Differential Security Evaluation of Simeck with Dynamic Key-guessing Techniques

Kexin Qiao[1,2], Lei Hu[1,2], Siwei Sun[1,2]

[1]State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences,
Beijing 100093, China
[2]Data Assurance and Communication Security Research Center, Chinese Academy of Sciences,
Beijing 100093, China
{qiaokexin,hulei,sunsiwei}@iie.ac.cn

Abstract. The Simeck family of lightweight block ciphers was proposed in CHES 2015 which combines the good design components from NSA designed ciphers SIMON and SPECK. Dynamic key-guessing techniques were proposed by Wang *et al.* to greatly reduce the key space guessed in differential cryptanalysis and work well on SIMON. In this paper, we implement the dynamic key-guessing techniques in a program to automatically give out the data in dynamic key-guessing procedure and thus simplify the security evaluation of SIMON and Simeck like block ciphers regarding differential attacks. We use the differentials from Kölbl *et al.*'s work and also a differential with lower Hamming weight we find using Mixed Integer Linear Programming method to attack Simeck and improve the previously best results on all versions of Simeck by 2 rounds.

Keywords: Simeck, Dynamic Key-guessing, Differential Cryptanalysis

1 Introduction

SIMON and SPECK [17] are two lightweight block cipher families designed by NSA that have attracted numerous cryptanalysis since their publication in 2013 [1,4,7,11,12,15,16,20]. SIMON is optimized for hardware implementation and SPECK is optimized for software. In CHES 2015, Yang*et al.* combine their good components and get a new design of block cipher family, called Simeck [10]. The Simeck family applies a slightly modified version of SIMON's round function and reuses it in the key schedule like SPECK does. The hardware implementations of Simeck block cipher family are even smaller than that of SIMON in terms of area and power consumption in [10].

In [15], a new differential attack applying dynamic key-guessing techniques was proposed to work on the reduced SIMON family. The basic idea of the attack is to merge the classic differential attack [6] and the modular differential attack which is widely used to attack hash functions [3,8,9,22,23]. This technique is aimed at block ciphers with bitwise AND operator. Based on observations of differential propagation of the AND operator, attackers can deduce values of some subkey bits and thus greatly reduce the key space that need to be guessed. With differentials with high probability in previous papers [1,7,19], dynamic key-guessing techniques were used to improved the best previous cryptanalysis results by 2 to 4 rounds on family of SIMON block cipher in [15].

As dynamic key-guessing techniques were newly proposed, the designers of Simeck did not consider its security regarding this technique. The designers of Simeck give some other security analysis results including differential attacks [6], linear attacks [14], impossible differential attacks [5], *etc.* mainly following the attack procedure of SIMON due to their similarity. In [2] and [21], cryptanalysis covering more rounds are given. In [21], the authors give differentials with high probability of all three versions and launch differential attacks covering 19, 26 and 33 rounds of Simeck32/64, Simeck48/96 and Simeck64/128 respectively. Though authors of [21] noticed the dynamic key-guessing method but they did not implement it.

In this paper, we reveal some details in implementing the dynamic key-guessing techniques and thus make it easy to launch a differential attack with these techniques on SIMON and Simeck like block ciphers. Specifically, we write a program to calculate the complexity in dynamic key-guessing procedure and then estimate the complexities in differential cryptanalysis on family of Simeck block ciphers. We find a 13-round differential of Simeck32/64 with lower hamming weight with probability $2^{-29.64}$. Applying this differential and differentials from [21] to attack Simeck with dynamic key-guessing techniques, we improve the best previous results on all versions of Simeck block ciphers by 2 rounds. The comparison of the cryptanalysis results for Simeck is in Table 1.

The organization of the paper is as follows. In Section 2 we give a brief introduction of the Simeck family block cipher. In Section 3 we describe Wang *et al.*'s dynamic key-guessing techniques in a general way and provide some details in implementing the techniques. In Section 4 we give a 13-round differential of Simeck32/64 found by MILP method and use it as well as differentials in references to launch differential attack with dynamic key-guessing techniques on Simeck. We conclude the paper in Section 5.

