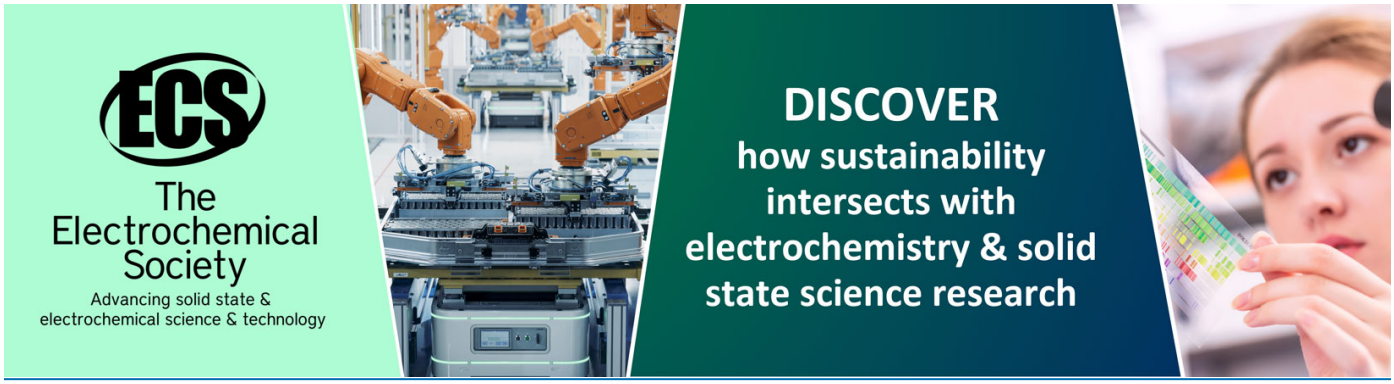
Differential-Linear Cryptanalsis on SIMECK32/64 and SIMON32/64



Differential-Linear Cryptanalsis on SIMECK32/64 and SIMON32/64

Feng Zhang[1*[0000-0003-1161-5978]], Feng Li[1], Wenzheng Zhang[1]

[1] Science and Technology on Communication Security Laboratory, Chengdu 610041, Sichuan, China

[*]Corresponding author's e-mail: sleepaloner@163.com

Abstract. In this paper, we give differential-linear cryptanalysis of SIMON, which is a family of lightweight block ciphers published by the National Security Agency, and SIMECK, which is a family of lightweight block ciphers proposed by Yang et al. Firstly, all input difference and output masks with one active bit are traversed to obtain a 9-round SIMON32/64 differential-linear distinguisher and a 10-round SIMECK32/64 differential-linear distinguisher. Then, a 12-round SIMON32/64 differential-linear distinguisher with bias $2^{-12.69}$ and a 13-round SIMECK32/64 differential-linear distinguisher with bias $2^{-14.03}$ can be obtained by searching one round of differential characteristics forward and two rounds of linear approximations backward. The dynamic key guessing technique proposed by Wang et al. has excellent advantages in the SIMON-like cipher key recovery process. Therefore, we have applied it to differential-linear cryptanalysis. Then, the 12-round SIMON32/64 differential-linear distinguisher is extended forward by four rounds and backward by four rounds to attack the 20-round SIMON32/64 with time complexity $2^{55.68}$ and data complexity 2^{28}. And the 13-round SIMECk32/64 differential-linear distinguisher is extended forward by four rounds and backward by four rounds to attack the 21-round SIMECK32/64 with time complexity $2^{50.67}$ and data complexity 2^{30}. These are the best differential-linear cryptanalysis results for SIMON32/64 and SIMECK32/64 in the open literature.

1. Introduction

With the rapid development of the Internet of Things and other related industries, micro-computing devices are being used more and more extensively. Still, due to their limited computing and storage capacity, traditional block cipher cannot be applied. As a result, the industry has introduced the concept of lightweight block ciphers, which have the advantages of small resource consumption, low operating power and high operating efficiency compared with traditional block ciphers. In recent years, scholars have proposed many lightweight block ciphers, such as GIFT, TWINE, SIMON, SPECK, etc. Among the lightweight block ciphers, one uses only three basic operations, namely Addition, Rotation and XOR, called ARX cipher. There is also an ARX cipher that replaces Addition with AND, also known as the AND-RX cipher. The advantage of the ARX cipher is that it is very elementary in design and cheap to implement in hardware and software, making it ideal for a wide range of resource-constrained devices.

SIMON [1] is a family of lightweight block ciphers published by the National Security Agency in 2013, with a Feistel structure and ten versions, offering significant advantages in terms of hardware implementation. SIMECK [2] is a family of lightweight block ciphers proposed by Yang et al. at CHES 2015, with three versions. It combines the round function of SIMON and the key schedule of SPECK. Both SIMON and SIMECK are very typical AND-RX ciphers.

1.1. Related Works.

Cryptanalysis on SIMON and SIMECK includes differential cryptanalysis, linear cryptanalysis, impossible differential cryptanalysis, zero-correlation linear cryptanalysis, integral cryptanalysis, etc. In 2014, Wang et al. [3] proposed dynamic key-guessing techniques for the key recovery process of differential cryptanalysis and attacked the 21-round SIMON32/64 based on a 13-round differential distinguisher. In 2015, Chen et al. [4] used dynamic key-guessing in linear cryptanalysis and attacked the 23-round SIMON32/64 based on a 13-round linear distinguisher. In 2015, Chen et al. [5] applied dynamic key-guessing to impossible differential cryptanalysis, attacking the 19-round SIMON32/64 based on an 11-round impossible differential distinguisher. In 2018, Chu et al. [6] applied dynamic key-guessing to integral cryptanalysis. They attacked the 24-round SIMON32/64 based on a 14-round integral distinguisher, which is the best cryptanalysis result for the SIMON32/64 in single-key conditions. In 2015, Qiao et al. [7] used dynamic key-guessing to attack the 22-round SIMECK32/64 based on a 13-round differential distinguisher. In 2016, Qin et al. [8] applied dynamic key-guessing to attack the 23-round SIMECK32/64 based on a 13-round linear distinguisher. That is the best cryptanalysis result for the SIMECK32/64 in single-key conditions.

In 2018, Chen et al. [9] gave the first differential-linear cryptanalysis of SIMON32/64. They combined a 2-round differential with probability 1 with a 13-round linear approximation with bias $2^{-15.68}$ to construct a 15-round differential-linear distinguisher with bias $2^{-30.36}$, and then attacked SIMON32 for 18 and 19 rounds. However, for the distinguisher they obtained, at least $2^{58.72}$ pairs of plaintexts satisfying input difference are needed to distinguish the cipher from the random permutation, which is clearly not achieved. In 2022, Hu et al. [15] constructed a 13-round differential-linear distinguisher for SIMON32/64. Then, the 15-round and 16-round SIMON32/64 were attacked.

It can be found that the dynamic key-guessing technique is very effective in the SIMON-like cipher key recovery process, allowing 8 to 10 rounds of attack on the distinguisher. Similarly, it can be found that the differential-linear cryptanalysis for SIMON32/64 can only attack 3 to 4 rounds on the distinguisher without the dynamic key guessing technique. Therefore, applying dynamic key-guessing techniques to differential-linear cryptanalysis would be interesting.

