Clustering Effect in Simon and Simeck

Gaëtan Leurent[1], Clara Pernot[1], and André Schrottenloher[2]

[1] Inria, France firstname.lastname@inria.fr
[2] Cryptology Group, CWI, Amsterdam, The Netherlands
firstname.lastname@m4x.org

Abstract. Simon and Simeck are two lightweight block ciphers with a simple round function using only word rotations and a bit-wise AND operation. Previous work has shown a strong clustering effect for differential and linear cryptanalysis, due to the existence of many trails with the same inputs and outputs.
In this paper, we explore this clustering effect by exhibiting a class of high probability differential and linear trails where the active bits stay in a fixed window of w bits. Instead of enumerating a set of good trails contributing to a differential or a linear approximation, we compute the probability distribution over this space, including all trails in the class. This results in stronger distinguishers than previously proposed, and we describe key recovery attacks against Simon and Simeck improving the previous results by up to 7 rounds. In particular, we obtain an attack against 42-round Simeck64, leaving only two rounds of security margin, and an attack against 45-round Simon96/144, reducing the security margin from 16 rounds to 9 rounds.

Keywords: Lightweight cipher · Simon· Simeck· differential cryptanalysis · linear cryptanalysis · clustering effect

1 Introduction

Simon and Simeck are two lightweight block ciphers with a simple round function and very good hardware and software performances. Simon [5] was designed by Beaulieu, Shors, Smith, Treatman-Clark, Weeks and Wingers and published without a rationale, but has been considered for ISO standardisation. It follows a Feistel structure with a very simple round function:

$$f(x) = ((x \lll 8) \wedge (x \lll 1)) \oplus (x \lll 2).$$

Simeck is an academic variant of Simon designed by Yang, Zhu, Suder, Aagaard and Gong, and published at CHES 2015 [28]. It has the same number of rounds, and the same round function as Simon, but with different rotation amounts:

$$f(x) = ((x \lll 5) \wedge x) \oplus (x \lll 1).$$

The key schedule of Simeck is also modified to reuse the function f.

Previous work has shown that the best attacks against these ciphers use differential cryptanalysis or linear cryptanalysis [1,9,12,18,24], and has provided a detailed analysis of differential paths and linear trails using various techniques and tools [6,17,21,27]. Moreover they show a strong clustering effect for differential characteristics and linear trails. There exist many trails with the same input and output, and the probability of a differential (respectively the potential of a linear approximation) is significantly higher than the probability of the best characteristic (respectively the best linear trail). In order to estimate the probability of a differential or the potential of a linear approximation, we have to combine the effect of as many trails as possible with the corresponding input/output. This generates a lower bound on the quality of the differential or linear approximation. For instance, the best differential characteristic for 27-round Simeck64 has probability 2^{-70} [18], but a 27-round differential $(\mathtt{0}, \mathtt{11}) \to (\mathtt{5}, \mathtt{2})$ with probability $2^{-60.75}$ was given in [16].

Our Contribution. In this work, we explore this clustering effect in a more systematic way. Instead of building a list of trails with a given input/output, we consider a class of high probability trails where the active bits stay in a fixed window of w bits. In particular, we observe that the differentials and linear hulls used in most previous attacks fit in this framework.

Using properties of the round function, we compute efficiently the probability distribution over this space by multiplication of the differential transition matrix, or the linear correlation matrix. This provides a tighter lower bound on the probability of the differential (or the potential of the linear approximation) than used in previous works, because we implicitly consider *all* trails with intermediate states fitting in the window. Concretely, the 27-round differential $(\mathtt{0}, \mathtt{11}) \to (\mathtt{5}, \mathtt{2})$ has probability at least $2^{-56.06}$ for Simeck64. In general, we obtain a good understanding of the propagation of differences and linear masks in this class: there is a high probability to stay in the class because of the slow diffusion of Simon and Simeck.

We observe that this class includes many high quality distinguishers with input/output that are independent of the number of rounds targeted by the attacks. In particular, we use distinguishers with a single active bit in the input and output, because we can add more rounds of key-recovery than when using distinguishers with multiple active bits. Concretely, for Simeck64, the differential $(\mathtt{0}, \mathtt{1}) \to (\mathtt{1}, \mathtt{0})$ has probability at least $2^{-54.72}$ over 27 rounds, and $2^{-60.41}$ over 30 rounds.

Finally, we use the distinguishers to build key-recovery attacks, using dynamic key-guessing [24,26] for differential attacks, and the Fast Walsh Transform approach of [15] for linear cryptanalysis. We observe that Simon and Simeck are rotation-invariant, so that any differential or linear attack can be repeated several times using rotations of the original distinguisher. In particular, we can exploit attack parameters with low success rates, and repeat them several times to increase the success rate. We compare our results with the best previous analysis in Table 1.

Table 1. Summary of previous and new attacks against Simon and Simeck. Attacks marked with † recover information about subkey bits, but the advantage is too low to attack the cipher. Attacks marked with ‡ use the duality between linear and differential distinguishers, which is not exact.

Cipher	Rounds	Attacked	Data	Time	Ref	Note
Simeck48/96	36	30	$2^{47.66}$	$2^{88.04}$	[25]	Linear † ‡
		32	2^{47}	$2^{90.9}$	**New**	Linear
Simeck64/128	44	37	$2^{63.09}$	$2^{121.25}$	[25]	Linear † ‡
		42	$2^{63.5}$	$2^{123.9}$	**New**	Linear
Simon96/96	52	37	2^{95}	$2^{87.2}$	[26]	Differential
		43	2^{94}	$2^{89.6}$	**New**	Linear
Simon96/144	54	38	$2^{95.2}$	2^{136}	[12]	Linear
		45	2^{95}	$2^{136.5}$	**New**	Linear
Simon128/128	68	50	2^{127}	$2^{119.2}$	[26]	Differential
		53	2^{127}	2^{121}	**New**	Linear
Simon128/192	69	51	2^{127}	$2^{183.2}$	[26]	Differential
		55	2^{127}	$2^{185.2}$	**New**	Linear
Simon128/256	72	53	$2^{127.6}$	2^{249}	[12]	Linear
		56	2^{126}	2^{249}	**New**	Linear

Outline. We begin with preliminaries about differential and linear cryptanalysis in general in Section 2. Then we apply them to Simon-like ciphers in Section 3, starting with previous results and explaining our main contribution. We explain in detail how to apply these ideas to Simeck with differential cryptanalysis (Section 4) and linear cryptanalysis (Section 5). We apply the same techniques to Simon in Section 6, and conclude in Section 7.

The code used to compute the probabilities of differentials and linear approximations (Table 4), as well as the success probability of linear attacks (Section 5), is available at https://github.com/Clustering-Simon.

1.1 Notations

The following notations are used in this paper:

n/κ	block size and key size
$x^{(i)}$	left part of the input of round i
x_j	j-th bit of x
r	number of rounds
P/C	plaintext and ciphertext
$\tilde{P}/\tilde{C}$	plaintext after the first round the ciphertext before the last round
D	data complexity
C_1	time complexity to run an attack a single time
F_W/F_R	probability distribution function for a wrong/right key guess
P_S	success probability of an attack
a, b, c	rotation constants: $f(x) = ((x \lll a) \wedge (x \lll b)) \oplus (x \lll c)$

1.2 Description of Simon and Simeck

Simonn/κ and Simeckn/κ are Feistel block ciphers with block size $n \in \{32, 48, 64, 96, 128\}$ and key size $\kappa \in \{n, 1.5n, 2n\}$. There are 10 versions of Simon, with the following parameters:

n	32	48		64		96		128		
κ	64	72	96	96	128	96	144	128	192	256
r	32*	36	36*	42	44*	52	54	68	69	72

There are 3 versions of Simeck, using a subset of the Simon parameters marked with *; in particular, Simeck has $\kappa = 2n$, and we often omit the κ parameter. The plaintext P is divided in two parts of $n/2$ bits named $x^{(0)}$ and $x^{(-1)}$, which correspond to the initialization of the left and the right parts of our Feistel network. For the round i, we denote by $x^{(i)}$ and $x^{(i-1)}$ the left and the right part of the input of this round. The round function is:

$$x^{(i+1)} = x^{(i-1)} \oplus f(x^{(i)}) \oplus k^{(i)}, \qquad \text{with}$$
$$f(x) = ((x \lll a) \wedge (x \lll b)) \oplus (x \lll c).$$

We denote by $x \lll d$ the cyclic rotation of d bits by left, $\wedge$ the bitwise AND, and $\oplus$ the bitwise exclusive or (XOR). The j-th bit of x is noted x_j where the index j is taken modulo $n/2$. The rotations of Simon are defined as $(a, b, c) = (1, 8, 2)$, while those of Simeck are defined as $(a, b, c) = (0, 5, 1)$ (the rotation amounts are independent of the block size).

Since there is no whitening key, the first and last round functions do not depend on the key. We define $\tilde{P}$ as the plaintext after the first round, $\tilde{C}$ as the ciphertext before the last round, and we use them as input for our analysis:

$$P = (x^{(0)}, x^{(-1)}) \quad \tilde{P} = (x^{(-1)} \oplus f(x^{(0)}), x^{(0)})$$
$$C = (x^{(r)}, x^{(r-1)}) \quad \tilde{C} = (x^{(r-1)}, x^{(r)} \oplus f(x^{(r-1)}))$$

The input of round 1 corresponds to $\tilde{P} \oplus (k^{(0)} \| 0^{n/2})$ (see Figure 5).

The key schedule allows to derive the subkeys $k^{(i)}$ for $0 \leq i < r$ from the master key k. First, the master key is divided into $2\kappa/n$ words $(k^{(2\kappa/n-1)}, \ldots, k^{(1)}, k^{(0)})$. Then, the subkeys $k^{(i)}$ for $i \geq 2\kappa/n$ are obtained using a recursion formula. For Simeck, the recursion is defined as

$$k^{(i+4)} = k^{(i)} \oplus f(k^{(i+1)}) \oplus C \oplus z^{(i)},$$

with C and $z^{(i)}$ constants depending on the block size and f is the same function as used in the data path. Simon uses a different key schedule, that is linear. We omit further details because our analysis does not exploit them.

2 Differential and Linear Cryptanalysis

We begin with some preliminaries on differential and linear cryptanalysis.

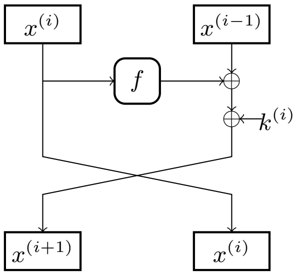

Fig. 1. Round function of Simon and Simeck.

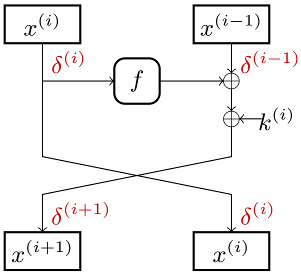

Fig. 2. Differential characteristic with probability $\Pr_x[\delta^{(i)} \xrightarrow{f} \delta^{(i-1)} \oplus \delta^{(i+1)}]$

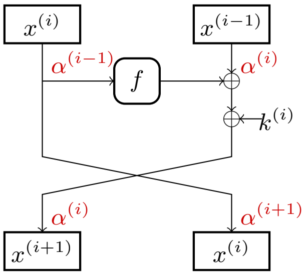

Fig. 3. Linear trail with correlation $c(\alpha^{(i-1)} \oplus \alpha^{(i+1)} \xrightarrow{f} \alpha^{(i)})$

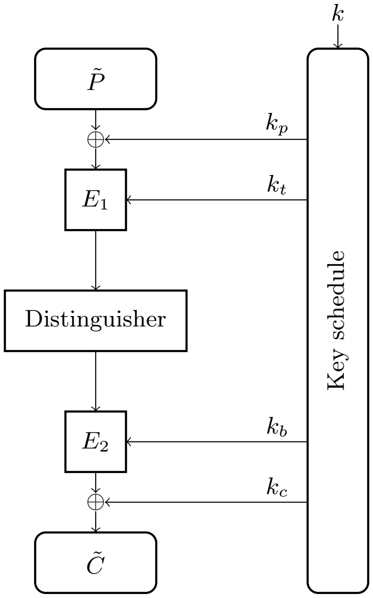

Fig. 4. General description of a cipher.