Table 1: Comparison of Cryptanalysis Results of Simeck

Versions	Total Rounds	Attacked Rounds	Time Complexity	Data Complexity	Success Prob.	Reference
Simeck32/64	32	18	$2^{63.5}$	2^{31}	47.7%	[2]
		19	2^{36}	2^{31}	-	[21]
		20	$2^{62.6}$	2^{32}	-	[10]
		21	$2^{48.5}$	2^{30}	41.7%	This paper
		22	$2^{57.9}$	2^{32}	47.1%	This paper
Simeck48/96	36	24	2^{94}	2^{45}	47.7%	[2]
		24	$2^{94.7}$	2^{48}	-	[10]
		26	2^{62}	2^{47}	-	[21]
		28	$2^{68.3}$	2^{46}	46.8%	This paper
Simeck64/128	44	25	$2^{126.6}$	2^{64}	-	[10]
		27	$2^{120.5}$	2^{61}	47.7%	[2]
		33	2^{96}	2^{63}	-	[21]
		34	$2^{116.3}$	2^{63}	55.5%	This paper
		35	$2^{116.3}$	2^{63}	55.5%	This paper

2 The Simeck Lightweight Block Cipher

2.1 Notations

In this paper, we use notations as follows.

X^{r-1}	input of the r-th round
L^{r-1}	left half of X_{r-1}
R^{r-1}	right half of X_{r-1}
K^{r-1}	subkey used in r-th round
X_i	i-th bit of X, indexed from left to right
$X \ggg r$	right rotation of X by r bits
$\oplus$	bitwise exclusive OR (XOR)
$\wedge$	bitwise AND
ΔX	$X \oplus X'$
$+$	addition operation
$\%$	modular operation
$\cup$	union of sets
$\cap$	intersection of sets

2.2 Description of Simeck

The Simeck lightweight block cipher was introduced in [10]. It is a Feistel structure and is denoted by Simeck$2n/mn$, where $2n$ is the block size and mn the master key size. It includes three versions: Simeck32/64, Simeck48/96 and Simeck64/128 with number of rounds n_r=32, 36 and 44 respectively. The left half of input texts to i-th round is $L^{i-1} = \{X_n^{i-1}, X_{n+1}^{i-1}, \cdots, X_{2n-1}^{i-1}\}$ and the right half is $R^{i-1} = \{X_0^{i-1}, X_1^{i-1}, \cdots, X_{n-1}^{i-1}\}$ and the subkey is $K^{i-1} = \{K_0^{i-1}, K_1^{i-1}, \cdots, K_{n-1}^{i-1}\}$. The round function of Simeck is

$$(L^i, R^i) = (R^{i-1} \oplus F(L^{i-1}) \oplus K^{i-1}, L^{i-1})$$

where

$$F(x) = (x \wedge (x \lll 5)) \oplus (x \lll 1)$$

for $i = 1, \cdots n_r$. It can be seen that the round function of Simeck is similar to that of SIMON. For coherence, we denote the rotation offsets by a, b and c. In Simeck, $a = 0, b = 5, c = 1$ and in SIMON $a = 1, b = 8, c = 2$.

The structure of the key schedule of Simeck is similar to that of SPECK while the update function reuses the round function of Simeck with constants acting as round key. We refer the readers to [10] for details of Simeck.

3 Evaluating Security Regarding Differential Attack with Dynamic Key-guessing Techniques

Differential attack [6] is one of the most powerful attacks on iterative block ciphers. If there is an input difference that results in an output difference with high probability against a reduced-round block cipher (called a differential), by adding extra rounds before and after the differential, an attacker can choose and encrypt an amount of plaintext pairs that may satisfy the input difference, and then guess the subkey bits in the added rounds that influence the differential. Right guess will lead conspicuous number of plaintext and ciphertext pairs to the differential.