Table 1. Summary of differential, linear type cryptanalysis

Cipher	Attack rounds	Attack	Data	Time	Reference
SIMON32/64	15	Differential-Linear	2^{26}	$2^{27.09}$	[15]
	16	Differential-Linear	2^{26}	$2^{40.59}$	[15]
	18	Differential-Linear	2^{32}	2^{19}	[9]
	19	Differential-Linear	2^{32}	2^{35}	[9]
	20	Differential-Linear	2^{28}	$2^{55.68}$	Section 5
	23	Linear	$2^{31.19}$	$2^{61.84}$	[4]
SIMECK32/64	19	Differential	2^{31}	2^{40}	[11]
	20	Impossible Differential	2^{32}	$2^{62.5}$	[2]
	20	Differential-Linear	2^{24}	$2^{44.76}$	Section 4.3
	21	Zero Correlation	2^{32}	$2^{58.78}$	[12]
	21	Differential-Linear	2^{30}	$2^{50.67}$	Section 4.3
	22	Differential	2^{32}	$2^{57.9}$	[7]
	23	Linear	$2^{31.91}$	$2^{61.78}$	[8]

1.2. Our Contributions.

In this paper, we first use statistical analysis to search for suitable differential-linear distinguishers. Considering the subsequent key recovery process, we traverse the output difference and output mask with one active bit. We use statistical analysis to calculate the bias for each distinguisher. We can obtain a 12-round SIMECK32/64 differential-linear distinguisher with bias $2^{-11.21}$, a 13-round SIMECK32/64 differential-linear distinguisher with bias $2^{-14.03}$ and a 12-round SIMON32/64

differential-linear distinguisher with bias $2^{-12.69}$. We next apply differential-linear cryptanalysis based on these three distinguishers.

We notice that dynamic key-guessing techniques are widely used in the key recovery process for SIMON-like ciphers. Still, there is no example of their application in differential-linear cryptanalysis. A general procedure for dynamic key-guessing under differential-linear cryptanalysis is therefore given. We extend the differential-linear distinguisher forward by four rounds, backward by four rounds, and then perform a differential-linear attack. We attack the 20-round SIMON32/64 based on the 12-round SIMON32/64 differential-linear distinguisher with data complexity 2^{28} and time complexity $2^{55.68}$. We attack the 21-round SIMECK32/64 based on the 13-round SIMECK32/64 differential-linear distinguisher with data complexity 2^{30} and time complexity $2^{50.67}$. These are the best differential-linear cryptanalysis result for SIMON32/64 and SIMECK32/64 in the open literature. A summary of the differential, linear type cryptanalysis for SIMON32/64 and SIMECK32/64 is given in Table 1.

1.3. Outline.

The rest of this paper is organized as follows. Section 2 describes the notation and a brief introduction to the SIMON and SIMECK ciphers. In Section 3, we introduce differential-linear cryptanalysis, giving an algorithm for searching the differential-linear distinguishers of the SIMON and SIMECK ciphers, obtaining a 12-round differential-linear distinguisher for SIMON32/64 and a 13-round differential-linear distinguisher for SIMECK32/64. In Section 4, we apply dynamic key-guessing to the differential-linear cryptanalysis key recovery process, give a specific attack process for the 20-round SIMECK32/64, and attack the 21-round SIMECK32/64. In Section 5, we provide a differential-linear attack on the 20-round of SIMON32/64. In Section 6, the whole paper is concluded.

2. Preliminaries

2.1. Notations

Table 2. Notations

Notation	Meaning	Notation	Meaning
X^{r-1}	input of the r-th round	ΔX	the difference of X and X^*
K^{r-1}	subkey used in the r-th round	X ⋘ r	the left rotation of X by r bits
L^{r-1}	left half input of the r-th round	$\oplus$	bitwise XOR
R^{r-1}	right half input of the r-th round	$\cap$	bitwise AND
X_i	the i-th bit of X, the left is the lowest bit	$+$	addition operation
K_i	the i-th bit of K, the left is the lowest bit	%	modular operation

2.2. Descriptions of SIMON and SIMECK

The SIMON family is a Feistel structure with a block size of 2n, where n is 16, 24, 32, 48, 64, and a key size of mn, where m is 2, 3, 4, denoted as SIMON2n/mn. All versions of SIMON are given in Table 3. The round function of SIMON is given in Figure 1, where $\alpha = 8, \beta = 1, \gamma = 2$.

For the convenience of subsequent operations, let $L^i = X_n^i, X_{n+1}^i, ..., X_{2n-1}^i$, $R^i = X_0^i, X_1^i, ..., X_{n-1}^i$ and $K^i = K_0^i, K_1^i, ..., K_{n-1}^i$, then the wheel function is as follows:

$$X_{j+n}^i = \left(X_{(j+1)\%n+n}^{i-1} \cap X_{(j+8)\%n+n}^{i-1}\right) \oplus X_{(j+2)\%n+n}^{i-1} \oplus X_j^{i-1} \oplus K_j^{i-1},$$
$$X_j^i = X_{j+n}^{i-1},$$

where $j = 0, 1, ..., n-1$.

Table 3. All Versions of SIMON

Block size 2n	Key size mn	Rounds r
32	64	32
48	72/96	36
64	96/128	42/44

96	96/144	52/54
128	128/192/256	68/69/72

The SIMECK family is a Feistel structure with a block size of 2n, where n is 16, 24, 32, and a key size of mn, where m is 2, 3, 4, denoted as SIMECK2n/mn. There are three versions: SIMECK32/64(32 rounds), SIMECK48/96(36 rounds) and SIMECK64/128(44 rounds). The round function of SMIECK is basically the same as SIMON, the difference lies in the parameters of the rotation shift, where $\alpha = 5, \beta = 0, \gamma = 1$. The round function is denoted as follows:

$$X_{j+n}^{i} = \left(X_{(j+0)\%n+n}^{i-1} \cap X_{(j+5)\%n+n}^{i-1}\right) \oplus X_{(j+1)\%n+n}^{i-1} \oplus X_{j}^{i-1} \oplus K_{j}^{i-1}, \quad (1)$$

$$X_{j}^{i} = X_{j+n}^{i-1},$$

where $j = 0, 1, ..., n-1$.

For the key extensions of SIMON and SIMECK, we refer the readers to [1] and [2] for more details.

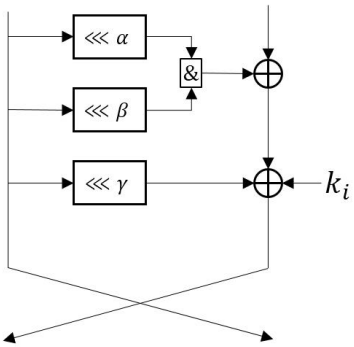


Figure 1. The Round Function of SIMON and SIMECK

3. Differential-linear Cryptanalysis

Biham and Shamir introduced differential cryptanalysis, the idea of which is to distinguish a cipher from a random permutation by finding a high probability difference. Let $\Delta_I, \Delta_O \in \mathbb{F}_2^n$ be the input difference and output difference of the cipher E_0, respectively, the probabilities of which are defined as follows.

$$Pr(\Delta_I \to \Delta_O) = Pr_{x \in \mathbb{F}_2^n}(E_0(x) \oplus E_0(x \oplus \Delta_I) = \Delta_O) \quad (2)$$

Matsui proposed linear analysis, the idea of which is to distinguish a cipher from a random permutation by finding a high correlation linear approximation. Let $\lambda_I, \lambda_O \in \mathbb{F}_2^n$ be the input mask and output mask of the cipherE_1, respectively, then its correlation is defined as follows.

$$Cor = 2|\, Pr_{x \in \mathbb{F}_2^n}(x \odot \lambda_I = E_1(x) \odot \lambda_O) - \frac{1}{2}| \quad (3)$$

The strength of the linear approximation will also be expressed in bias, denoted as ε, where $\varepsilon = \frac{1}{2}Cor$.