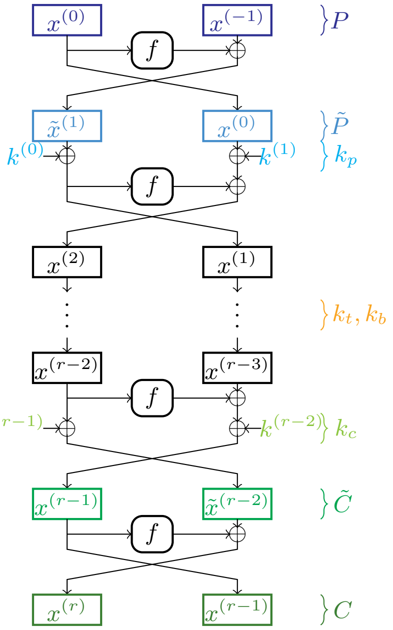

Fig. 5. Simon/Simeck with our notations: $\tilde{x}^{(1)}$ and $\tilde{x}^{(r-2)}$ respectively stand for $x^{(-1)} \oplus f(x^{(0)})$ and $x^{(r)} \oplus f(x^{(r-1)})$.

2.1 Differential cryptanalysis

Differential cryptanalysis is a technique introduced by Biham and Shamir [7,8], exploiting the propagation of differences in (reduced versions of) a cipher. Starting from a well-chosen difference δ, the distribution of $E_k(x) \oplus E_k(x \oplus \delta)$ is non-uniform, and there exist differences δ' such that $\Pr_{k,x}[E_k(x) \oplus E_k(x \oplus \delta) = \delta']$ is high (significantly higher than 2^{-n}). Such a pair (δ, δ') is called a *differential*.

In practice, we use the notion of *differential characteristic* (or trail) to estimate the probability of a differential. A differential characteristic $(\delta_0, \delta_1, \dots, \delta_r)$ specifies the intermediate state difference after each round of the function. Therefore, we can easily compute the probability that each round follows the characteristic, and we estimate the probability of the differential as the product of the probability of each round, assuming that they are independent.

More formally, we use the following notations for the probability of the round function, the probability of a characteristic, and the probability of a differential, where R denotes the round function of a cipher, and $E_k^{(r)}$ is a reduced version of the cipher with r rounds:

$$\begin{aligned}\Pr[\delta \to \delta'] &= \Pr_x[R(x) \oplus R(x \oplus \delta) = \delta'] \\ \Pr[\delta_0 \to \delta_1 \to \dots \to \delta_r] &= \Pr_{k,x}[E_k^{(i)}(x) \oplus E_k^{(i)}(x \oplus \delta_0) = \delta_i, \forall i \le r] \\ \Pr[\delta \overset{r}{\rightsquigarrow} \delta'] &= \Pr_{k,x}[E_k^{(r)}(x) \oplus E_k^{(r)}(x \oplus \delta) = \delta']\end{aligned}$$

Lai, Massey and Murphy have defined the notion of a Markov cipher (Simon and Simeck with independent round keys are Markov ciphers), where the probability of a characteristic is the product of the probabilities of the round function transitions [19]:

$$\Pr[\delta_0 \to \delta_1 \to \dots \to \delta_r] = \prod_{i=1}^{r} \Pr[\delta_{i-1} \to \delta_i] \enspace .$$

When there is a dominant characteristic, it can be used as an approximation of the probability of the differential. In general, the probability of a differential is the sum over all compatible characteristics:

$$\Pr[\delta_0 \overset{r}{\rightsquigarrow} \delta_r] = \sum_{\delta_1, \delta_2, \dots \delta_{r-1}} \prod_{i=1}^{r} \Pr[\delta_{i-1} \to \delta_i] \enspace .$$

If we write all the transition probabilities $\Pr[\delta \to \delta']$ in a differential transition matrix A, the probabilities of all r-round differentials are given by A^r, as shown by [19]. Computing A^r is infeasible for practical ciphers, but this approach can be applied to a set of predetermined characteristics, and provide a good approximation of the probability of a differential.

Differential distinguisher. In order to distinguish a cipher with a high probability differential (δ, δ') from a random permutation, we collect D ciphertexts corresponding to pairs of plaintexts $(P, P \oplus \delta)$, and we compute the number of pairs following the differential:

$$Q = \#\{P : E(P) \oplus E(P \oplus \delta) = \delta'\} \enspace .$$

The expected value of Q is $D / \Pr[\delta \rightsquigarrow \delta']$ for the cipher, and $D/2^n$ for a random permutation; therefore the distinguisher succeeds with high probability when $D = \mathcal{O}(1/\Pr[\delta \rightsquigarrow \delta'])$.

2.2 Linear cryptanalysis

Linear cryptanalysis was introduced by Matsui [22]; it uses linear approximations of the round function in order to obtain a biased approximation of the (reduced) cipher. A *linear approximation* is a pair of masks (α, α') such that the distribution of $x \cdot \alpha \oplus E_k(x) \cdot \alpha'$ is biased ($|\Pr_x[x \cdot \alpha = E_k(x) \cdot \alpha'] - 1/2| \gg 2^{-n/2}$ for most keys k), where $x \cdot y = \bigoplus_i x_i y_i$ denotes the dot product. Since the correlation is expected to be zero when averaged over all keys, we define the key-dependent correlation as follows:

$$c_k(\alpha \overset{r}{\rightsquigarrow} \alpha') = 2 \Pr_x[x \cdot \alpha = E_k^{(r)}(x) \cdot \alpha'] - 1 \ \ .$$

In practice, we use *linear trails* where a mask is specified for each intermediate state. For an iterative cipher $E_k = R_k^{(r)} \circ \cdots \circ R_k^{(2)} \circ R_k^{(1)}$, we can express the correlation $c_k(\alpha_0 \overset{r}{\rightsquigarrow} \alpha_r)$ as the sum of the correlation over all corresponding linear trails by defining the correlation of the keyed round function $R_k^{(i)}$ [14]:

$$\begin{aligned} c_k(\alpha_0 \overset{r}{\rightsquigarrow} \alpha_r) &= \sum_{\alpha_1, \alpha_2, \ldots \alpha_{r-1}} \prod_{i=1}^{r} c_k^{(i)}(\alpha_{i-1} \to \alpha_i) \\ c_k^{(i)}(\alpha_{i-1} \to \alpha_i) &= 2 \Pr_x[x \cdot \alpha = R_k^{(i)}(x) \cdot \alpha'] - 1 \ \ . \end{aligned}$$

If the cipher is a key-alternating cipher with independent round keys, the correlation of the keyed round function can be expressed in terms of the correlation of the unkeyed round function:

$$\begin{aligned} c_k(\alpha_0 \overset{r}{\rightsquigarrow} \alpha_r) &= \sum_{\alpha_1, \alpha_2, \ldots \alpha_{n-1}} (-1)^{\bigoplus_i k_i \cdot \alpha_i} \prod_{i=1}^{r} c(\alpha_{i-1} \to \alpha_i) \\ c(\alpha \to \alpha') &= 2 \Pr_x[x \cdot \alpha = R(x) \cdot \alpha'] - 1 \ \ . \end{aligned}$$

Therefore, the correlation of a linear approximation is the sum of the correlations over all linear trails, with signs that depend on the key. When there is a single dominant trail, we can approximate the correlation of the linear approximation as the correlation of the trail, up to a change of sign. However when there are several dominant trails, they can interact constructively or destructively depending on the key.

Nyberg [23] defined the expected linear potential as the expected value of the square correlation for a random key, and showed that it is equal to the sum of the squared correlation over all linear trails (assuming a key-alternating cipher with independent keys):

$$\begin{aligned} \mathrm{ELP}(\alpha_0 \overset{r}{\rightsquigarrow} \alpha_r) &= \mathrm{Exp}_k(c_k^2(\alpha_0 \overset{r}{\rightsquigarrow} \alpha_r)) \\ &= \sum_{\alpha_1, \alpha_2, \ldots \alpha_{r-1}} \prod_{i=1}^{r} c^2(\alpha_{i-1} \to \alpha_i) \ \ . \end{aligned}$$

Similarly to the differential case, we can compute the expected linear potential for all linear approximations by computing the powers of a correlation matrix C with coefficients $c^2(\alpha \to \alpha')$.

Linear distinguisher. In order to distinguish a cipher with a biased linear approximation (α, α') from a random permutation, we collect D known plaintexts/ciphertexts, and we evaluate the experimental correlation

$$Q = (\#\{P, C : P \cdot \alpha \oplus C \cdot \alpha' = 0\} - \#\{P, C : P \cdot \alpha \oplus C \cdot \alpha' = 1\})/D$$

The expected value of Q is larger (in absolute value) for the cipher than for a random permutation, and this can be detected with high probability when $D = \mathcal{O}(\mathrm{ELP}[\alpha \rightsquigarrow \alpha']^{-1})$ (see Section 5.2 for more details).

2.3 Last-round Key Recovery

In order to turn a statistical distinguisher (differential or linear) into a key recovery attack, we add a few rounds at the top and/or bottom, and partially encrypt/decrypt the available data to evaluate the statistical property. We denote the statistic used by the distinguisher as Q, and we assume that it can be evaluated by guessing only a subset of the key, shown as (k_p, k_t, k_b, k_c) in Figure 4. We let $\kappa_g = \kappa_p + \kappa_t + \kappa_b + \kappa_c$ denote the corresponding number of key bits. We denote the value obtained for key candidate k as $Q(k)$, and consider it as a random variable (depending on the choice of the encryption key, and the data set). We evaluate $Q(k)$ for all key candidates; in a naive approach, this requires $D \times 2^{\kappa_g}$ operations. However, this can often be reduced to roughly $D + 2^{\kappa_g}$ operations using algorithmic tricks (details are given in the next sections).

By analysing the theoretical behaviour of the distinguisher, we can predict the distribution of the random variables. We denote the probability distribution function of the statistic for the right key as F_R, and for wrong keys as F_W. We rank the key candidates according to $Q(k)$, and expect that the correct key will be in the top candidates if the distinguisher is strong enough (w.l.o.g., we assume that the statistic used gives a higher value for the right key).

More precisely, we aim to have the correct key among the top $2^{\kappa_g - a}$ candidates, where a is called the advantage (in bits). If the key schedule of the cipher is simple enough, the attacker can reconstruct the $2^{\kappa - a}$ master keys corresponding to these candidates and exhaustively test them. In particular, the key schedule of Simon is linear, so that master key candidates can be constructed from any subkey bits using linear algebra. The complexity of this type of attack is roughly:

$$T = D + 2^{\kappa_g} + 2^{\kappa - a} \quad .$$

In order to keep a fraction 2^{-a} of the key candidates, we set a threshold of $s = F_W^{-1}(1 - 2^{-a})$ and keep all keys with $Q(k) \geq s$. The attack succeeds if the value of Q corresponding to the right key is higher than the threshold, this happens with probability:

$$P_S = 1 - F_R(s) = 1 - F_R(F_W^{-1}(1 - 2^{-a})) \quad . \tag{1}$$

As a first condition, the parameters must satisfy:

$$D \leq 2^n \qquad D \ll 2^\kappa \qquad 2^{\kappa_g} \ll 2^\kappa \qquad 2^a \gg 1 \qquad P_S \gg 0 \tag{2}$$

Complex key schedules. When the key schedule is complex and non-linear, reconstructing the master key candidates corresponding to the κ_g recovered bits can be an issue. In particular when adding rounds on both sides of the distinguisher, the attacker recovers candidates for keys bits in the first and last rounds, but it is not possible to efficiently build the corresponding candidates for the master key. In particular, some previous attacks on Simeck [16,24,25] use a small advantage a and compute the time complexity as $2^{\kappa-a}$, but the key recovery would actually have a complexity higher than 2^{κ} with the parameters used. Instead the attacker can focus on the recovered bits on a single side of the distinguisher, and exhaustively search the missing bits, with a cost of $2^{\kappa-a+\min\{\kappa_p+\kappa_t,\kappa_b+\kappa_c\}}$.

