 2017. Invited talk.

[GP10] Henri Gilbert and Thomas Peyrin. Super-sbox cryptanalysis: Improved attacks for aes-like permutations. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*, pages 365–383. Springer, 2010.

[IPS13] Mitsugu Iwamoto, Thomas Peyrin, and Yu Sasaki. Limited-birthday distinguishers for hash functions - collisions beyond the birthday bound can be meaningful. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 504–523. Springer, 2013.

[KPR20] Liliya Kraleva, Raluca Posteuca, and Vincent Rijmen. Cryptanalysis of the permutation based algorithm SpoC. Cryptology ePrint Archive, Report 2020/1072, 2020. Presented at the Fourth Lightweight Cryptography Workshop organized by NIST.

[LMS+15] Mario Lamberger, Florian Mendel, Martin Schläffer, Christian Rechberger, and Vincent Rijmen. The rebound attack and subspace distinguishers: Application to whirlpool. *J. Cryptology*, 28(2):257–296, 2015.

[LSSW18] Yunwen Liu, Yu Sasaki, Ling Song, and Gaoli Wang. Cryptanalysis of reduced sliscp permutation in sponge-hash and duplex-ae modes. In Carlos Cid and Michael J. Jacobson Jr., editors, *Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers*, volume 11349 of *Lecture Notes in Computer Science*, pages 92–114. Springer, 2018.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced whirlpool and grøstl. In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.

[Nat19] National Institute of Standards and Technology. Lightweight Cryptography (LWC) Standardization project, 2019. https://csrc.nist.gov/projects/lightweight-cryptography.

[Nay11] María Naya-Plasencia. How to improve rebound attacks. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 188–205. Springer, 2011.

[STW+14] Yu Sasaki, Yuuki Tokushige, Lei Wang, Mitsugu Iwamoto, and Kazuo Ohta. An automated evaluation tool for improved rebound attack: New distinguishers and proposals of shiftbytes parameters for grøstl. In Josh Benaloh, editor, *Topics in Cryptology - CT-RSA 2014 - The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings*, volume 8366 of *Lecture Notes in Computer Science*, pages 424–443. Springer, 2014.

[Tea20] SpoC Team. LWC workshop talk on cryptanalysis of SpoC. A message was posted to NIST LWC forum, 2020. 23rd October.

[TMÇ+19] Meltem Sönmez Turan, Kerry McKay, Çagdas Çalik, Donghoon Chang, and Lawrence Bassham. Status Report on the First Round of the NIST Lightweight Cryptography Standardization Process. https://csrc.nist.gov/publications/detail/nistir/8268/final, 2019.

[YZS+15] Gangqiang Yang, Bo Zhu, Valentin Suder, Mark D. Aagaard, and Guang Gong. The simeck family of lightweight block ciphers. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *Lecture Notes in Computer Science*, pages 307–329. Springer, 2015.