In 1994, Langford and Hellman introduced differential-linear cryptanalysis, combining differential and linear cryptanalysis for the first time. Their idea is to divide the cipher E into two sub-ciphers E_0 and E_1, i.e. $E = E_1 \circ E_0$, where there exists a truncated differential distinguisher $(\Delta_I \to \Delta_O)$ with probability 1 for E_0 and a linear distinguisher $(\lambda_I \to \lambda_O)$ with high bias for E_1. The relationship between the input difference and the output linear mask is then used to distinguish the cipher from the random permutation. Then its bias is defined as follows.

$$\varepsilon = |\, Pr_{x \in \mathbb{F}_2^n}\left(\lambda_O \odot E(x) = \lambda_O \odot E(x \oplus \Delta_I)\right) - \frac{1}{2}| \quad (4)$$

Subsequently, Biham et al. improved the above method by extending the previous differential distinguisher from a probability of 1 to a probability of p, allowing for a wide range of applications of differential-linear cryptanalysis. Here, the correlation of differential-linear distinguisher is $Cor_{\Delta_I,\lambda_O} = pq^2$, where p is the probability of the differential distinguisher and q is the correlation of the linear distinguisher. And the data complexity required is $D = \mathcal{O}(\epsilon p^{-2} q^{-4})$, where $\epsilon \in \mathbb{N}$ is a small constant.

Next, the method of statistical analysis was used to search for a suitable differential-linear distinguisher. During the attack on SIMON and SIMECK specifically, the distinguisher needed to be extended forward and backward, so as few active bits as possible for both the input differential and output masks were needed. The number of active bits was fixed in both the input difference and output mask of the differential-linear distinguisher to one, and then the total number of rounds was chosen to search using Algorithm 1. The method of statistical analysis was used to obtain the average value of $Pr_{x\in\mathbb{F}_2^n}\big(\lambda_O \odot E(x) = \lambda_O \odot E(x \oplus \Delta_I)\big)$, and further obtain the value of bias ε.

With Algorithm 1, a 9-round differential-linear distinguisher (0x0000, 0x0002)→(0x0001, 0x0000) for SIMECK with bias $\varepsilon = 2^{-7.21}$, a 10-round differential-linear distinguisher (0x0000, 0x0002)→(0x0001, 0x0000) for SIMECK with bias $\varepsilon = 2^{-10.03}$ and a 9-round differential-linear distinguisher (0x0000, 0x0008)→(0x0800, 0x0000) for SIMON with bias $\varepsilon = 2^{-8.68}$ were searched.

Algorithm 1 Search for optimal differential-linear distinguishers

```
Input: times of test T, Number of plaintext N
Output: Counter
Counter reset;
for a = 0 : 15 do \\ a: Position of 1 bit in the input difference
  for b = 0 : 15 do \\ b: Position of 1 bit in the output mask
    Counter2 reset;
    for i=0 : T do
      Generate a random key;
      for j = 0 : N do
        Generate random plaintext x;
        if λ_O ⊙ E(x) = λ_O ⊙ E(x ⊕ Δ_I) do
          Counter2 += 1;
        end
      end
    end
    Counter[a][b]←sum(Counter2)/TN-1/2;
  end
end
```

Table 4. Three Differential-Linear Distinguishers

cipher	round	Input difference	output mask	bias
SIMECK32/64	12	0x00020004	0x00018000	$2^{-11.21}$
	13	0x00020004	0x00018000	$2^{-14.03}$
SIMON32/64	12	0x00080020	0x08000200	$2^{-12.69}$

For SIMECK32/64, fixing the output difference as (0x0000, 0x0002), 1-round of difference (0x0002, 0x0004)→(0x0000, 0x0002) with probability 2^{-2} was searched. And fixing the output mask as (0x0001, 0x0000), 2-rounds of linear approximation (0x0001, 0x0000) →(0x0001, 0x8000) with bias 2^{-2} was searched. For SIMON32/64, fixing the output difference as (0x0000, 0x0008), 1-round of difference (0x0008, 0x0020) →(0x0000, 0x0008) with probability 2^{-2} was searched. And fixing the output mask as (0x0800, 0x0000), 2-rounds of linear approximation (0x0800, 0x0000)→(0x0800, 0x0200) with bias 2^{-2} was searched. They were combined with the previously searched

differential-linear distinguishers to give three new differential-linear distinguishers as shown in Table 4.

4. Differential-Linear Attack on SIMECK32/64

4.1. Dynamic key-guessing

The basic idea of dynamic key-guessing based on differential cryptanalysis and linear cryptanalysis is derived from Observation 1 and Observation 2 below.

Observation 1 Let $\Delta x = x \oplus x^*, \Delta y = y \oplus y^*$, then

$$\begin{aligned}(x \cap y) \oplus (x^* \cap y) &= \Delta x \cap y\\ (x \cap y) \oplus (x \cap y^*) &= x \cap \Delta y\\ (x \cap y) \oplus (x^* \cap y^*) &= (x \cap \Delta y) \oplus (\Delta x \cap y) \oplus (\Delta x \cap \Delta y)\end{aligned}$$

According to the round function of SIMECK32/64 and observation 1 the following three equations can be obtained.

$$\begin{aligned}\Delta X^{i+1}_{j+n} = &\left(\Delta X^{i}_{(j+0)\%n+n} \cap X^{i}_{(j+5)\%n+n}\right) \oplus \left(X^{i}_{(j+0)\%n+n} \cap \Delta X^{i}_{(j+5)\%n+n}\right) \oplus\\ &\left(\Delta X^{i}_{(j+0)\%n+n} \cap \Delta X^{i}_{(j+5)\%n+n}\right) \oplus \Delta X^{i}_{(j+1)\%n+n} \oplus \Delta X^{i}_{j}\\ X^{i}_{(j+0)\%n+n} = &\left(X^{i-1}_{(j+0)\%n+n} \cap X^{i-1}_{(j+5)\%n+n}\right) \oplus X^{i-1}_{(j+1)\%n+n} \oplus X^{i-1}_{(j+0)\%n} \oplus K^{i-1}_{(j+0)\%n}\\ X^{i}_{(j+5)\%n+n} = &\left(X^{i-1}_{(j+0)\%n+n} \cap X^{i-1}_{(j+10)\%n+n}\right) \oplus X^{i-1}_{(j+6)\%n+n} \oplus X^{i-1}_{(j+5)\%n} \oplus K^{i-1}_{(j+5)\%n}\end{aligned}$$

Then the key-guessing process is as follows.

(1) When $\left(\Delta X^{i}_{(j+0)\%n+n}, \Delta X^{i}_{(j+5)\%n+n}\right) = (0,0)$ and $\Delta X^{i}_{(j+1)\%n+n} \oplus \Delta X^{i}_{j} \oplus \Delta X^{i+1}_{j+n} = 1$, there is no solution of subkey $K^{i-1}_{(j+0)\%n}$ and $K^{i-1}_{(j+5)\%n}$.

(2) When $\left(\Delta X^{i}_{(j+0)\%n+n}, \Delta X^{i}_{(j+5)\%n+n}\right) = (0,0)$ and $\Delta X^{i}_{(j+1)\%n+n} \oplus \Delta X^{i}_{j} \oplus \Delta X^{i+1}_{j+n} = 0$, there are two solutions for both subkey $K^{i-1}_{(j+0)\%n}$ and $K^{i-1}_{(j+5)\%n}$.

(3) When $\left(\Delta X^{i}_{(j+0)\%n+n}, \Delta X^{i}_{(j+5)\%n+n}\right) = (0,1)$, there is one solution for subkey $K^{i-1}_{(j+0)\%n}$ and two solutions for subkey $K^{i-1}_{(j+5)\%n}$.