3 Analysis of Simon-like ciphers

Since the round function of Simon-like ciphers is quadratic, we can efficiently compute the exact probability of a differential or linear transition through the function f. This was explored in details by Kölbl, Leander and Tiessen [17]:

- For a given α, there is an affine space U_α such that

  $$\Pr_x[f(\alpha \oplus x) \oplus f(x) = \beta] = \begin{cases} 2^{-\dim(U_\alpha)} & \text{if } \beta \in U_\alpha \\ 0 & \text{otherwise} \end{cases}$$

  U_α is a coset of the image of a linear function:

  $$U_\alpha = \mathrm{Img}\big(x \mapsto f(x) \oplus f(x \oplus \alpha) \oplus f(\alpha)\big) \oplus f(\alpha)$$

  Given the Feistel structure of the round function, we deduce

  $$\Pr[(\delta_L, \delta_R) \to (\delta'_L, \delta'_R)] = \begin{cases} 2^{-\dim(U_{\delta_L})} & \text{if } \delta_L = \delta'_R \text{ and } \delta_R \oplus \delta'_L \in U_{\delta_L} \\ 0 & \text{otherwise} \end{cases}$$

- For a given β, there is an affine space V_β such that

  $$c(x \cdot \alpha, f(x) \cdot \beta)^2 = \begin{cases} 2^{-\dim(V_\beta)} & \text{if } \alpha \in V_\beta \\ 0 & \text{otherwise} \end{cases}$$

  V_β is a coset of the image of a linear function:

  $$V_\beta = \mathrm{Img}\Big(x \mapsto \big((\beta \wedge (x \lll a-b)) \oplus ((\beta \wedge x) \ggg a-b)\big) \ggg b\Big) \oplus (\beta \ggg c)$$

  For the Feistel-based round function, this implies

  $$c((\alpha_L, \alpha_R) \to (\alpha'_L, \alpha'_R))^2 = \begin{cases} 2^{-\dim(V_{\alpha_R})} & \text{if } \alpha_R = \alpha'_L \text{ and } \alpha_L \oplus \alpha'_R \in V_{\alpha_R} \\ 0 & \text{otherwise} \end{cases}$$

This provides an efficient representation of the differential transition matrix A and of the squared correlation matrix C. However, computing the transitions over the full space is still infeasible for $n > 32$, because we need at least to store a vector with 2^n elements.

Table 2. An example of 12-round iterative trail (differential and linear) for Simeck. We show a list of active bits.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
δ_i	∅	0	1	0,2	3	0,2,3,4	1,2	0,2,4	3	0,2	1	0	∅	0
α_i	∅	4	3	4,2	1	4,2,1,0	3,2	4,2,0	1	4,2	3	4	∅	4

3.1 A class of high probability trails

In this work we consider a class of trails that are only active in a window of w bits of each word (e.g., the w least significant bits). Several previous works have already shown that there exist iterative trails in this class for Simeck [3,24,25] and Simon [21]; we give an example in Table 2. More generally, Simon and Simeck have a relatively slow diffusion. If a difference is restricted to the w least significant bits, it will stay on the $w+5$ (for Simeck) or $w+8$ (for Simon) least significant bits after one round. Moreover, the diffusion to bit $w+5$ (respectively $w+8$) is non-linear; if it is absorbed then the difference stays on $w+1$ (respectively $w+2$) bits only. Therefore, we expect many high probability trails in this class. We detail our results on Simeck in this section, and we discuss Simon in Section 6.

Let $w \leq n/2$ and $\varDelta_w$ be the vector space of differences active only in the w least significant bits (LSBs) of a word. Let $\varDelta_w^2$ be the product $\varDelta_w \times \varDelta_w$ where the two words are considered. For a given $\delta_0, \delta_r \in \varDelta_w$, we can compute a lower bound of the probability of the differential $\delta_0 \to \delta_r$ by summing over all characteristics with intermediate differences in $\varDelta_w^2$:

$$\Pr[\delta_0 \overset{r}{\underset{w}{\rightsquigarrow}} \delta_r] = \sum_{\delta_1, \delta_2, \dots \delta_{n-1} \in \varDelta_w^2} \prod_{i=1}^{r} \Pr[\delta_{i-1} \to \delta_i] \leq \Pr[\delta_0 \overset{r}{\rightsquigarrow} \delta_r]$$

As mentioned in Section 2.1, we can compute these values by evaluating A_w^r where the coefficients of the matrix A_w are the probabilities of transition $\Pr[\delta \to \delta']$ for all $\delta, \delta' \in \varDelta_w^2$. In order to reduce the memory requirement, we do not explicitly build the matrix A_w but we use the properties of the previous section to compute it on the fly. Moreover, we focus on the probabilities $\Pr[\delta_0 \overset{r}{\underset{w}{\rightsquigarrow}} \delta']$ for a fixed δ_0, i.e., a single line of A_w^r. Indeed, we can evaluate $A_w^r \times e_{\delta_0}$ (where e_{δ_0} is the basis vector corresponding to δ_0) using iterated matrix-vector products.

This is shown as Algorithm 1: we use a vector X to represent the probability distribution of the differences, and we update it iteratively. The complexity of the algorithm is bounded by $r \times 2^{2w} \times \max_{\alpha \in \varDelta_w} |U_\alpha|$ elementary operations. By increasing w, the lower bound is refined but the complexity increases, as seen in Figure 6. Our results show that the lower bounds grows very slowly after $w = 16$, therefore we expect to have a rather tight approximation. Moreover, we have performed experiments on 20-round distinguishers that closely match the prediction (Figure 7 and Figure 8). In practice, it takes about a week to run the algorithm with $w = 18$ and $r = 30$ using 1TB of RAM on a 48-core machine.

We have used this approach to evaluate the probability of differentials used in previous attacks against Simeck, and we find that the probability is significantly better than estimated in previous works (See Table 3). In particular, our approach covers a huge number of trails than cannot be listed individually (See Table 4).

For large numbers of rounds, the best characteristics we have identified with this search are a set of 64 characteristics with essentially the same probability, of the form (using a hexadecimal notation to represent the value in $\varDelta_w$)

$$\{(\mathtt{1},\mathtt{2}),(\mathtt{1},\mathtt{3}),(\mathtt{1},\mathtt{22}),(\mathtt{1},\mathtt{23}),(\mathtt{2},\mathtt{5}),(\mathtt{2},\mathtt{7}),(\mathtt{2},\mathtt{45}),(\mathtt{2},\mathtt{47})\}$$
$$\to$$
$$\{(\mathtt{2},\mathtt{1}),(\mathtt{3},\mathtt{1}),(\mathtt{22},\mathtt{1}),(\mathtt{23},\mathtt{1}),(\mathtt{5},\mathtt{2}),(\mathtt{7},\mathtt{2}),(\mathtt{45},\mathtt{2}),(\mathtt{47},\mathtt{2})\}$$

However, we note that the characteristic $(\mathtt{0},\mathtt{1}) \to (\mathtt{1},\mathtt{0})$ is almost as good and will lead to a more efficient key-recovery (because it has fewer active bits). Therefore, we focus on this characteristic in the following. The corresponding probabilities are given in Table 4.

3.2 Links between Linear and Differential trails

Alizadeh et al. have shown a duality between differential and linear trails in Simon [2], that also applies to Simeck. Given a differential trail with probability p:

$$(\alpha_0,\beta_0) \to (\alpha_1,\beta_1) \to \ldots \to (\alpha_r,\beta_r)$$

we can convert it into a linear trail:

$$(\overleftarrow{\beta}_0,\overleftarrow{\alpha}_0) \to (\overleftarrow{\beta}_1,\overleftarrow{\alpha}_1) \to \ldots \to (\overleftarrow{\beta}_r,\overleftarrow{\alpha}_r)$$

where $\overleftarrow{x}$ denotes bit-reversed x. If all the non-linear gates are independent, the linear trail has squared correlation p. This explains that linear distinguishers and differential distinguishers of Simon-like ciphers are very similar. However they are not equivalent: when trails are more dense, there are dependencies when two different AND gates share an input, and the probabilities of the linear and differential trail are not the same.

Since our approach applies almost identically to differential cryptanalysis and linear cryptanalysis, we have also applied it to linear cryptanalysis. We consider masks in the set $\varLambda_w^2$ active only in the w least significant bits, and we compute a lower bound on the ELP by summing over trails with intermediate masks in the set $\varLambda_w^2$. Since the diffusion of linear masks goes from most significant bits to least significant bits, the highest-bias trail with a single active bit is $(2^{w-1},0) \to (0,2^{w-1})$. For simplicity, we rotate the trail by $w-1$ bits and display it as $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$. We obtain a set of 64 (almost) optimal trails, corresponding to the bit-reversed versions of the optimal differential characteristics. We represent them after a rotation of $w-7$ bits for simplicity:

$$\{(\mathtt{20},\mathtt{40}),(\mathtt{22},\mathtt{40}),(\mathtt{60},\mathtt{40}),(\mathtt{62},\mathtt{40}),(\mathtt{50},\mathtt{20}),(\mathtt{51},\mathtt{20}),(\mathtt{70},\mathtt{20}),(\mathtt{71},\mathtt{20})\}$$
$$\to$$
$$\{(\mathtt{40},\mathtt{20}),(\mathtt{40},\mathtt{22}),(\mathtt{40},\mathtt{60}),(\mathtt{40},\mathtt{62}),(\mathtt{20},\mathtt{50}),(\mathtt{20},\mathtt{51}),(\mathtt{20},\mathtt{70}),(\mathtt{20},\mathtt{71})\}$$

Table 3. Comparison of our lower bound on the differential probability for Simeck (with $w = 18$), and estimates used in previous attacks.

Rounds	Differential	Proba (previous)	Ref	Proba (new)
26	$(\texttt{0},\texttt{11}) \to (\texttt{22},\texttt{1})$	$2^{-60.02}$	[18]	$2^{-54.16}$
26	$(\texttt{0},\texttt{11}) \to (\texttt{2},\texttt{1})$	$2^{-60.09}$	[25]	$2^{-54.16}$
27	$(\texttt{0},\texttt{11}) \to (\texttt{5},\texttt{2})$	$2^{-61.49}$	[21]	$2^{-56.06}$
27	$(\texttt{0},\texttt{11}) \to (\texttt{5},\texttt{2})$	$2^{-60.75}$	[16]	"
28	$(\texttt{0},\texttt{11}) \to (\texttt{A8},\texttt{5})$	$2^{-63.91}$	[16]	$2^{-59.16}$

Algorithm 1. Computation of $\Pr[(\delta_L, \delta_R) \overset{r}{\underset{w}{\rightsquigarrow}} (\delta'_L, \delta'_R)]$

```
$X \leftarrow [0$ for $i \in \Delta_w^2]$
$X[\delta_L, \delta_R] \leftarrow 1$
for $0 \le i < r$ do
    $Y \leftarrow [0$ for $i \in \Delta_w^2]$
    for $\alpha \in \Delta_w$ do
        for $\beta \in \Delta_w$ do
            for $\gamma \in U_\alpha$ do
                $Y[\beta, \alpha \oplus \gamma] = Y[\beta, \alpha \oplus \gamma] + 2^{-\dim(U_\alpha)} X[\alpha, \beta]$
    $X \leftarrow Y$
return $X[\delta'_L, \delta'_R]$
```

Our results are given in Table 4 (where the trail $(\texttt{1},\texttt{2}) \to (\texttt{2},\texttt{1})$ corresponds to $(\texttt{20},\texttt{40}) \to (\texttt{40},\texttt{20})$), and show that the results obtained for linear cryptanalysis and differential cryptanalysis are very close, but not identical.