In [15], Wang *et al.* proposed dynamic key-guessing techniques to greatly reduce the number of secret key bits that need to be guessed in differential cryptanalysis. These techniques were based on observations that some subkey bits can be deduced from equations invoked by certain input differences of AND operator. Different input differences of AND operator result in different conditions of subkey bits involved in the equations. Before using these observations, attackers should find out the sufficient bit conditions that act as equations in the extended rounds to make the differential hold. Thus the preprocessing phase of differential cryptanalysis with dynamic key-guessing techniques is divided into two stages when a differential with high probability of the cipher has been found: firstly, generate the extended path and identify the sufficient bit conditions to be processed and secondly generate the related subkey bits corresponding to each bit condition in the first stage. In the following we illustrate the differential attacks with dynamic key-guessing techniques in a general way and reveal some details of the implementation of the technique. We refer the readers to [15] for some principles of the technique.

3.1 Generate the Extended Path with Sufficient Bit Conditions

Suppose a differential with probability p covering R rounds has been found, we prefix r_0 rounds on the top and append r_1 rounds at the bottom. To get the differential path of the prefixed r_0 rounds, "decrypt" the input difference of the differential according to the rules that the output differences of AND operator is 0 if and only if its input differences are $(0,0)$. Otherwise set the output difference of AND operator to $*$. For the appended r_1 rounds, "encrypt" the output difference of the differential according to the same rules.

The bit conditions to be processed in the extended path are sufficient bit-difference conditions to make the differential path hold. Specifically, when the input differences of AND operator are not $(0,0)$ and its output difference is definite (0 or 1, not $*$), then this output difference is a sufficient bit condition. Note that the prefixed r_0 rounds should be processed in encryption direction to lable sufficient bit conditions and the appended r_1 rounds should be processed in decryption direction.

3.2 Data Collection

Suppose there are l_0 conditions in the plaintext differences and l_1 sufficient bit conditions in ΔX^1. Divide the plaintexts into $2^{l_0+l_1}$ structures with $2^{2n-l_0-l_1}$ plaintexts in each structure.

For two structures with different bits in positions with difference 1 in the above (l_0+l_1) bits, save the corresponding ciphertexts into a table indexed by ciphertext bits in positions with difference 0. Suppose there are l_2 ciphertext bits are with difference 0, then for each such structure pair, there are about $2^{2(2n-l_0-l_1)-l_2}$ plaintext pairs remaining.

We build 2^t plaintext structures, and filter out the remaining pairs by decrypting one round. Suppose there are another k bit conditions to be satisfied in $\Delta X^{r_0+R+r_1-1}$ after one round decryption of the ciphertexts, then there are $2^{t-1+2(2n-l_0-l_1)-l_2-k}$ pairs left. Store them in a table T. At the same time, we expect to get $\lambda_r = 2^{t-1+2n-l_0-l_1} \cdot p$ right pairs.

The plaintext pairs in the table T can still be filtered by bit conditions in ΔX^2 and $\Delta X^{r_0+R+r_1-2}$ as some plaintext pairs may result in no subkey bit solution to equations regarding sufficient bit conditions in ΔX^2 and $\Delta X^{r_0+R+r_1-2}$. The procedure of generating subkey bits related to each sufficient bit condition is described in next subsection.