(4) When $\left(\Delta X^{i}_{(j+0)\%n+n}, \Delta X^{i}_{(j+5)\%n+n}\right) = (1,0)$, there are two solutions for subkey $K^{i-1}_{(j+0)\%n}$ and one solution for subkey $K^{i-1}_{(j+5)\%n}$.

(5) When $\left(\Delta X^{i}_{(j+0)\%n+n}, \Delta X^{i}_{(j+5)\%n+n}\right) = (1,1)$, there are two solutions for subkey $K^{i-1}_{(j+0)\%n} \oplus K^{i-1}_{(j+5)\%n}$.

Table 5. An Example for Linear Guessing

$x_0 \oplus x_0^{'}$	$x_0 \oplus k_0$	Δy	k_1
0	0	0	0,1
	1	$x_1 \oplus x_1^{'}$	0,1 or no solution
1	0	$x_1^{'} \oplus k_1$	$x_1^{'}$
	1	$x_1 \oplus k_1$	x_1

Observation 2 Let $x, k \in \mathbb{F}_2^2, y = f(x,k) = (x_0 \oplus k_0) \cap (x_1 \oplus k_1) = 0$, and get the value of k. The most directly method is to iterate over the values of k, which requires 2^2 operations. But when one of the two sides of the AND operation is 0, y will be 0 directly and there is no need to know the value of the other side. Using this property it is only need to calculate the value of $x_0 \oplus k_0$ to get the value of k. It is easier to guess the key: when $x_0 \oplus k_0 = 0$, $k_1 = 0,1$; when $x_0 \oplus k_0 = 1$, $k_1 = x_1$.

In differential-linear cryptanalysis, the situation is slightly more complicated and it is required to consider $\Delta y = \left((x_0 \oplus k_0) \cap (x_1 \oplus k_1)\right) \oplus \left(\left(x_0^{'} \oplus k_0\right) \cap \left(x_1^{'} \oplus k_1\right)\right) = 0$. At this point, on the base of Observation 2, the values of $x_0 \oplus x_0^{'}$ and $x_1 \oplus x_1^{'}$ are firstly calculated, and then the value of k

can be obtained by calculating the value of $x_0 \oplus k_0$. The specific guessing process is shown in Table 5.

4.2. Distinguisher extended forward by four rounds

According to the 12-round SIMECK32/64 differential-linear distinguisher the input difference (0x0002, 0x0004) is extended 4 rounds forward, as follows. For AND operations, the output difference is 0 when and only when both input differences are 0, otherwise the output difference is *, where * means either 0 or 1. For XOR operations, when neither input difference is *, it is a normal XOR operation, otherwise the output difference is *. For the r-th round of bit difference $\Delta X_i \neq *$, $\Delta X_i = 0$ or $\Delta X_i = 1$ is said to be a bit difference condition if the input differences of the corresponding AND operations in r-1-th round are not all 0. By extending the distinguisher forward three rounds, a total of 26 bit difference conditions can be obtained. Among them, 8 bit difference conditions in the first round are unrelated to the subkey, leaving 18 bit difference conditions related to the subkey. The plaintext pairs satisfying the conditions are constructed according to the bit difference conditions unrelated to the subkey, and some of the subkey are guessed according to the bit difference conditions related to the subkey. The extended results are shown in Table 6, where the bolded ones represent the bit difference conditions.

Table 6. SIMECK Extended Forward by Four Rounds

round	Input difference/Output mask
0	0, 0, *, *, *, 0, *, *, *, *, *, 1, *, *, *, *, *, 0, 0, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *
1	0, **0**, **0**, *, **0**, **0**, **0**, *, *, *, **0**, **1**, *, *, *, **0**, 0, 0, *, *, *, 0, *, *, *, *, 1, *, *, *, *, *, 0
2	0, 0, **0**, **0**, **0**, 0, **0**, **0**, *, **0**, 0, **0**, **1**, *, **1**, 0, 0, 0, 0, *, 0, 0, 0, *, *, *, 0, 1, *, *, *, 0
3	0, 0, 0, **0**, 0, 0, 0, **0**, **0**, **0**, 0, 0, **0**, **1**, **0**, 0, 0, 0, 0, 0, 0, 0, 0, 0, *, 0, 0, 0, 1, *, 1, 0
4	0, 0, 0, 0, 0, 0, 0, 0, **0**, 0, 0, 0, 0, **0**, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
4-16	12-round SIMECK32/64 differential-linear distinguisher
16	0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

*4.2.1. Data Collection.*With difference conditions at the 6-bit positions X_i^0(i = 0, 16, 17, 21, 26, 31) of the plaintext and difference conditions at the 8-bit positions X_i^1(i = 17, 18, 20, 21, 22, 26, 27, 31) of the output of the first round, since the output difference of the first round is independent of the round key of the first round, the following 8 equations can be established according to the definition of the round function.

$$X_j^1 = \left(X_{(j+0)\%n+n}^0 \cap X_{(j+5)\%n+n}^0\right) \oplus X_{(j+1)\%n+n}^0 \oplus X_{j-n}^0 \tag{5}$$

where j = 17, 18, 20, 21, 22, 26, 27, 31. Fixing the 6-bit positions of the plaintext and the 8-bit positions of the output of the first round, traversing the rest of the plaintext bit positions and solving these equations, a set of plaintexts can be obtained. Thus, the entire plaintext space can be divided into 2^{14} sets with 2^{18} plaintexts in each set. The plaintexts in the same set have the same 6-bit positions in the plaintext and 8-bit positions in the output of the first round, and we will denote these 14 bits as structural parameters, and each set is determined by the structural parameters.

Take the set M_1 and M_2 with 2 different structural parameters X_{26}^0, X_{27}^1 and the remaining 12 with the same structural parameters, and combine all the plaintexts in M_1 and all the plaintexts in M_2 to obtain 2^{36} pairs of plaintexts. Choosing 2^6 such sets gives $2^{36-1+6} = 2^{41}$ pairs of plaintexts, which are then encrypted to give 2^{41} pairs of plaintexts and ciphertexts. The amount of data required in this process is 2^{24} plaintexts, of which about $2^{6+18-1} = 2^{23}$ pairs of plaintexts can be obtained to satisfy the input difference of the differential-linear distinguisher.

4.2.2. Filtering Wrong Plaintext Pairs. For the bit difference condition $\Delta X_{19}^2 = 0$, we can get $\left(\Delta X_{24}^1 \cap X_{19}^1\right) \oplus \left(X_{24}^1 \cap \Delta X_{19}^1\right) \oplus \left(\Delta X_{24}^1 \cap \Delta X_{19}^1\right) \oplus \Delta X_{19}^0 = 0$, when $\left(\Delta X_{24}^1, \Delta X_{19}^1, \Delta X_{19}^0\right) = (0,0,1)$, it has a probability of 1/8, then there is no key solution to the equation, so for the selected

plaintext pairs, the plaintext pairs with such error events can be filtered in advance. Similarly, when $(\Delta X_{23}^1, \Delta X_{19}^1 \oplus \Delta X_{18}^0) = (0,1)$, $(\Delta X_{25}^1, \Delta X_{20}^0) = (0,1)$, and $(\Delta X_{28}^1, \Delta X_{29}^1 \oplus \Delta X_{28}^0) = (0,0)$, the equations have no key solutions, all with probability 1/4. When $(\Delta X_{28}^1, \Delta X_{23}^1, \Delta X_{24}^1 \oplus \Delta X_{23}^0) = (0,0,1)$, $(\Delta X_{30}^1, \Delta X_{25}^1, \Delta X_{25}^0) = (0,0,1)$, and $(\Delta X_{19}^1, \Delta X_{30}^1, \Delta X_{30}^0) = (0,0,0)$, the equations have no key solutions, all with probability 1/8. After filtering these wrong plaintext pairs, approximately $2^{41} \times \left(1-\frac{1}{8}\right)^4 \times \left(1-\frac{1}{4}\right)^3 \approx 2^{38.98}$ plaintext pairs remain.