3.3 Key Bits for Last-round Key Recovery

When a differential or linear distinguisher is extended into a key recovery attack, we have to study what are the key bits necessary to evaluate the statistical property after a few rounds. We denote the required key as k_p, k_t on the plaintext side, and k_b, k_c on the ciphertext side (see Figure 4), and the corresponding number of bits as κ_p, κ_t (respectively κ_b, κ_c). The total number of required key bits is denoted as $\kappa_g = \kappa_p + \kappa_t + \kappa_b + \kappa_c$. For simplicity, we focus on distinguishers with a single active bit, as used in this work.

Linear cryptanalysis. For linear cryptanalysis, we have to compute an internal bit $x_j^{(i)}$ from the plaintext (or from the ciphertext). We compute recursively the necessary key bits following the expression of the round function. In the case of Simeck (see Algorithm 3 in the full version [20]), the formula is:

$$x_j^{(i)} = (x_j^{(i-1)} \wedge x_{j+5}^{(i-1)}) \oplus x_{j+1}^{(i-1)} \oplus x_j^{(i-2)} \oplus k_j^{(i-1)} \quad . \tag{3}$$

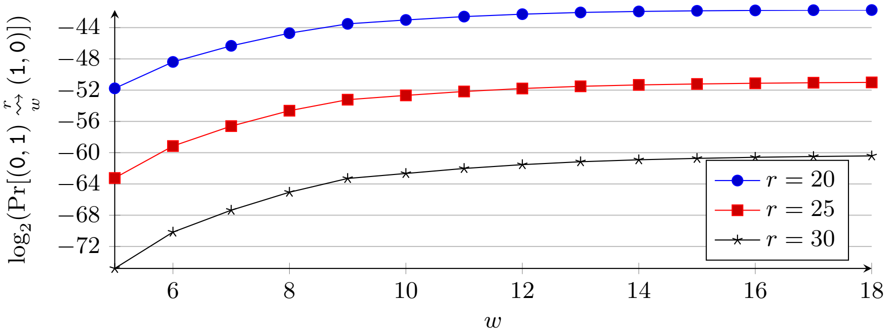

Fig. 6. Effect of w on the probability of Simeck differentials.

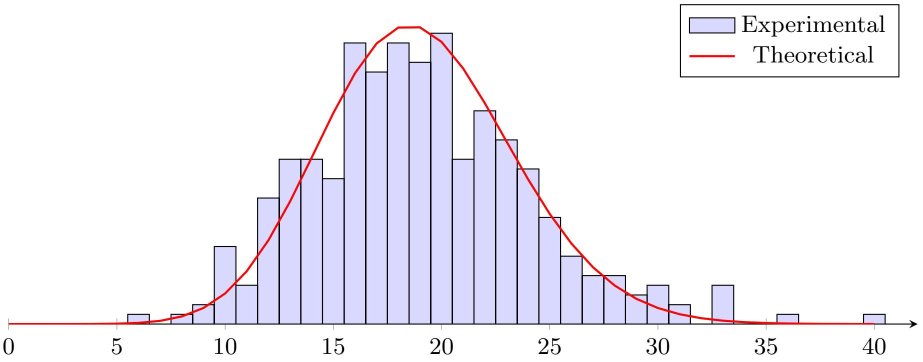

Fig. 7. Experimental verification of the 20-round differential distinguisher $(\mathtt{0},\mathtt{1}) \to (\mathtt{1},\mathtt{0})$ for Simeck64. We take 336 random keys with 2^{46} random plaintext pairs each, and we count the number of pairs following the differential. The theoretical curve is a Poisson distribution with parameter $\lambda = 2^{46} \times 2^{-41.75}$. We have 6408 good pairs in total, which gives an experimental probability of $2^{-41.75}$, matching our analysis.

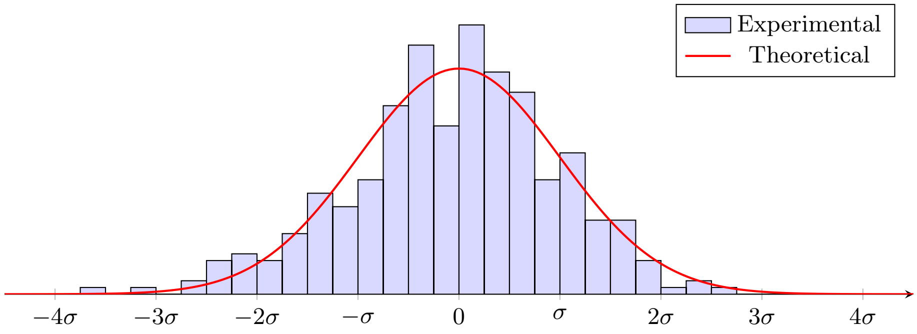

Fig. 8. Experimental verification of the 20-round linear distinguisher $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$ for Simeck64. We take 336 random keys with 2^{48} random plaintexts each, and we measure the experimental correlation over the available plaintexts. The theoretical curve is a normal distribution with parameter $\sigma^2 = \text{ELP} + B/N \approx 2^{-41.74} + 2^{-48} \approx 2^{-41.72}$. The average square correlation observed is $2^{-41.7}$, matching the analysis.

Table 4. Comparison of the probability of differentials and the linear potential of linear approximations for Simeck ($\log_2$, computed with $w = 18$). We also give the total number of trails included in the bound in parenthesis ($\log_2$)

	Differential			Linear		
Rounds	$(0,1) \to (1,0)$		$(1,2) \to (2,1)$	$(1,0) \to (0,1)$		$(1,2) \to (2,1)$
1	0	(0)	$-\infty$	0	(0)	$-\infty$
2	$-\infty$		−4.00	$-\infty$		−4.000
3	$-\infty$		−4.00	$-\infty$		−4.000
4	$-\infty$		$-\infty$	$-\infty$		$-\infty$
5	$-\infty$		$-\infty$	$-\infty$		$-\infty$
6	$-\infty$		$-\infty$	$-\infty$		$-\infty$
7	$-\infty$		$-\infty$	$-\infty$		$-\infty$
8	$-\infty$		$-\infty$	$-\infty$		$-\infty$
9	$-\infty$		$-\infty$	$-\infty$		$-\infty$
10	$-\infty$		$-\infty$	$-\infty$		$-\infty$
11	−23.25	(28.0)	−27.25	−23.81	(23.9)	−27.81
12	−26.40	(36.2)	−26.17	−26.39	(31.7)	−26.68
13	−28.02	(47.2)	−26.90	−27.98	(42.0)	−27.31
14	−30.06	(58.2)	−29.59	−29.95	(52.5)	−29.56
15	−31.93	(70.8)	−31.37	−31.86	(64.9)	−31.29
16	−33.96	(83.0)	−33.35	−33.76	(77.0)	−33.24
17	−35.48	(95.2)	−35.25	−35.09	(88.8)	−35.12
18	−37.95	(107.5)	−37.12	−37.94	(100.7)	−36.85
19	−39.92	(119.7)	−38.97	−39.93	(112.6)	−38.67
20	−41.75	(131.9)	−41.26	−41.74	(124.5)	−41.25
21	−43.47	(144.1)	−43.17	−43.56	(136.4)	−43.17
22	−45.42	(156.3)	−44.97	−45.45	(148.4)	−44.99
23	−47.27	(168.5)	−46.77	−47.30	(160.3)	−46.83
24	−49.14	(180.7)	−48.68	−49.14	(172.2)	−48.71
25	−51.01	(192.9)	−50.54	−51.00	(184.1)	−50.56
26	−52.88	(205.2)	−52.41	−52.86	(196.0)	−52.40
27	−54.72	(217.4)	−54.28	−54.68	(207.9)	−54.26
28	−56.64	(229.6)	−56.15	−56.59	(219.8)	−56.11
29	−58.53	(241.8)	−58.02	−58.47	(231.7)	−57.96
30	−60.41	(254.0)	−59.92	−60.36	(243.6)	−59.86
31	−62.29	(266.2)	−61.81	−62.24	(255.5)	−61.75
32	−64.17	(278.4)	−63.69	−64.12	(267.4)	−63.63
33	−66.05	(290.6)	−65.57	−66.00	(279.3)	−65.51
34	−67.93	(302.9)	−67.45	−67.90	(291.2)	−67.40
35	−69.81	(315.1)	−69.33	−69.78	(303.1)	−69.28
36	−71.69	(327.3)	−71.21	−71.65	(315.0)	−71.17
37	−73.57	(339.5)	−73.09	−73.53	(326.9)	−73.05
38	−75.45	(351.7)	−74.97	−75.40	(338.8)	−74.92

The algorithm returns the key bits that $x_j^{(i)}$ depends on: a list of (linear combinations of) key bits with a linear effect on $x_j^{(i)}$, and a list of (linear combinations of) key bits with non-linear effect.

Differential cryptanalysis. For differential cryptanalysis, we have to determine whether a pair of plaintexts (or ciphertexts) P, P' reaches a specific internal state difference $\Delta x^{(i)} = x^{(i)} \oplus x'^{(i)}$ after a few rounds. However, the plaintexts are not chosen randomly. Instead they have a specific pattern of differences that is fixed in advance, and known to potentially reach the target difference.

More precisely, we follow the approach of [24,26] to track the propagation of differences in the additional rounds, assuming that a pair follows the differential. We use the rule that the output difference of an AND operator is 0 if and only if its input differences are $(0, 0)$ and we identify bits with a fixed difference (0 or 1) and those with an unknown difference $(*)$, as shown in Table 6. The number of bits with a fixed difference for round i is denoted ℓ_i.

Next, we determine sufficient bit conditions for pairs following the input and output constraints of the extended path: a small set of bit conditions that ensure that we get the desired difference at the input and output of our differential when they are satisfied. The other differences are automatically satisfied if the bit conditions of the external rounds have already been checked. To determine the sufficient bit conditions, we proceed round by round, from the outer to the inner rounds. For each bit with a fixed output difference (0 or 1, not $*$), we look at the input of the associated AND operator, and if they are different from $(0, 0)$, then we add the condition corresponding to this output difference to the set of sufficient bit conditions.

Finally, for each sufficient bit condition, we compute the set of subkey bits required to check the condition. The goal is to determine which subkey bits are needed to compute a difference $\Delta x_j^{(i)}$. For Simeck, we use the following relation:

$$\begin{aligned}\Delta x_j^{(i)} = (\Delta x_j^{(i-1)} \wedge x_{j+5}^{(i-1)}) \oplus (\Delta x_{j+5}^{(i-1)} \wedge x_j^{(i-1)}) \oplus (\Delta x_j^{(i-1)} \wedge \Delta x_{j+5}^{(i-1)}) \\ \oplus \Delta x_{j+1}^{(i-1)} \oplus \Delta x_j^{(i-2)}. \quad (4)\end{aligned}$$

The resulting algorithm for the upper part is given in the full version [20].

Comparison. Table 5 shows the number of key bits to guess for a linear or differential attack on Simeck with a single active bit, depending on the number of rounds. (Similar tables for Simon are given in the full version [20]). The list of bits returned by the previous algorithms is simplified in order to keep only linearly independent combinations of bits. Moreover, we report the number of independent bits after removing bits that can be computed using the key schedule. We see that linear cryptanalysis requires a lower number of key bits to guess.

We now explain in detail key-recovery attacks against Simeck based on those distinguishers. We consider differential attacks in Section 4, and linear attacks in Section 5.

Table 5. Comparison of key recovery rounds for differential and linear attacks against Simeck64/128.

Key bits	Differential		Linear	
Rounds	total	independent	total	independent
1	0	0	0	0
2	2	2	2	2
3	9	9	7	7
4	27	27	16	16
5	56	56	30	30
6	88	88	50	48
7	120	114	75	68
8			104	88

4 Key-recovery attacks using Differential Cryptanalysis

In this section, we detail differential key-recovery attacks on Simeck. We explain the dynamic key-guessing technique [24,26] in Section 4.1 and we give our results in Section 4.2.