A 15-Step Trail for sLiSCP and sLiSCP-light

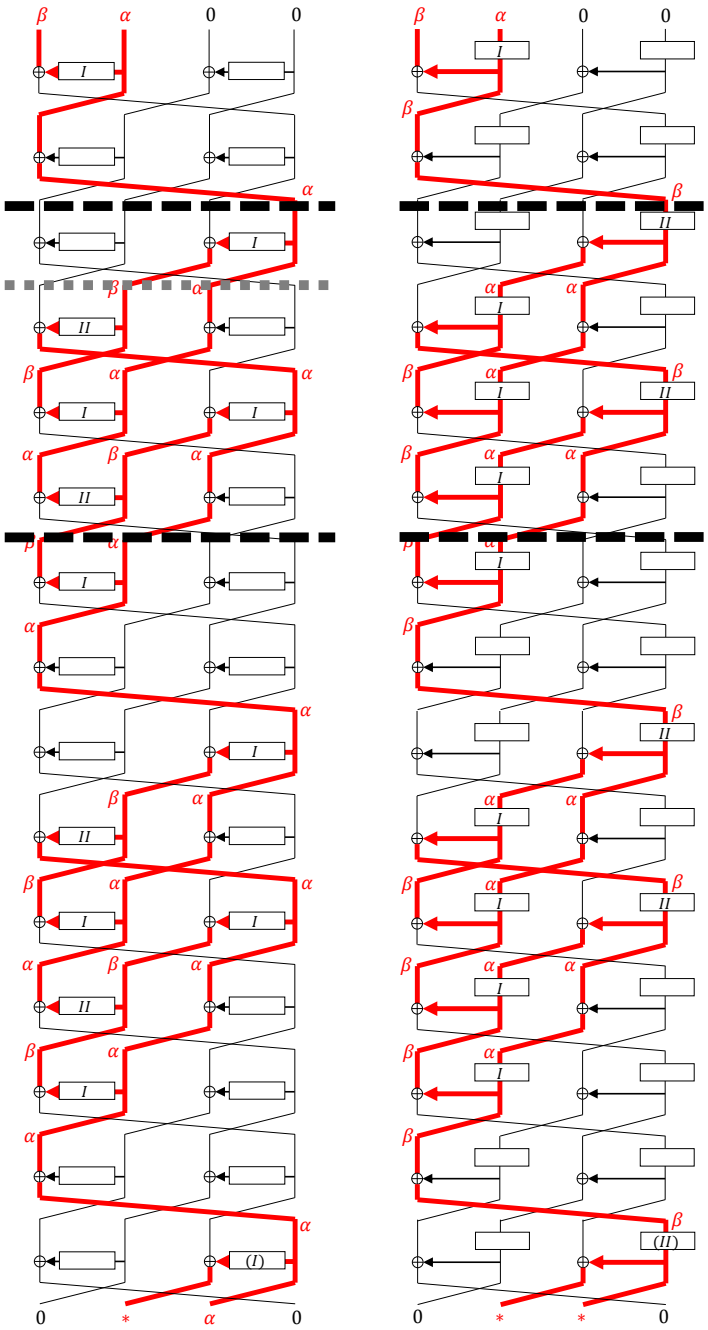

Figure 9: Left: 15-Step Distinguisher for sLiSCP. Right: 15-Step Distinguisher for sLiSCP-light. I and II denote the differential propagation $\alpha \to \beta$ and $\beta \to \alpha$, respectively. $(\cdot)$ denote that the differential propagation is not controlled, hence the output difference is unpredictable. The gray dotted line and black broken line are the border between the inbound and outbound phases in the previous work and our work, respectively.

B 16-Step Trails Extended from 15-Step Trails

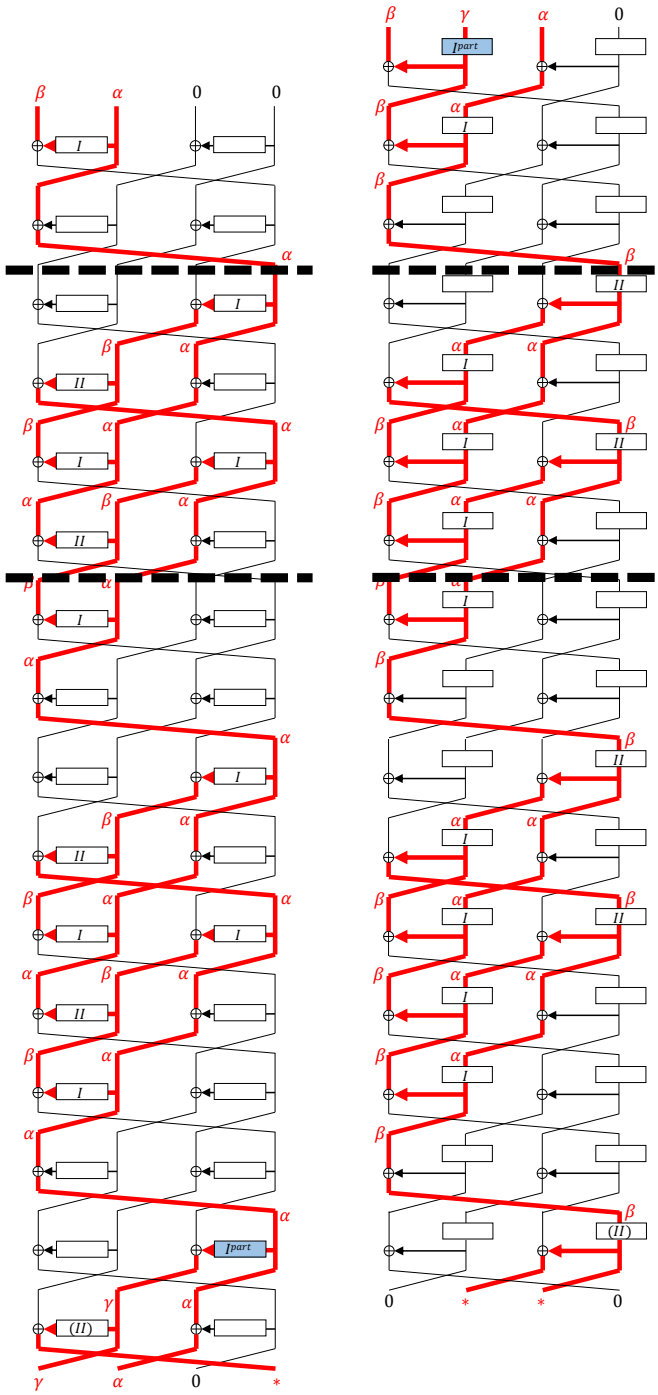

Figure 10: Left: 16-Step Distinguisher for `sLiSCP` with the Last Step Extended. Right: Extended 16-Step Distinguisher for `sLiSCP-light` with the First Step Extended. Both contain 1 partially controlled active Simeck box denoted by 'I^{part}'.