4.2.3. Computing Candidate Subkeys. A total of 22 bits subkey are included in the 18 bit difference conditions. The solution candidate subkeys are solved by the bit difference conditions.

- For $\Delta X_{22}^2 = 0$, we can get

$$\Delta X_{22}^2 = (X_{22}^1 \cap \Delta X_{27}^1) \oplus \Delta X_{23}^1 \oplus \Delta X_{22}^0,$$
$$X_{22}^1 = (X_{22}^0 \cap X_{27}^0) \oplus X_{23}^0 \oplus X_6^0 \oplus K_6^0,$$
$$\Delta X_{27}^1 = 1.$$

  There is one solution for subkey bit K_6^0 as follows:

$$K_6^0 = (X_{22}^0 \cap X_{27}^0) \oplus X_{23}^0 \oplus X_6^0 \oplus \Delta X_{23}^1 \oplus \Delta X_{22}^0.$$

- Similarly, for $\Delta X_{27}^2 = 0$, there is one solution for subkey bit K_0^0.
- For $\Delta X_{20}^2 = 0$, we can get

$$\Delta X_{20}^2 = (X_{20}^1 \cap \Delta X_{25}^1) \oplus \Delta X_{21}^1 \oplus \Delta X_{20}^0,$$
$$X_{20}^1 = (X_{20}^0 \cap X_{25}^0) \oplus X_{24}^0 \oplus X_4^0 \oplus K_4^0,$$
$$\Delta X_{21}^1 = 0.$$

  The case $(\Delta X_{25}^1, \Delta X_{20}^0) = (0,1)$ does not occur again since the wrong plaintext pair has been filtered in the second round. The solution for subkey bit K_4^0 can be divided into the following two cases: when $(\Delta X_{25}^1, \Delta X_{20}^0) = (0,0)$, the equation $\Delta X_{20}^2 = 0$ is constant and therefore two solutions for subkey bit K_4^0; when $\Delta X_{25}^1 = 1$, one solution exists. For each plaintext and ciphertext pair, an average of $\frac{4}{3}$ solutions of K_4^0 can be obtained.
- Similarly, for $\Delta X_{18}^2 = 0$, an average of $\frac{4}{3}$ K_2^0 solutions can be obtained; for $\Delta X_{28}^2 = 1$, an average of $\frac{4}{3}$ K_1^0 solutions can be obtained.
- For $\Delta X_{19}^2 = 0$, we can get

$$\Delta X_{19}^2 = (X_{19}^1 \cap \Delta X_{24}^1) \oplus (\Delta X_{19}^1 \cap X_{24}^1) \oplus (\Delta X_{19}^1 \cap \Delta X_{24}^1) \oplus \Delta X_{20}^1 \oplus \Delta X_{19}^0 = 0,$$
$$X_{19}^1 = (X_{19}^0 \cap X_{24}^0) \oplus X_{20}^0 \oplus X_3^0 \oplus K_3^0,$$
$$X_{24}^1 = (X_{24}^0 \cap X_{29}^0) \oplus X_{25}^0 \oplus X_8^0 \oplus K_8^0,$$
$$\Delta X_{20}^1 = 0.$$

  For the subkey bits K_3^0, K_8^0, the solutions can be divided into the following four cases: when $(\Delta X_{24}^1, \Delta X_{19}^1, \Delta X_{19}^0) = (0,0,0)$, the subkey bits K_3^0, K_8^0 have two solutions for each; when $(\Delta X_{24}^1, \Delta X_{19}^1) = (0,1)$, the subkey bits K_3^0 have two solutions and K_8^0 has one solution; when $(\Delta X_{24}^1, \Delta X_{19}^1) = (1,0)$, the subkey bit K_8^0 has two solutions and K_3^0 has one solution; when $(\Delta X_{24}^1, \Delta X_{19}^1) = (1,1)$, we can get $K_3^0 \oplus K_8^0$ and $K_3^0 \oplus K_8^0$ has two solutions. For each plaintext and ciphertext pair, an average of $\frac{16}{7}$ (K_3^0, K_8^0) solutions can be obtained.
- Similarly, for $\Delta X_{23}^2 = 0$, an average of $\frac{16}{7}$ (K_7^0, K_{12}^0) solutions can be obtained; for $\Delta X_{25}^2 = 0$, an average of $\frac{16}{7}$ (K_9^0, K_{14}^0) solutions can be obtained; for $\Delta X_{30}^2 = 1$, an average of $\frac{16}{7}$ (K_{14}^0, K_3^0) solutions can be obtained. Since the subkey bit K_3^0 is included in both the two bit difference conditions $\Delta X_{19}^2 = 0$ and $\Delta X_{30}^2 = 1$ in the second round, there exists a $\frac{1}{2}$ probability that the

solution obtained from the two bit difference conditions is contradictory, and K_{14}^{0} is similar.

- Thus, in the second round, the average number of solutions for subkey bits $K_{[0,1,2,3,4,6,7,8,9,12,14]}^{0}$ can be obtained as $\left(\frac{4}{3}\right)^{3} \times \left(\frac{16}{7}\right)^{4} \times 2^{-2}$.
- For the third round of bit difference conditions, the guessing continues on the base of the second round. For $\Delta X_{25}^{3} = 0$, one solution for K_{9}^{1} can be obtained, but a guess is needed for K_{10}^{0}. Similarly, one solution for $K_{[1,3,7]}^{1}$ can be obtained.
- For $\Delta X_{19}^{3} = 0$, when the equation has a solution, on average $\frac{4}{3}$ solutions of K_{3}^{1} can be obtained and the probability of the equation being solved is $\frac{3}{4}$, so on average one solution can be obtained. For $\Delta X_{29}^{3} = 1$, an average of one solution of K_{2}^{1} can be obtained. For $\Delta X_{24}^{3} = 0$, an average of two solutions of K_{8}^{1}, K_{13}^{1} can be obtained. However, the subkey bit K_{3}^{1} is both included in the two bit difference conditions in the third round, with a $\frac{1}{2}$ probability of contradiction.
- Thus, in the third round, the average number of solutions for subkey bits $K_{10}^{0}, K_{13}^{0}, K_{[1,2,3,7,8,9,13]}^{1}$ can be obtained as 2^{2}.
- For the fourth round of the bit difference condition, one solution of K_{2}^{2}, K_{8}^{2} can be obtained on the base of the second and third rounds.
- By solving these 18 bit difference conditions in turn, for each plaintext pair, with respect to the subkey bits $K_{[0,1,2,3,4,6,7,8,9,10,12,13,14]}^{0} K_{[1,2,3,7,8,9,13]}^{1} K_{2}^{2}, K_{8}^{2}$ can obtain on average $\left(\frac{4}{3}\right)^{3} \times \left(\frac{16}{7}\right)^{4} \approx 2^{6.02}$ solutions.

4.3. Distinguisher extended backward by four rounds

According to the output mask (0x0001, 0x8000) of the 12-round SIMECK32/64 differential-linear distinguisher expanded backward by four rounds.