4.1 The dynamic key-guessing technique

Offline phase. Starting from a differential $\varDelta_i \to \varDelta_o$ covering R rounds, we append r_0 rounds before and r_1 rounds after and we build the extended differential as explained in Section 3.3. Then we identify sufficient bit conditions as shown in Table 6, and we identify the key bits required to check whether a pair reaches the difference specified by the distinguisher.

Online phase. Here we describe how the attack takes place from the construction of the pairs to the recovery of the possible master keys. First, we build structures of plaintexts such that the bits with a fixed difference in round 1 are identical for all the plaintexts in each structure. Each structure is composed of $2^{n-\ell_1}$ plaintexts and if D denotes the data complexity, there are $D \times 2^{\ell_1 - n}$ structures. The structures are associated in pairs such that the differences between the two structures for bits with a fixed difference in round 1 correspond to the desired difference in the extended path. For each structure, the corresponding ciphertexts are saved in a table according to their value in the bits with fixed difference in $\widetilde{C}$. This allows to filter the pairs at the output and $2^{2(n-\ell_1)-\ell_{r-1}}$ pairs remains in each pair of structures.

Our goal is to associate partial key guesses to each of these pairs, such that they validate the internal differential. We proceed in a dynamic way. Round by round and for each sufficient bit condition, we associate to each pair the possible combinations of key bits that lead to the desired difference in the input and output of the distinguisher, according to the extended path. Some pairs and/or combinations of key guesses are progressively eliminated when they create an incompatibility with the required differential path. For each pair, when a combination of key bits that leads to Δ_i and Δ_o is found, we increment the counter corresponding to this combination of key bits. In total, at the end of the procedure we have incremented $\lambda_W \times 2^{\kappa_g}$ counters on average, with λ_W the average value of a counter for a wrong key guess. This process is partially compatible with the use of key schedule relations on one side of the distinguisher. Indeed, we guess key bits independently, and filter the combinations of bits which do not verify the relations given by the key schedule afterwards. The details of the attack (round by round) must be considered to calculate the time complexity of the attack.

When all pairs have been processed, we set a threshold s, and for each counter greater than s, we have to find the corresponding master keys. The information given by the κ_g key bits is separated into 2 parts: we have bits of information on the first subkeys, and others on the last subkeys. Due to the non-linear key-schedule of Simeck, we cannot directly combine this information and do an exhaustive search on the $\kappa - \kappa_g$ missing bits. Instead, we use only the side for which we have the most information bits, and then we do an exhaustive search on the missing bits.

Complexity and success probability. In order to compute the complexity and the success probability of this attack, we need to estimate the average value of the counters for the right key and for wrong key guesses: we denote those values λ_R and λ_W.

As explained before, structures are associated in pairs. Let S_1 and S_2 denote two structures that form a pair. For each plaintext $\tilde{P}_1$ in S_1 and for each key guess, we compute $\tilde{P}_2 = E_1^{-1}(E_1(\tilde{P}_1) \oplus \Delta_i)$ such that we have the relation $E_1(\tilde{P}_1) \oplus E_1(\tilde{P}_2) = \Delta_i$. $\tilde{P}_2$ necessarily belongs to S_2, due to structures construction. So, if D denotes the data complexity of the attack, for each key guess we have $D/2$ pairs with the desired difference at the input of our distinguisher.

For the right key guess, if our differential occurs with probability p, we have $\lambda_R = p \times D/2$ pairs that satisfy Δ_o. By construction, all these pairs belong to the structures and pass the filters. On the other hand, for wrong key guesses, $\tilde{P}_1$ and $\tilde{P}_2$ don't actually have a difference Δ_i at the input of the distinguisher with the real encryption key, so the probability that they have a fixed difference Δ_o at the output is $1/2^n$. The value of the counter for wrong key guesses is expected to be $\lambda_W = D/2^{n+1}$ on average[1].

We model the counter associated to the right key guess and a wrong key guess as a Poisson distribution with parameters λ_R and λ_W. We denote the cumulative distributive function as F_R and F_W. The probability that a counter associated to a wrong key guess is greater than a threshold s is $1 - F_W(s)$, and the expected number of counters greater than s is $2^{\kappa_g} \cdot (1 - F_W(s))$.

[1] The same result can be obtained using the formulas in [24] and the code provided by the authors of this paper.

Let $\kappa_{\min}$ and $\kappa_{\max}$ be the minimum and the maximum of $\kappa_p + \kappa_t$ and $\kappa_b + \kappa_c$. The cost of reconstructing the master keys from the remaining combinations is $2^{\kappa_g} \cdot (1 - F_W(s)) \times 2^{\kappa - \kappa_{\max}} = 2^{\kappa + \kappa_{\min}} \cdot (1 - F_W(s))$. The time complexity is therefore determined by this term, but also by the data and the time required to scan all the counters at the end of the key-recovery: 2^{κ_g}. Knowing that most of the counters are at 0, this term could be reduced to the number of remaining pairs: $2^{\kappa_g} \cdot \lambda_W$. The time complexity and success probability are:

$$C_1 = D + 2^{\kappa_g} \cdot \lambda_W + 2^{\kappa + \kappa_{\min}} \cdot (1 - F_W(s))$$

$$P_S = 1 - F_R(s)$$

4.2 40-round Key-recovery on Simeck64/128

We apply the methods described in the previous subsection with the differential $(\mathtt{0}, \mathtt{1})$ to $(\mathtt{1}, \mathtt{0})$ covering 30 rounds with probability $p = 2^{-60.41}$. We append 3 rounds before and 7 rounds after. The extended path is given in Table 6 and the details of the bits to guess round by round are given in the full version [20] (Table 12). Round by round, we use the sufficient bit conditions from Table 6 to guess the key bits that lead to the desired differences. When possible, we filter using the relations of the key schedule. In the rightmost column, we detail the time complexity of each step starting from 2^t pairs. To compute this complexity more precisely, we split the round 33 in two parts corresponding to the two sufficient bit conditions. In total, the complexity of guessing the key bits leading to Δ_i and Δ_o, and incrementing the corresponding counters is 2^{t+71}. During this step, $\kappa_{\min} = 9$ bits from the first subkeys and $\kappa_{\max} = 114$ from the last subkeys are guessed.

Attack parameters. If all the codebook ($D = 2^{64}$) is taken, knowing that $\ell_1 = 57$ and $\ell_{39} = 19$, we split the data into 2^{57} structures of 2^7 plaintexts and after constructing our pairs of structures and filtering the ciphertexts $\tilde{C}$, there remain $2^{57-1} \times 2^{7\times 2}/2^{19} = 2^{51}$ pairs. So $t = 51$ and the time complexity for the counter incrementing part is $2^{t+71} = 2^{122}$. The average value for the counter of the right key guess is $\lambda_R = p \times D/2 = 2^{2.59}$. And for a bad key guess, we expect the counter to be close to $\lambda_W = D/2^{64-1} = 2^{-1}$. So, if we choose to set the threshold s at 5, the complexity is $2^{120.89} + 2^{122} = 2^{122.54}$ with a success probability of 55%.

We show parameters with different time/data trade-offs, as well as parameters for other variants of Simeck in Table 7.

5 Key-recovery attacks using Linear Cryptanalysis

The first description of a last-round key recovery attack using linear cryptanalysis was given by Matsui's Algorithm 2 [22]. We consider a biased linear approximation $P' \cdot \alpha \oplus C' \cdot \beta$ with P' and C' intermediate values after a few rounds of

Table 6. Extended path for 40 rounds of Simeck64/128. Red bold bits represent the sufficient bit conditions.

r	Differential path		ℓ_i
0	000000000000000000000∗000∗∗001∗∗	0000000000000000∗000∗∗00∗∗∗01∗∗∗	50
1	00000000000000000000000000∗0001∗	000000000000000000000∗000∗∗001∗∗	57
2	000000000000000000000**0**000**00**000**01**	00000000000000000000000000∗0001∗	62
3	00000000000000000000000000**0**0000**0**	00000000000000000000000000000001	64
	30-round differential ($3 \rightarrow 33$)		
33	00000000000000000000000000000001	00000000000000000000000000**0**0000**0**	64
34	00000000000000000000000000∗0001∗	000000000000000000000**0**000**00**000**01**	62
35	000000000000000000000∗000∗∗001∗∗	0000000000000000**0**000**00**00**00**∗00**01**∗	57
36	0000000000000000∗000∗∗00∗∗∗01∗∗∗	00000000000**0**000**00**00**00**∗0**00**∗∗0**01**∗∗	50
37	00000000000∗000∗∗00∗∗∗0∗∗∗∗1∗∗∗∗	000000**0**000**00**00**00**∗0**00**∗∗**00**∗∗∗**01**∗∗∗	41
38	000000∗000∗∗00∗∗∗0∗∗∗∗∗∗∗∗∗∗∗∗∗∗	0**0**000**00**00**00**∗0**00**∗∗**00**∗∗∗**0**∗∗∗∗**1**∗∗∗∗	30
39	0∗000∗∗00∗∗∗0∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗	000000∗000∗∗00∗∗∗0∗∗∗∗∗∗∗∗∗∗∗∗∗∗	19
40	∗∗00∗∗∗0∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗	0∗000∗∗00∗∗∗0∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗∗	10

Table 7. Attack parameters for differential attacks on Simeck. C_1 is the time complexity to run the attack a single time, and the corresponding success probability is P_S. The average time is obtained as C_1/P_S, by assuming that the attack is repeated until it succeeds, using rotations of the initial differential.

Cipher	Rounds	$\kappa_{\min}$	$\kappa_{\max}$	D	λ_R	λ_W	s	C_1	P_S	Time
Simeck64/128	$40 = 3 + 30 + 7$	9	114	2^{64}	$2^{2.59}$	2^{-1}	6	$2^{122} + 2^{117.07}$	0.40	$2^{123.4}$
Simeck64/128	$40 = 3 + 30 + 7$	9	114	2^{64}	$2^{2.59}$	2^{-1}	5	$2^{122} + 2^{120.89}$	0.55	$2^{123.4}$
Simeck64/128	$40 = 3 + 30 + 7$	9	114	2^{63}	$2^{1.59}$	2^{-2}	4	$2^{121} + 2^{119.79}$	0.19	$2^{123.9}$
Simeck48/96	$30 = 2 + 22 + 6$	2	74	2^{48}	$2^{1.58}$	2^{-1}	5	$2^{75} + 2^{81.9}$	0.08	$2^{85.5}$
Simeck48/96	$30 = 2 + 22 + 6$	2	74	2^{47}	$2^{0.58}$	2^{-2}	3	$2^{74} + 2^{85.1}$	0.06	$2^{89.1}$
Simeck32/64	$22 = 3 + 13 + 6$	8	51	2^{32}	$2^{2.98}$	2^{-1}	5	$2^{58} + 2^{55.9}$	0.80	$2^{58.6}$
Simeck32/64	$22 = 3 + 13 + 6$	8	51	2^{31}	$2^{1.98}$	2^{-2}	4	$2^{57} + 2^{54.8}$	0.36	$2^{58.8}$

encryption/decryption. Given a set of D known plaintexts/ciphertexts pair (P, C), we can compute the intermediate values P' and C' for each partial key guess $k_g = (k_p, k_t, k_c, k_b)$ for the first and/or last rounds, and compute the experimental correlation of the linear approximation:

$$\begin{aligned} q(k_p, k_t, k_c, k_b) &= \frac{1}{D}\big(\#\{P, C : P' \cdot \alpha = C' \cdot \beta\} - \#\{P, C : P' \cdot \alpha \neq C' \cdot \beta\}\big) \\ &= \frac{1}{D}\sum_{P,C}(-1)^{P' \cdot \alpha \oplus C' \cdot \beta} \end{aligned}$$

$P' \cdot \alpha$ and $C' \cdot \beta$ are computed as a function of the partial key and some bits of the plaintexts/ciphertexts denoted as $\chi_c(C)$ and $\chi_p(P)$ respectively (we assume that the bit positions correspond to the key bits in k_p and k_c):

$$\begin{aligned} P' \cdot \alpha &= f(k_t, k_p \oplus \chi_p(P)) \\ C' \cdot \beta &= g(k_b, k_c \oplus \chi_c(C)) \end{aligned}$$

5.1 The FWT Approach of [13,15]

The time complexity of the attack is dominated by the time necessary to compute the statistic for all key candidates. Several tricks have been introduced to make this step more efficient. Since the values of $P' \cdot \alpha$ and $C' \cdot \beta$ do not depend on the full plaintext/ciphertext, we can "compress" the dataset using a distillation phase where we only count how many plaintext/ciphertext pairs reach each value of those bits [22]:

$$\begin{aligned}
&q(k_p, k_t, k_c, k_b) \\
&\quad = \frac{1}{D} \sum_{P,C} (-1)^{f(k_t, k_p \oplus \chi_p(P)) \oplus g(k_b, k_c \oplus \chi_c(C))} \\
&\quad = \frac{1}{D} \sum_{i \in \mathbb{F}_2^{\kappa_p}} \sum_{j \in \mathbb{F}_2^{\kappa_c}} \#\{P, C : \chi_p(P) = i, \chi_c(C) = j\} \times (-1)^{f(k_t, k_p \oplus i) \oplus g(k_b, k_c \oplus j)}
\end{aligned}$$

We remark that the previous expression is actually a convolution:

$$= \frac{1}{D} \sum_{i,j} \phi(i,j) \times \psi_{k_t,k_b}(k_p \oplus i, k_c \oplus j) = \frac{1}{D} (\phi * \psi_{k_t,k_b})(k_p, k_c),$$

with

$$\begin{aligned}
\phi(x,y) &= \#\{P, C : \chi_p(P) = x, \chi_c(C) = y\} \\
\psi_{k_t,k_b}(x,y) &= (-1)^{f(k_t,x) \oplus g(k_b,y)}
\end{aligned}$$

Therefore, for a given k_t, k_b, we can evaluate $q(k_p, k_t, k_c, k_b)$ for all k_p, k_c with complexity $\tilde{\mathcal{O}}(2^{\kappa_p + \kappa_c})$ using a Fast Walsh Transform. This was first observed in [13] (with additional rounds on one side only), and then generalized in [15]. The time complexity of the analysis is reduced to $\tilde{\mathcal{O}}(D + 2^{\kappa_g})$.

5.2 Statistical Models to Estimate the Success Probability

We follow the analysis of Blondeau and Nyberg [10,11], taking into account the impact of the variance of the correlation due to the random key, and the sampling model with a factor B depending on the type of attack: $B = 1$ if the plaintexts are randomly chosen with repetition, and $B = (2^n - D)/(2^n - 1)$ if they are distinct (in the following, we assume distinct plaintexts).

Single dominant characteristic. When there is a single dominant characteristic with absolute bias ε, the correlation of the approximation is either ε or $-\varepsilon$ depending of the key. The empirical correlation for the right key follows one of two possible normal distributions, with parameters

$$\mu_R = \pm\varepsilon \qquad\qquad \sigma_R^2 = B/D + 2^{-n}.$$

When the key guess is wrong, we assume that the computed statistic follows the correlation of a random permutation; it follows a normal distribution with parameters

$$\mu_W = 0 \qquad\qquad \sigma_W^2 = B/D + 2^{-n}.$$

Since there are two possible distributions for the right key, we have to slightly modify the analysis of (1). For an attack with gain a, we set a threshold $s = F_W^{-1}(1 - 2^{-a-1}) = \sigma_W \Phi^{-1}(1 - 2^{-a-1})$ and keep key candidates with $|q| \geq s$. The attack succeeds with probability $P_S = 1 - F_R(s)$ when $\mu_R > 0$, and $P_S = F_R(-s)$ otherwise:

$$P_S = \Phi\left(\frac{|\varepsilon| - \sigma_W \Phi^{-1}(1 - 2^{-a-1})}{\sigma_R}\right),$$

where Φ is the cumulative distribution function of the standard normal distribution.

Single approximation with many trails. When using a single linear hull with many high correlation trails (rather than a dominant trail), the correlations for the right and wrong keys follow normal distributions with parameters:

$$\begin{aligned} \mu_R &= 0 & \sigma_R^2 &= B/D + \mathrm{ELP} \\ \mu_W &= 0 & \sigma_W^2 &= B/D + 2^{-n}, \end{aligned}$$

Following [10], we estimate the expected linear potential ELP using the correlations ε_τ for characteristics τ in a set $\mathcal{S}$ of dominating characteristics:

$$\mathrm{ELP} \approx 2^{-n} + \sum_{\tau \in \mathcal{S}} \varepsilon_\tau^2,$$

using the results of Section 3.1 to compute $\sum_{\tau \in \mathcal{S}} \varepsilon_\tau^2$ with $\mathcal{S}$ the set of characteristics with masks in Λ_w.

The distributions are both centered on zero, but since the variance is larger for the right key, we can sort the keys according to the absolute value of the measured correlation, and we expect a larger value for the right key than for wrong keys. More precisely, using a threshold $s = \sigma_W \Phi^{-1}(1 - 2^{-a-1})$ on the absolute value of the correlation, the success rate is given by [10, Theorem 2]:

$$P_S = 2 - 2\Phi\left(\frac{\sigma_W}{\sigma_R} \Phi^{-1}(1 - 2^{-a-1})\right).$$

Equivalently, we can consider the squared correlation, which follows a χ^2 distribution with one degree of freedom, and use the generic formula (1) with the following distributions:

$$F_R/\sigma_R \sim \chi_1^2 \qquad\qquad F_W/\sigma_W \sim \chi_1^2$$

Multiple approximations. When using M linear approximations, there are different ways to exploit the information to rank the keys. Again, we follow the analysis of [10], and we rank the keys according to

$$Q(k) = \sum q_i(k)^2$$

According to [10], we can model the statistics for the right key as a Gaussian distribution with parameters

$$\begin{aligned} \sigma_R^2 &= 2B^2M + 4BD\sum_i \mathrm{ELP}_i + 2D^2 \sum_i \mathrm{ELP}_i^2 \\ \mu_R &= BM + D\sum_i \mathrm{ELP}_i \end{aligned}$$

On the other hand the statistic for the wrong key is proportional to a χ^2 distribution with M degrees of freedom:

$$F_W/(B + D2^{-n}) \sim \chi^2_M$$

In our analysis, we consider either a single approximation $(\mathtt{1},\mathtt{0}) \rightarrow (\mathtt{0},\mathtt{1})$, or the approximation $(\mathtt{1},\mathtt{0}) \rightarrow (\mathtt{0},\mathtt{1})$ combined with lower quality approximations that can be used with the same key bits.

5.3 12-round Key-recovery

We apply the previous techniques to Simeck64, starting from the linear approximation $(\mathtt{0},\mathtt{1}) \rightarrow (\mathtt{0},\mathtt{1})$, and adding 8 rounds on the plaintext side and 4 rounds on the ciphertext side. Following Equation 3, $x_0^{(8)}$ can be computed from $\kappa_p = 54$ bits of $\tilde{P}$, $\kappa_p = 54$ bits of the whitening key $k_p = k^{(0)} \| k^{(1)}$, and $\kappa_t = 50$ additional key bits. Similarly, $x_0^{(r-5)}$ can be computed from $\kappa_c = 14$ bits of $\tilde{C}$, $\kappa_c = 14$ bits of the whitening key $k^{(r-1)} \| k^{(r-2)}$, and $\kappa_b = 2$ additional key bits:

$$\begin{aligned} k_p &= k^{(0)}_{[0,-1,\dots,-23,-25,-26,-27,-30,-31]}, k^{(1)}_{[0,-1,\dots,-18,-20,-21,-22,-25,-26,-30]} \\ k_t &= k^{(2)}_{[0,-1,\dots,-13,-15,-16,-17,-20,-21,-25]}, \\ &\quad k^{(3)}_{[0,-1,-2,-3,-5,-6,-7,-8,-10,-11,-12,-15,-16,-20]}, \\ &\quad k^{(4)}_{[0,-1,-2,-5,-6,-7,-10,-11,-15]}, k^{(5)}_{[0,-1,-5,-6,-10]}, k^{(6)}_{[0,-5]} \\ k_b &= k^{(r-3)}_{[0,-5]} \\ k_c &= k^{(r-1)}_{[0,-1,-2,-5,-6,-7,-10,-11,-15]}, k^{(r-2)}_{[0,-1,-5,-6,-10]} \end{aligned}$$

We ignore bits that have a linear effect because they only flip the sign of the imbalance. Moreover, we can use key schedule relations to reduce κ_t by 2:

$$\begin{aligned} k_0^{(6)} &= k_0^{(2)} \oplus k_{-1}^{(3)} \oplus k_0^{(3)} \wedge k_{-5}^{(3)} \oplus c_0^{(6)} \\ k_{-5}^{(6)} &= k_{-5}^{(2)} \oplus k_{-6}^{(3)} \oplus k_{-5}^{(3)} \wedge k_{-10}^{(3)} \oplus c_{-5}^{(6)} \end{aligned}$$

There are 14 additional relations between bits of κ_t and κ_p.

The attack is decomposed in three phases:

Distillation phase. Compute $\phi(x,y) = \#\{P, C : \chi_p(P) = x, \chi_c(C) = y\}$ for $0 \leq x < 2^{\kappa_p}$, $0 \leq y < 2^{\kappa_c}$.
This only requires to set up $2^{\kappa_p+\kappa_c}$ counters, and to iterate over the D available plaintext/ciphertext pairs.

Analysis phase. For each guess of k_t, k_b, for all $0 \leq x < 2^{\kappa_p}$, $0 \leq y < 2^{\kappa_c}$, compute $\psi_{k_t,k_b}(x,y) = (-1)^{f(k_t,x) \oplus g(k_b,y)}$, then evaluate the convolution $\phi * \psi_{k_t,k_b}$ using the Fast Walsh Transform.
For each k_t, k_b, this requires $2^{\kappa_p+\kappa_c}$ evaluations of f and g to generate ψ_{k_t,k_b}, and $3(\kappa_p+\kappa_c)2^{\kappa_p+\kappa_c}$ additions and $2^{\kappa_p+\kappa_c}$ multiplications to evaluate the convolution. Assuming that the cost of $\kappa_p+\kappa_c$ additions and the cost of a multiplication are comparable to the cost of an encryption call, the total complexity of the analysis phase is $\mathcal{O}(2^{\kappa_g})$ using a memory of size $2^{\kappa_p+\kappa_c}$.

Search phase. For all keys with $q(k_p, k_t, k_c, k_b) \geq s$, exhaustively try all master keys corresponding to k_p, k_t, k_c, k_b.
With a threshold $s = F_W^{-1}(1-2^{-a})$ we expect a fraction 2^{-a} of the keys to remain. We iterate over 2^{88+16} candidates k_p, k_t, k_c, k_b that satisfy the 14 key schedule equations between k_t and k_p, keep only the 88 independent bits of k_p, k_t for keys meeting the threshold, and exhaustively search the remaining 40 bits, with a complexity of $2^{88+16-a} \times 2^{40} = 2^{144-a}$.

With our parameters, we have $\kappa_g = 118$ (after removing the two relations between bits of k_t). Using the Walsh transform pruning technique of [15] (and partially precomputing the Walsh transform of ψ), the complexity of the analysis phase is reduced to[2]:

$$68\rho_A 2^{68} + 2\rho_M 2^{118} + \rho_A 2^{64}(2^{54} + 39 \times 2^{40}) + \rho_A 2^{90}(2^{14} + 13 \times 2^{14}) \approx 2\rho_M 2^{118}$$

with ρ_A the cost of an addition, and ρ_M the cost of a multiplication. Assuming that 2 multiplications correspond to roughly one evaluation of the cipher, we end up with a complexity of 2^{κ_g}. This variant uses a memory of $2^{\kappa_p+\kappa_c} + 2^{\kappa_p+\kappa_t} + 2^{\kappa_c+\kappa_b} = 2^{68} + 2^{102} + 2^{16} \approx 2^{102}$ elements.