The condition needed to be satisfied is $\Delta X_{0}^{16} \oplus \Delta X_{31}^{16} = 0$. For the simplicity of the analysis, introduce $x_i, k_i (i \in [0,9])$, the exact values of which are given in Table 7. Let $f = \mathrm{X}_{0}^{16} \oplus \mathrm{X}_{31}^{16}$, then we can get

$$f = \Big(\big(((x_1 \oplus k_1) \cap (x_2 \oplus k_2)) \oplus x_3 \oplus k_3 \oplus x_8 \oplus k_8\big) \cap \big(((x_2 \oplus k_2) \cap (x_4 \oplus k_4)) \oplus x_5 \oplus k_5 \oplus x_9 \oplus k_9\big)\Big) \oplus \big((x_3 \oplus k_3) \cap (x_5 \oplus k_5)\big) \oplus \big((x_6 \oplus k_6) \cap (x_7 \oplus k_7)\big) \oplus x_0 \oplus k_0.$$

Table 7. Variable in The Process of SIMECK Extended Backward

x_0	$\left(\mathrm{X}_{2}^{19} \cap \mathrm{X}_{7}^{19}\right) \oplus \mathrm{X}_{3}^{19} \oplus \mathrm{X}_{18}^{19} \oplus \mathrm{X}_{1}^{19} \oplus \mathrm{X}_{15}^{19}$	k_0	$\mathrm{K}_{2}^{18} \oplus \mathrm{K}_{1}^{17} \oplus \mathrm{K}_{15}^{17} \oplus \mathrm{K}_{0}^{16}$
x_1	$\left(\mathrm{X}_{0}^{19} \cap \mathrm{X}_{5}^{19}\right) \oplus \mathrm{X}_{1}^{19} \oplus \mathrm{X}_{16}^{19}$	k_1	K_{0}^{18}
x_2	$\left(\mathrm{X}_{5}^{19} \cap \mathrm{X}_{10}^{19}\right) \oplus \mathrm{X}_{6}^{19} \oplus \mathrm{X}_{21}^{19}$	k_2	K_{5}^{18}
x_3	$\left(\mathrm{X}_{1}^{19} \cap \mathrm{X}_{6}^{19}\right) \oplus \mathrm{X}_{2}^{19} \oplus \mathrm{X}_{17}^{19}$	k_3	K_{1}^{18}
x_4	$\left(\mathrm{X}_{10}^{19} \cap \mathrm{X}_{15}^{19}\right) \oplus \mathrm{X}_{11}^{19} \oplus \mathrm{X}_{26}^{19}$	k_4	K_{10}^{18}
x_5	$\left(\mathrm{X}_{6}^{19} \cap \mathrm{X}_{11}^{19}\right) \oplus \mathrm{X}_{7}^{19} \oplus \mathrm{X}_{22}^{19}$	k_5	K_{6}^{18}
x_6	$\left(\mathrm{X}_{15}^{19} \cap \mathrm{X}_{4}^{19}\right) \oplus \mathrm{X}_{0}^{19} \oplus \mathrm{X}_{31}^{19}$	k_6	K_{15}^{18}
x_7	$\left(\mathrm{X}_{4}^{19} \cap \mathrm{X}_{9}^{19}\right) \oplus \mathrm{X}_{5}^{19} \oplus \mathrm{X}_{20}^{19}$	k_7	K_{4}^{18}
x_8	X_{0}^{19}	k_8	K_{0}^{17}
x_9	X_{5}^{19}	k_9	K_{5}^{17}

We only need to guess the 5-bit key for k_1, k_2, k_3, k_6, k_8.

- Guess k_1, k_2.

Table 8. Guess k_1, k_2

Guess	$\mathrm{x}_1 \oplus \mathrm{k}_1, \mathrm{x}_2 \oplus \mathrm{k}_2$	f

k_1, k_2	0,0	f_0
	0,1	f_1
	1,0	f_0
	1,1	f_3

The representation of f_i as follows,

$$f_0=\left((x_3\oplus k_3\oplus x_8\oplus k_8)\cap(x_5\oplus k_5\oplus x_9\oplus k_9)\right)\oplus\left((x_3\oplus k_3)\cap(x_5\oplus k_5)\right)\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

$$f_1=\left((x_3\oplus k_3\oplus x_8\oplus k_8)\cap(x_4\oplus k_4\oplus x_5\oplus k_5\oplus x_9\oplus k_9)\right)\oplus\left((x_3\oplus k_3)\cap(x_5\oplus k_5)\right)\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

$$f_3=\left((1\oplus x_3\oplus k_3\oplus x_8\oplus k_8)\cap(x_4\oplus k_4\oplus x_5\oplus k_5\oplus x_9\oplus k_9)\right)\oplus\left((x_3\oplus k_3)\cap(x_5\oplus k_5)\right)\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

- Continue to guess k_3, k_8 with the example of f_0.

Table 9. Guess k_3, k_8

Guess	$x_3\oplus k_3, x_8\oplus k_8$	f_0
k_3, k_8	0,0	f_{00}
	0,1	f_{01}
	1,0	f_{02}
	1,1	f_{03}

The representation of f_{ij} as follows,

$$f_{00}=\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

$$f_{01}=x_5\oplus k_5\oplus x_9\oplus k_9\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

$$f_{02}=x_5\oplus k_5\oplus x_9\oplus k_9\oplus x_5\oplus k_5\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

$$f_{03}=x_5\oplus k_5\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus x_0\oplus k_0$$

- Final guess k_6 with the example of f_{00}. When $x_6\oplus k_6=0$, $f_{000}=x_0\oplus k_0$. When $x_6\oplus k_6=1$, $f_{001}=x_7\oplus k_7\oplus x_0\oplus k_0$.

The condition to be satisfied is $\Delta f=0$ and the expression is as follows.

$$\Delta f=\left(\left(\left((x_1\oplus k_1)\cap(x_2\oplus k_2)\right)\oplus x_3\oplus k_3\oplus x_8\oplus k_8\right)\cap\left(\left((x_2\oplus k_2)\cap(x_4\oplus k_4)\right)\oplus x_5\oplus k_5\oplus x_9\oplus k_9\right)\right)\oplus\left((x_3\oplus k_3)\cap(x_5\oplus k_5)\right)\oplus\left((x_6\oplus k_6)\cap(x_7\oplus k_7)\right)\oplus\left(\left(\left((x_1'\oplus k_1)\cap(x_2'\oplus k_2)\right)\oplus x_3'\oplus k_3\oplus x_8'\oplus k_8\right)\cap\left(\left((x_2'\oplus k_2)\cap(x_4'\oplus k_4)\right)\oplus x_5'\oplus k_5\oplus x_9'\oplus k_9\right)\right)\oplus\left((x_3'\oplus k_3)\cap(x_5'\oplus k_5)\right)\oplus\left((x_6'\oplus k_6)\cap(x_7'\oplus k_7)\right)\oplus\Delta x_0$$

At this point, $u_i=x_i\oplus x_i'(i=1,2,3,6,8)$ is first computed, for a total of 2^5 possible cases. Then for each case, the key-guessing process described above is carried out. For example, when $u_1=1$, if $x_1\oplus k_1=0$, then $x_1'\oplus k_1=1$, essentially still carrying out one round of the above key-guessing process.

Then it can be extended one more round backward, and in order to get $x_i(i\in[0,9])$ in the Table 7, the 12 subkey bits $K^{19}_{[0-7,9,10,11,15]}$ need to be guessed, where the amount of key to be guessed is 17 bits.

During the backward expansion of 4 rounds, for each plaintext and ciphertext pairs, an average of 2^8 solutions of the subkey can be obtained.