5.4 Attack Parameters

We use this attack to target 41 or 42 rounds of Simeck64, and smaller variants of Simeck, with various time/data trade-offs summarized in Table 8. We explain two attacks in detail here and include others in the full version [20] (Appendix B).

42-round Simeck64 with $2^{63.5}$ plaintexts. The ELP of the linear approximation $(1,0) \to (0,1)$ over 30 rounds is $2^{-64} + 2^{-60.36} = 2^{-60.25}$. The complexity of the analysis phase is 2^{118}. With an advantage of $a = 24$, the complexity of the search phase is 2^{120} and the success probability is $P_S = 8.3\%$.

[2] With their notations, we have $k_0 = 54$, $k_1 = 50$, $k_2 = 2$, $k_3 = 14$, $l_{12} = 2$, $l_0 = 14$, $l_3 = 0$

Table 8. Attack parameters for linear attacks on Simeck. C_1 is the time complexity to run the attack a single time, and the corresponding success probability is P_S. The average time is obtained as C_1/P_S.

Cipher	Rounds	#app.	Capacity	D	Adv.	C_1	P_S	Time
Simeck64	$41 = 7{+}29{+}5$	5	$2^{-57.34}$	2^{63}	52	$5 \times 2^{105} + 2^{106}$	0.23	2^{110}
	$41 = 8{+}29{+}4$	1	$2^{-58.44}$	2^{62}	26	$2^{118} + 2^{118}$	0.11	$2^{122.2}$
	$42 = 8{+}30{+}4$	1	$2^{-60.25}$	$2^{63.5}$	24	$2^{118} + 2^{120}$	0.08	$2^{123.9}$
	$42 = 8{+}30{+}4$	1	$2^{-60.25}$	2^{64}	29	$2^{118} + 2^{115}$	0.10	$2^{121.5}$
Simeck48	$32 = 7{+}21{+}4$	1	$2^{-43.50}$	2^{47}	26	$2^{87} + 2^{86}$	0.10	$2^{90.9}$
Simeck32	$23 = 5{+}13{+}5$	1	$2^{-27.68}$	$2^{31.5}$	37	$2^{58} + 2^{56}$	0.07	$2^{62.2}$

Since Simeck is rotation-invariant, we can repeat the attack 32 times by rotating the linear approximation used. On average we expect the attack to succeed after $1/P_S = 12$ attempts, leading to an average complexity of $2^{123.9}$. If we fix this as the maximum complexity, we expect a success rate of roughly $1 - 1/e \approx 63\%$.

41-round Simeck64 with 2^{63} plaintexts. Alternatively, we can use multiple linear approximations to reduce the time complexity of a 41-round attack. We use the following 29-round linear approximations:

$$\begin{aligned}
(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1}) &: \mathrm{ELP} = 2^{-64} + 2^{-58.47} \\
(\mathtt{1},\mathtt{0}) \to (\mathtt{1},\mathtt{0}) &: \mathrm{ELP} = 2^{-64} + 2^{-60.36} \\
(\mathtt{1},\mathtt{0}) \to (\mathtt{1},\mathtt{1}) &: \mathrm{ELP} = 2^{-64} + 2^{-60.36} \\
(\mathtt{0},\mathtt{1}) \to (\mathtt{0},\mathtt{1}) &: \mathrm{ELP} = 2^{-64} + 2^{-60.36} \\
(\mathtt{1},\mathtt{1}) \to (\mathtt{0},\mathtt{1}) &: \mathrm{ELP} = 2^{-64} + 2^{-60.36}
\end{aligned}$$

The extra approximations have been chosen because the corresponding masks can be computed from the same keys bits as the main approximation $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$; they have a combined capacity of $2^{-57.39}$. Thanks to the higher capacity, we can aim for a higher advantage $a = 52$, and obtain a success rate of 23% with 2^{63} plaintexts. Therefore, we split the key recovery rounds as 7 rounds on the plaintext side and 5 on the ciphertext side (rather than 8 and 4), leading to parameters $\kappa_p = 45$, $\kappa_t = 30$, $\kappa_c = 23$, $\kappa_b = 7$; this reduces the complexity of the analysis phase to 5×2^{105}, while the search phase has a complexity of $2^{128+23+7-52} = 2^{106}$.

5.5 Experimentations

We have performed experimentations to verify the theory leading to the probabilities of success P_S. To do this, we take a set of D plaintext/ciphertext pairs and we compute the experimental correlation $Q(k)$ for the right key and for a random sample of wrong keys. We choose an advantage a and we consider

Table 9. Comparison of the theoretical (P_S) and experimental success probability for linear attacks. We perform 1000 experiments, taking D pairs of plaintext/ciphertext and testing whether the correlation associated to the right key is among the 2^{-a} highest correlations with a sample of random wrong keys.

Cipher	Rounds	D	# app.	capacity	# wrong keys	Adv.	Success	P_S
Simeck32/64	15	2^{31}	1	$2^{-30.73}$	2^8	5	7.4%	9.9%
Simeck32/64	15	2^{31}	5	$2^{-29.05}$	2^8	5	9%	7.4%
Simeck32/64	16	2^{31}	1	$2^{-31.59}$	2^8	5	4.9%	4.5%
Simeck32/64	16	2^{31}	5	$2^{-29.44}$	2^8	5	3.6%	2.3%
Simeck64/128	12	2^{28}	1	$2^{-26.39}$	2^{12}	10	9.9%	10.1%
Simeck64/128	12	2^{28}	5	$2^{-25.17}$	2^{12}	10	14.8%	14.8%

that a success is obtained when the correlation of the right key is among the 2^{-a} highest correlations. This experiment was repeated 1000 times with random keys to compute an experimental success probability. Our results are presented in Table 9. We compare attacks with a single approximation $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$ and attacks with five approximations $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$, $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$, $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$, $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$, $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$, as used in the 41-round attack with 2^{63} plaintexts. The experimental probability of success is close to the prediction, confirming that both the model for the success probability, and our estimation of the ELP are accurate.

For Simeck32, we compute the exact ELP of the linear approximation using our algorithm with $w = 16$; we obtain an ELP of $2^{-30.73}$ over 15 rounds for $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$ and $2^{-31.59}$ for the four other approximations.

6 Application to Simon

Similarly to Simeck, previous results [21] have shown the existence of iterative trails for Simon with a single active bit in the input and output, where all intermediate states fit in a small window of active bits. This makes Simon an interesting target for our analysis, just like Simeck. In this section we focus on linear cryptanalysis, and we obtain improved attack against Simon96 and Simon128, gaining between 3 and 7 rounds compared to previous attacks.

Previous works have shown that Simon offers a higher security against differential and linear cryptanalysis than Simeck. In particular, iterative trails have a higher weight, and require a larger window of active bits. Moreover, we notice that the trail $(\texttt{1},\texttt{0}) \to (\texttt{0},\texttt{1})$ is only possible for round numbers of the form $4r + 1$, because some bits have a linear update when the trail is limited to a window smaller than the word size. More precisely, for linear trails, the high order bits follow a pattern

$$(\texttt{10*\ldots},\texttt{0*0\ldots}) \to (\texttt{0*0\ldots},\texttt{10*\ldots}) \to (\texttt{10*\ldots},\texttt{0*1\ldots}) \to (\texttt{0*1\ldots},\texttt{10*\ldots}) \to (\texttt{10*\ldots},\texttt{0*0\ldots})$$

Since the transition matrix is sparser than for Simeck we can run our analysis with larger values of w. The lower bound on the ELP that we obtain for the trail

Table 10. Attack parameters for linear cryptanalysis of Simon.

Cipher	Rounds	$\kappa_p, \kappa_t, \kappa_c, \kappa_b$	D	a	C_1	P_S	Time	Approx
S128/256	$56 = 8{+}41{+}7$	80, 65, 64, 37	2^{126}	10	$2^{246} + 2^{246}$	0.26	2^{249}	$(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$
S128/192	$55 = 7{+}42{+}6$	64, 37, 56, 23	2^{127}	10	$2^{180} + 2^{182}$	0.13	$2^{185.2}$	$(\mathtt{4},\mathtt{0}) \to (\mathtt{4},\mathtt{1})$
S128/128	$53 = 6{+}42{+}5$	47, 18, 38, 9	2^{127}	10	$2^{112} + 2^{118}$	0.13	2^{121}	$(\mathtt{4},\mathtt{0}) \to (\mathtt{4},\mathtt{1})$
S96/144	$45 = 6{+}33{+}6$	47, 18, 47, 18	2^{95}	10	$2^{130} + 2^{134}$	0.19	$2^{136.5}$	$(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$
S96/96	$43 = 5{+}33{+}5$	30, 7, 30, 7	2^{94}	10	$2^{74} + 2^{86}$	0.08	$2^{89.6}$	$(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$

Table 11. ELP of the linear approximation $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$ and probability of the differential $(\mathtt{0},\mathtt{1}) \to (\mathtt{1},\mathtt{0})$ for Simon ($\log_2$, computed with $w = 19$)

r	13	17	21	25	29	33	37	41	45
ELP	−41.99	−46.30	−67.87	−77.90	−87.25	−92.60	−113.06	−123.07	−132.95
Pr	−40.68	−47.31	−67.56	−78.08	−86.96	−94.62	−113.67	−124.22	−133.66

$(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$ is given in Table 11. The bounds show a smaller linear potential (and differential probability) for Simon than for Simeck. However, we see in Figure 9 that the linear potential still increases significantly with the window size w; this indicates that our bound is not as tight as on Simeck. Further work is needed to capture the full clustering effect on Simon, and this could further reduce the security margin of the cipher.

The approximations $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$ can be extended by one round on either side with correlation 2^{-2}, leading to two active bits. After rotating the approximation by two bits to $(\mathtt{4},\mathtt{0}) \xrightarrow{r} (\mathtt{0},\mathtt{4})$ with ELP c, the extended approximations are $(\mathtt{4},\mathtt{0}) \xrightarrow{r+1} (\mathtt{4},\mathtt{1})$ and $(\mathtt{1},\mathtt{4}) \xrightarrow{r+1} (\mathtt{0},\mathtt{4})$ with ELP $2^{-2}c$, and $(\mathtt{1},\mathtt{4}) \xrightarrow{r+2} (\mathtt{4},\mathtt{1})$ with ELP $2^{-4}c$.

We summarize the parameters of the best attacks we have identified against Simon96 and Simon128 in Table 10; our analysis does not seem to improve previous results on Simon32, Simon48 and Simon64. As shown in the full version [20] (Table 13 and 14), we don't have any key-schedule relation between bits of k_t or k_b, but we have relations between k_t and k_p or between k_b and k_c. Therefore we can use the Walsh transform pruning technique of [15], as in the Simeck attacks. We explain two attacks in detail below.

56-rounds Simon128/256. We use the 41-round linear approximation $(\mathtt{1},\mathtt{0}) \to (\mathtt{0},\mathtt{1})$ with ELP lower bound of $2^{-123.07}+2^{-128}$. We add 8 rounds on the plaintext side, and 7 rounds on the ciphertext side, obtaining parameters $\kappa_p = 80$, $\kappa_t = 65$, $\kappa_c = 64$, $\kappa_b = 37$. The complexity of the analysis phase is about 2^{246}. With 2^{126} data, and an advantage $a = 10$, we have a success probability of 26% and the search phase has a complexity of 2^{246}. Finally we obtain an average complexity of $(2^{246} + 2^{246})/0.26 \approx 2^{249}$.

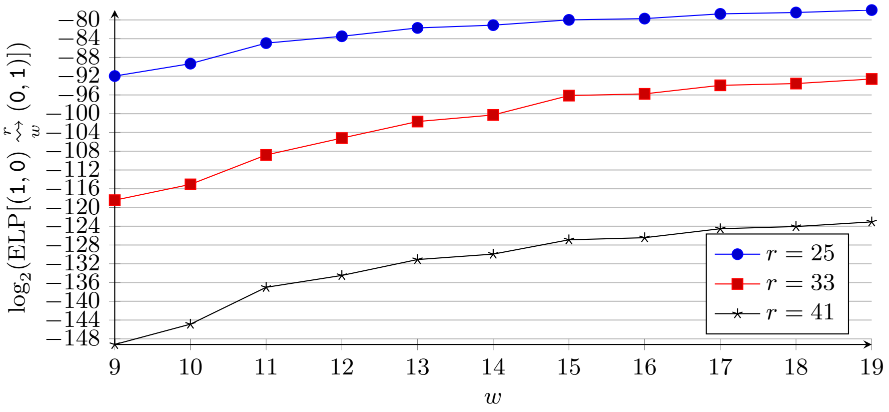

Fig. 9. Effect of w on the probability of Simon linear hulls.

55-rounds Simon128/192. For Simon128/192, we use the 42-round linear approximation $(\mathtt{4}, \mathtt{0}) \to (\mathtt{4}, \mathtt{1})$ with an ELP lower bound of $2^{-125.07} + 2^{-128}$. We add 7 rounds on the plaintext side, and 6 rounds on the ciphertext side, obtaining parameters $\kappa_p = 64$, $\kappa_t = 37$, $\kappa_c = 56$, $\kappa_b = 23$. The complexity of the analysis phase is about 2^{180}. With 2^{127} data, and an advantage $a = 10$, we have a success probability of 13% and the search phase has a complexity of 2^{182}. Finally we obtain an average complexity of $(2^{180} + 2^{182})/0.13 \approx 2^{185.2}$.

45-rounds Simon96/144. We use the 33-round linear approximation $(\mathtt{1}, \mathtt{0}) \to (\mathtt{0}, \mathtt{1})$ with an ELP lower bound of $2^{-93.57} + 2^{-96}$. We add 6 rounds on each side, obtaining parameters $\kappa_p = 47$, $\kappa_t = 18$, $\kappa_c = 47$, $\kappa_b = 18$. The complexity of the analysis phase is about 2^{130}. With 2^{95} data, and an advantage $a = 10$, we have a success probability of 19% and the search phase has a complexity of 2^{134}. Finally we obtain an average complexity of $(2^{130} + 2^{134})/0.19 \approx 2^{136.5}$.

7 Perspectives

Our work provides the first attack against 42-round Simeck64, showing that the security margin is very slim (the full version has 44 rounds). Moreover, if the designers of Simeck had proposed a 128-bit variant with the same number of rounds as Simon128, the full version would be broken by our analysis.

We also improve significantly previous attacks on Simon. In particular we show that Simon96/144 only has 17% of the rounds as security margin, while the designers wrote [4]:

> *"After almost 4 years of concerted effort by academic researchers, the various versions of Simon and Speck retain a margin averaging around 30%, and in every case over 25%. The design team's analysis when making stepping decisions was consistent with these numbers."*

Comparison of differential and linear cryptanalysis. Our work shows that differential and linear attacks against Simon and Simeck are very similar. The differential characteristics and linear approximations are almost equivalent; we use trails with a single input/output bit in both cases, and the differential probability p is almost the same as the linear potential ELP (See Table 4). Using advanced techniques (dynamic key-guessing and Fast Walsh Transform respectively), both attacks have a complexity that is essentially $D + 2^{\kappa_g}$ (with $D = \mathcal{O}(1/p)$ or $D = \mathcal{O}(1/\operatorname{ELP})$ respectively).

However, there is an important difference in the key-recovery part. As seen in Table 5, we have to guess more key bits for differential cryptanalysis than for linear cryptanalysis (for the same number of additional rounds). This explains why linear cryptanalysis is more efficient than differential cryptanalysis on those ciphers, as shown by previous analysis.

Impact of the rotations. The main difference between Simeck and Simon is the value of the rotations of the round function. In order to find which combinations of the rotation constants a, b and c would be a bad or a good choice, we reuse Algorithm 1 to obtain a lower bound for the probability of a differential for several values of a, b and c. We set the following parameters: $w = 13$, $r = 30$, and $0 \leq a, b, c \leq 10$. A lower bound of the probability of all the differentials with input $(\delta_L, \delta_R) = (\mathtt{0}, \mathtt{1})$ is computed, and this allows to confirm that the following trivial conditions must be verified: a, b and c must all be different, c must be different from 0, and a, b and c must be of different parity: the three shifts must not be all even or all odd.

We also notice that (a, b, c) must not be of the form $(i, 2j - i, j)$ or $(2j - i, i, j)$ and that the bias we observe strongly decreases when c increases. However, this does not allow us to conclude that taking a large value of c ensures a better security since it is possible that there exists other differentials with high bias. Our approach does not allow to conclude when a and b are smaller than c because in this case, the difference cannot remain in a fixed window after a large number of rounds. This is due to the fact that the difference generated by the linear part can never cancel out with the difference of the non-linear part.

Alternative class. Instead of using differences or masks in a fixed window of w bits, we could consider low-weight values. Indeed, this class also includes the iterative trails given in previous works, and should contain many high-probability trails. We have implemented the search algorithm with this alternative class, using 32-bit words of weight at most 5 (a set of $2^{17.9}$ values).

In terms of resources, this requires roughly the same amount of memory as with a window of size $w = 18$, but it runs about 50 times faster, because there are fewer possible transitions at each round: the probability to reach a value with weight lower than 5 is smaller than the probability to stay in a fixed window. However, the bounds given with this class on Simeck and Simon are not competitive with those obtained with a fixed window. On Simeck, using words of weight 5 gives probabilities comparable to using a window with $w = 9$, largely below $w = 18$. On Simon, the gap is smaller: we obtain results similar to using a window of size $w = 15$. In both cases, using a fixed window requires fewer resources (time and memory) to achieve the same quality of results.

Acknowledgements. The second author is funded by a grant from Région Ile-de-France. and the third author by ERC ADG 740972 (ALGSTRONGCRYPTO). This work was also supported by ANR grant ANR-20-CE48-0017 (SELECT).

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential cryptanalysis of round-reduced Simon and Speck. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 525–545. Springer, Heidelberg (Mar 2015)
2. Alizadeh, J., AlKhzaimi, H., Aref, M.R., Bagheri, N., Gauravaram, P., Kumar, A., Lauridsen, M.M., Sanadhya, S.K.: Cryptanalysis of SIMON variants with connections. In: RFIDSec. Lecture Notes in Computer Science, vol. 8651, pp. 90–107. Springer (2014)
3. Bagheri, N.: Linear cryptanalysis of reduced-round SIMECK variants. In: Biryukov, A., Goyal, V. (eds.) INDOCRYPT 2015. LNCS, vol. 9462, pp. 140–152. Springer, Heidelberg (Dec 2015)
4. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: Notes on the design and analysis of SIMON and SPECK. Cryptology ePrint Archive, Report 2017/560 (2017), http://eprint.iacr.org/2017/560
5. Beaulieu, R., Treatman-Clark, S., Shors, D., Weeks, B., Smith, J., Wingers, L.: The simon and speck lightweight block ciphers. In: 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC). pp. 1–6 (2015)
6. Beierle, C.: Pen and paper arguments for SIMON and SIMON-like designs. In: Zikas, V., De Prisco, R. (eds.) SCN 16. LNCS, vol. 9841, pp. 431–446. Springer, Heidelberg (Aug / Sep 2016)
7. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. In: Menezes, A.J., Vanstone, S.A. (eds.) CRYPTO'90. LNCS, vol. 537, pp. 2–21. Springer, Heidelberg (Aug 1991)
8. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology **4**(1), 3–72 (Jan 1991)
9. Biryukov, A., Roy, A., Velichkov, V.: Differential analysis of block ciphers SIMON and SPECK. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 546–570. Springer, Heidelberg (Mar 2015)
10. Blondeau, C., Nyberg, K.: Improved parameter estimates for correlation and capacity deviates in linear cryptanalysis. IACR Trans. Symm. Cryptol. **2016**(2), 162–191 (2016), http://tosc.iacr.org/index.php/ToSC/article/view/570
11. Blondeau, C., Nyberg, K.: Joint data and key distribution of simple, multiple, and multidimensional linear cryptanalysis test statistic and its impact to data complexity. Des. Codes Cryptogr. **82**(1-2), 319–349 (2017)
12. Chen, H., Wang, X.: Improved linear hull attack on round-reduced simon with dynamic key-guessing techniques. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 428–449. Springer, Heidelberg (Mar 2016)
13. Collard, B., Standaert, F.X., Quisquater, J.J.: Improving the time complexity of Matsui's linear cryptanalysis. In: Nam, K.H., Rhee, G. (eds.) ICISC 07. LNCS, vol. 4817, pp. 77–88. Springer, Heidelberg (Nov 2007)

14. Daemen, J., Govaerts, R., Vandewalle, J.: Correlation matrices. In: Preneel, B. (ed.) FSE'94. LNCS, vol. 1008, pp. 275–285. Springer, Heidelberg (Dec 1995)
15. Flórez-Gutiérrez, A., Naya-Plasencia, M.: Improving key-recovery in linear attacks: Application to 28-round PRESENT. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 221–249. Springer, Heidelberg (May 2020)
16. Huang, M., Wang, L., Zhang, Y.: Improved automatic search algorithm for differential and linear cryptanalysis on SIMECK and the applications. In: Naccache, D., Xu, S., Qing, S., Samarati, P., Blanc, G., Lu, R., Zhang, Z., Meddahi, A. (eds.) ICICS 18. LNCS, vol. 11149, pp. 664–681. Springer, Heidelberg (Oct 2018)
17. Kölbl, S., Leander, G., Tiessen, T.: Observations on the SIMON block cipher family. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 161–185. Springer, Heidelberg (Aug 2015)
18. Kölbl, S., Roy, A.: A brief comparison of simon and simeck. In: Bogdanov, A. (ed.) Lightweight Cryptography for Security and Privacy - 5th International Workshop, LightSec 2016, Aksaray, Turkey, September 21-22, 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10098, pp. 69–88. Springer (2016)
19. Lai, X., Massey, J.L., Murphy, S.: Markov ciphers and differential cryptanalysis. In: Davies, D.W. (ed.) EUROCRYPT'91. LNCS, vol. 547, pp. 17–38. Springer, Heidelberg (Apr 1991)
20. Leurent, G., Pernot, C., Schrottenloher, A.: Clustering effect in simon and simeck. Cryptology ePrint Archive, Report 2021/1198 (2021), https://ia.cr/2021/1198
21. Liu, Z., Li, Y., Wang, M.: Optimal differential trails in SIMON-like ciphers. IACR Trans. Symm. Cryptol. **2017**(1), 358–379 (2017)
22. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT'93. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (May 1994)
23. Nyberg, K.: Linear approximation of block ciphers (rump session). In: Santis, A.D. (ed.) EUROCRYPT'94. LNCS, vol. 950, pp. 439–444. Springer, Heidelberg (May 1995)
24. Qiao, K., Hu, L., Sun, S.: Differential security evaluation of simeck with dynamic key-guessing techniques. In: Camp, O., Furnell, S., Mori, P. (eds.) Proceedings of the 2nd International Conference on Information Systems Security and Privacy, ICISSP 2016, Rome, Italy, February 19-21, 2016. pp. 74–84. SciTePress (2016)
25. Qin, L., Chen, H., Wang, X.: Linear hull attack on round-reduced simeck with dynamic key-guessing techniques. In: Liu, J.K., Steinfeld, R. (eds.) ACISP 16, Part II. LNCS, vol. 9723, pp. 409–424. Springer, Heidelberg (Jul 2016)
26. Wang, N., Wang, X., Jia, K., Zhao, J.: Differential attacks on reduced SIMON versions with dynamic key-guessing techniques. Sci. China Inf. Sci. **61**(9), 098103:1–098103:3 (2018)
27. Wang, X., Wu, B., Hou, L., Lin, D.: Automatic search for related-key differential trails in SIMON-like block ciphers based on MILP. In: Chen, L., Manulis, M., Schneider, S. (eds.) ISC 2018. LNCS, vol. 11060, pp. 116–131. Springer, Heidelberg (Sep 2018)
28. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 307–329. Springer, Heidelberg (Sep 2015)