4.4. 20 and 21 rounds of SIMECK32/64 key recovery

We extended the 12-round SIMECK32/64 differential-linear distinguisher (0x0002, 0x0004) → (0x0001, 0x8000) with a bias of $2^{-11.21}$ by 4 rounds forward and 4 rounds backward to make key recovery for 20 rounds of SIMECK32/64, and gave the pseudo-code for the key recovery process as shown in Algorithm 2.

During the attack 22+21=43 bits subkey are involved and 22+12=34 bits subkey need to be guessed. The amount of data required in the whole process is 2^{24}. The time complexity is calculated as follows. After filtering the wrong pairs, there are $2^{38.98}$ plaintext pairs, which require $2^{39.98}$ full rounds of encryption, and then decrypting them one round, which requires $2^{39.98} \times 1/20 \approx 2^{35.66}$ full rounds of encryption. In total, $2^{39.98} + 2^{35.66} \approx 2^{40.05}$ full rounds of encryption are required. The counter update phase gives an average of $2^{6.02+8}$ subkey solutions at a time, so a total of $2^{38.98+6.02+8} = 2^{53}$ addition operations are required. According to the paper [3], considering an addition operation as 1/16 round of encryption, the $2^{53}/(16 \times 20) \approx 2^{44.68}$ full rounds of encryption is required. The remaining 39 linearly independant subkeys exhaustively require 2^{39} full rounds of encryption. The total time complexity is therefore approximately $2^{40.05} + 2^{39} + 2^{44.68} \approx 2^{44.76}$.

And then key recovery was performed for 21 rounds of SIMECK32/64. A 13-round SIMECK32/64 differential-linear distinguisher (0x0002, 0x0004) → (0x0001, 0x8000) with a bias of $2^{-14.03}$ is extended forward by 4 rounds and backward by 4 rounds to form a 21-round encryption for key recovery.

Take the set M_1 and M_2 with 2 different structural parameters X_{26}^0, X_{27}^1 and the remaining 12 with the same structural parameters, and combine all the plaintexts in M_1 and all the plaintexts in M_2 to obtain 2^{36} pairs of plaintexts. Choosing 2^{12} such sets gives $2^{36-1+12} = 2^{47}$ pairs of plaintexts, which are then encrypted to give 2^{47} pairs of plaintexts and ciphertexts. The amount of data required in this process is 2^{30} plaintexts, of which about $2^{12+18-1} = 2^{29}$ pairs of plaintexts can be obtained to satisfy the input difference of the differential-linear distinguisher.

Algorithm 2 20 rounds of SIMECK32/64 key recovery

Input: Master key, Round 0 and 1 bit conditions position d_0,d_1
Output: Counter
// plaintext P_1, P_2, Round variables X_1, X_2, Input difference per round dX
Counter reset.
for i = 0 : 2^6-1 **do**
  P_1, P_2 ←Solve(d_0,d_1)
  for j = 0 : 2^{18}-1 **do**
    for w = 0 : 2^{18}-1 **do**
      $X_1[0], X_2[0] \leftarrow P_1[j], P_2[w]$
      $dX[0] \leftarrow XOR(X_1[0], X_2[0])$
      if $dX[0]$ is not failed **then**
        res_1 ←Guess the key of the distinguisher forward extension
        $X_1[20], X_2[20] \leftarrow Encrypt(X_1[0], X_2[0])$
        res_2 ←Guess the key of the distinguisher backward extension
        Counter ← res_1, res_2
      end
    end
  end
end

During the attack 43 bits subkey are involved and 34 bits subkey need to be guessed. The amount of data required in the whole process is 2^{30}. The time complexity is calculated as follows. After filtering the wrong pairs, there are $2^{44.98}$ plaintext pairs, which require $2^{45.98}$ full rounds of encryption, and then decrypting them one round, which requires $2^{39.98} \times 1/20 \approx 2^{41.66}$ full rounds

of encryption. In total, $2^{45.98}+2^{41.66}\approx 2^{46.05}$ full rounds of encryption are required. The counter update phase gives an average of $2^{6.02+8}$ subkey solutions at a time, so a total of $2^{44.98+6.02+8}=2^{59}$ addition operations are required, where requires $2^{59}/(16\times 21)\approx 2^{50.61}$ full rounds of encryption. The remaining 39 linearly independant subkeys exhaustively require 2^{39} full rounds of encryption. The total time complexity is therefore approximately $2^{46.05}+2^{39}+2^{50.61}\approx 2^{50.67}$.

5. Differential-Linear Attack on SIMON32/64

Extending the 12-round SIMON32/64 differential-linear distinguisher (0x0008, 0x0020)→(0x0800, 0x0200) with a bias of $2^{-12.69}$ forward by four rounds, as shown in Table 10.

Table 10. SIMON Extended Forward by Four Rounds

round	Input difference/Output mask
0	*, 0, *, *, 1, *, *, *, *, *, *, 0, *, 0, *
1	*, *, **0**, **0**, *, **0**, **1**, *, *, **0**, *, *, *, **0**, **0**, **0**, **0**, *, 0, *, *, 1, *, *, *, *, *, *, 0, *, 0, *, *
2	**0**, 0, *, **0**, 0, **0**, **0**, **0**, **1**, *, **0**, 0, **1**, 0, **0**, **0**, *, *, 0, 0, *, 0, 1, *, *, 0, *, *, 0, 0, 0, 0
3	**0**, **0**, 0, 0, **0**, 0, 0, **0**, **0**, 0, **1**, **0**, 0, 0, 0, 0, 0, 0, *, 0, 0, 0, 0, 0, 1, *, 0, 0, 1, 0, 0, 0
4	0, 0, **0**, 0, 0, 0, 0, 0, 0, **0**, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
4-16	12-round SIMON32/64 differential-linear distinguisher
16	0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0

There are bit difference conditions at 4 positions in the plaintext and 9 positions in the first round, so the entire plaintext space can be divided into 2^{13} sets, each with 2^{19} plaintexts.

Take the set M_1 and M_2 with 2 different structural parameters X_{20}^0, X_{22}^1 and the remaining 11 with the same structural parameters, and combine all the plaintexts in M_1 and all the plaintexts in M_2 to obtain 2^{38} pairs of plaintexts. Choosing 2^9 such sets gives $2^{38-1+9}=2^{46}$ pairs of plaintexts, which are then encrypted to give 2^{46} pairs of plaintexts and ciphertexts. The amount of data required in this process is 2^{28} plaintexts, of which about $2^{9+19-1}=2^{27}$ pairs of plaintexts can be obtained to satisfy the input difference of the differential-linear distinguisher.

There are a total of 19 bit difference conditions associated with subkeys, including 27 bit subkeys, which are K_i^0(i=0-15), K_i^1(i=0,1,2,3,4,5,9,11,15), and K_i^2(i=1,3). Filtering a portion of the plaintext pairs according to the second round of bit difference conditions leaves $2^{46}\times\left(1-\frac{1}{8}\right)^3\times\left(1-\frac{1}{4}\right)^5\approx 2^{43.35}$ pairs of plaintext pairs.

According to the output mask (0x0800, 0x0200) of the 12-round SIMECK32/64 differential-linear distinguisher expanded backward by three rounds. Let $f=\mathrm{X}_6^{16}\oplus\mathrm{X}_{20}^{16}$ with the following expression, the specific values of each variable are given in Table 11.

$$f=\Big(\big((x_1\oplus k_1)\cap(x_2\oplus k_2)\big)\oplus x_3\oplus k_3\oplus x_8\oplus k_8\Big)\cap\Big(\big((x_2\oplus k_2)\cap(x_4\oplus k_4)\big)\oplus x_5\oplus k_5\oplus x_9\oplus k_9\big)\oplus\big((x_3\oplus k_3)\cap(x_5\oplus k_5)\big)\oplus\big((x_6\oplus k_6)\cap(x_7\oplus k_7)\big)\oplus x_0\oplus k_0.$$

Table 11. Variable in The Process of SIMECK Extended Backward

x_0	$\left(\mathrm{X}_{11}^{19}\cap\mathrm{X}_2^{19}\right)\oplus\mathrm{X}_{12}^{19}\oplus\mathrm{X}_{26}^{19}\oplus\mathrm{X}_8^{19}\oplus\mathrm{X}_4^{19}$	k_0	$\mathrm{K}_{10}^{18}\oplus\mathrm{K}_8^{17}\oplus\mathrm{K}_4^{17}\oplus\mathrm{K}_6^{16}$
x_1	$\left(\mathrm{X}_9^{19}\cap\mathrm{X}_0^{19}\right)\oplus\mathrm{X}_{10}^{19}\oplus\mathrm{X}_{24}^{19}$	k_1	K_8^{18}
x_2	$\left(\mathrm{X}_0^{19}\cap\mathrm{X}_7^{19}\right)\oplus\mathrm{X}_1^{19}\oplus\mathrm{X}_{31}^{19}$	k_2	K_{15}^{18}
x_3	$\left(\mathrm{X}_{10}^{19}\cap\mathrm{X}_1^{19}\right)\oplus\mathrm{X}_{11}^{19}\oplus\mathrm{X}_{25}^{19}$	k_3	K_9^{18}
x_4	$\left(\mathrm{X}_7^{19}\cap\mathrm{X}_{14}^{19}\right)\oplus\mathrm{X}_8^{19}\oplus\mathrm{X}_{22}^{19}$	k_4	K_6^{18}
x_5	$\left(\mathrm{X}_1^{19}\cap\mathrm{X}_8^{19}\right)\oplus\mathrm{X}_2^{19}\oplus\mathrm{X}_{16}^{19}$	k_5	K_0^{18}
x_6	$\left(\mathrm{X}_6^{19}\cap\mathrm{X}_{13}^{19}\right)\oplus\mathrm{X}_7^{19}\oplus\mathrm{X}_{21}^{19}$	k_6	K_5^{18}
x_7	$\left(\mathrm{X}_{13}^{19}\cap\mathrm{X}_4^{19}\right)\oplus\mathrm{X}_{14}^{19}\oplus\mathrm{X}_{28}^{19}$	k_7	K_{12}^{18}
x_8	X_7^{19}	k_8	K_7^{17}
x_9	X_{14}^{19}	k_9	K_{14}^{17}

The same expressions can be found as in the previous analysis of SIMECK32/64, so only the 5-bit keys k_1, k_2, k_3, k_6, k_8 need to be guessed.

Then it can be extended one more round backward, and in order to get $x_i (i \in [0,9])$ in the Table 11, the 13 bits subkey $K^{19}_{[0,1,2,4,6-14]}$ need to be guessed, where the amount of key to be guessed is 18 bits. During the backward expansion of 4 rounds, for each plaintext and ciphertext pair, an average of 2^8 solutions of the subkey can be obtained.

During the attack 27+22=49 bits subkey are involved and 27+18=45 bits subkey need to be guessed. The amount of data required in the whole process is 2^{28}. The time complexity is calculated as follows. After filtering the wrong pairs, there are $2^{43.35}$ plaintext pairs, which require $2^{44.35}$ full rounds of encryption, and then decrypting them one round, which requires $2^{44.35} \times 1/20 \approx 2^{40.03}$ full rounds of encryption. In total, $2^{44.35} + 2^{40.03} \approx 2^{44.42}$ full rounds of encryption are required. The counter update phase gives an average of $2^{14.65+8}$ subkey solutions at a time, so a total of $2^{43.35+14.65+8} = 2^{64}$ addition operations are required, where requires $2^{64}/(16 \times 20) \approx 2^{55.68}$ full rounds of encryption. The remaining 15 linearly independant subkeys exhaustively require 2^{15} full rounds of encryption. The total time complexity is therefore approximately $2^{44.42} + 2^{15} + 2^{55.68} \approx 2^{55.68}$.

6. Conclusions

In this paper, we gave a 13-round SIMECK32/64 differential-linear distinguisher and a 12-round SIMON32/64 differential-linear distinguisher. Then we first applied dynamic key-guessing techniques to the key recovery process of the differential-linear cryptanalysis. We attacked the 20-round SIMON32/64 based on the 12-round SIMON32/64 differential-linear distinguisher and the 21-round SIMECK32/64 based on the 13-round SIMECK32/64 differential-linear distinguisher. The dynamic key guessing technique is currently only used in SIMON-like ciphers, which can significantly increase the number of attack rounds. Future work is differential-linear cryptanalysis of other versions of SIMON and SIMECK.

References

[1] Beaulieu R, Shors D, Smith J, et al 2013 The SIMON and SPECK families of lightweight block ciphers cryptology eprint archive
[2] Yang G, Zhu B, Suder V, et al 2015 The simeck family of lightweight block ciphers International Workshop on Cryptographic Hardware and Embedded Systems (Heidelberg: Springer) p 307-329
[3] Wang N, Wang X, Jia K, et al 2014 Differential attacks on reduced SIMON versions with dynamic key-guessing techniques Cryptology ePrint Archive
[4] Chen H and Wang X 2016 Improved linear hull attack on round-reduced Simon with dynamic key-guessing techniques International Conference on Fast Software Encryption (Heidelberg: Springer) p 428-449.
[5] Chen Z, Wang N and Wang X 2015 Impossible differential cryptanalysis of reduced round SIMON Cryptology ePrint Archive
[6] Chu Z, Chen H, Wang X, et al 2018 Improved integral attacks on SIMON32 and SIMON48 with dynamic key-guessing techniques Security and Communication Networks
[7] Qiao K, Hu L and Sun S 2016 Differential analysis on simeck and simon with dynamic key-guessing techniques International Conference on Information Systems Security and Privacy (Cham: Springer) p 64-85
[8] Qin L, Chen H and Wang X 2016 Linear hull attack on round-reduced simeck with dynamic key-guessing techniques Australasian Conference on Information Security and Privacy (Cham: Springer) p 409-424
[9] Chen Y and Zhang W 2018 Differential-linear cryptanalysis of SIMON32/64 International Journal of Embedded Systems vol 10(3) p 196-202

[10] Sun L, Fu K and Wang M 2015 Improved zero-correlation cryptanalysis on SIMON International Conference on Information Security and Cryptology (Cham: Springer) p 125-143
[11] Kölbl S and Roy A 2016 A brief comparison of Simon and Simeck International Workshop on Lightweight Cryptography for Security and Privacy (Cham: Springer) p 69-88
[12] Zhang K, Guan J, Hu B, et al 2018 Security evaluation on Simeck against zero‐correlation linear cryptanalysis IET Information Security vol 12(1) p 87-93
[13] Langford S K and Hellman M E 1994 Differential-linear cryptanalysis Annual International Cryptology Conference (Heidelberg: Springer) p 17-25
[14] Biham E, Dunkelman O and Keller N 2002 Enhancing differential-linear cryptanalysis International Conference on the Theory and Application of Cryptology and Information Security (Heidelberg: Springer) p 254-266
[15] Hu Y, Dai Z and Sun B 2022 Differential-Linear Cryptanalysis of the SIMON Algorithm Netinfo Security 22(9) p 63-75