3.3 Generate Related Subkey Bits for Each Sufficient Bit Condition

For each sufficient bit condition, we get two kinds of subkey bits related to this bit - the subkey bits as variables of equations and subkey bits that need to be guessed to get the specific equation. In encryption direction, we have an equation for sufficient bit condition $\Delta X^i_{j+n} = 0$ or 1 where $j \in [0, n-1]$ and

$$\begin{aligned} \Delta X^i_{j+n} =& \Delta X^{i-1}_{(j+a)\%n+n} \wedge X^{i-1}_{(j+b)\%n+n} \oplus \Delta X^{i-1}_{(j+b)\%n+n} \wedge X^{i-1}_{(j+a)\%n+n} \\ & \oplus \Delta X^{i-1}_{(j+a)\%n+n} \wedge \Delta X^{i-1}_{(j+b)\%n+n} \oplus \Delta X^{i-1}_{(j+c)\%n+n} \oplus \Delta X^{i-2}_{j+n}, \end{aligned} \tag{1}$$

where

$$\begin{aligned} X^{i-1}_{(j+b)\%n+n} =& X^{i-2}_{(j+b+a)\%n+n} \wedge X^{i-2}_{(j+b+b)\%n+n} \\ & \oplus X^{i-2}_{(j+b+c)\%n+n} \oplus X^{i-2}_{(j+b)\%n} \oplus K^{i-2}_{(j+b)\%n}, \\ X^{i-1}_{(j+a)\%n+n} =& X^{i-2}_{(j+a+a)\%n+n} \wedge X^{i-2}_{(j+a+b)\%n+n} \\ & \oplus X^{i-2}_{(j+a+c)\%n+n} \oplus X^{i-2}_{(j+a)\%n} \oplus K^{i-2}_{(j+a)\%n}. \end{aligned} \tag{2}$$

When $(\Delta X^{i-1}_{(j+a)\%n+n}, \Delta X^{i-1}_{(j+b)\%n+n}) = (0, 0)$ and $\Delta X^{i-1}_{(j+c)\%n+n} \oplus \Delta X^{i-2}_{j+n} \neq \Delta X^i_{j+n}$, it is an invalid equation and we get no subkey bit solution. Therefore, for sufficient bit conditions in ΔX_2 and $\Delta X^{r_0+R+r_1-2}$, this property can be used to filter out the wrong plaintext pairs as $\Delta X^1, \Delta X^0$ and $\Delta X^{r_0+R+r_1-1}, \Delta X^{r_0+R+r_1}$ are independent to keys. For remaining plaintext pairs in table T, filter out the wrong ones with sufficient bit conditions in ΔX^2 and $\Delta X^{r_0+R+r_1-2}$. Put the remaining plaintext pairs in a table T_1.

We refer to $\Delta X^{i-1}_{(j+a)\%n+n}, \Delta X^{i-1}_{(j+b)\%n+n}, \Delta X^{i-1}_{(j+c)\%n+n} \oplus \Delta X^{i-2}_{j+n}$ as parameter differences for equation $\Delta X^i_{j+n} = 0$ or 1. For valid equations, the subkey bits related to the equation $\Delta X^i_{j+n} = 0$ or 1 are divided into the following 3 conditions:

1. When

$$(\Delta X^{i-1}_{(j+a)\%n+n}, \Delta X^{i-1}_{(j+b)\%n+n}) = (1, 0),$$

the variables of the equation are the subkey bits that are linear to $X^{i-1}_{(j+b)\%n+n}$ and the subkey bits to be guessed are those that influence

$$X^{i-2}_{(j+b+a)\%n+n}, X^{i-2}_{(j+b+b)\%n+n}, X^{i-2}_{(j+b+c)\%n+n}, X^{i-2}_{(j+b)\%n}$$

and have not been deduced or guessed before;

2. When

$$(\Delta X^{i-1}_{(j+a)\%n+n}, \Delta X^{i-1}_{(j+b)\%n+n}) = (0, 1),$$

the variables of the equation are the subkey bits that are linear to $X^{i-1}_{(j+a)\%n+n}$ and the subkey bits to be guessed are those that influence

$$X^{i-2}_{(j+a+a)\%n+n}, X^{i-2}_{(j+a+b)\%n+n}, X^{i-2}_{(j+a+c)\%n+n}, X^{i-2}_{(j+a)\%n}$$

and have not been deduced or guessed before;

3. When

$$(\Delta X^{i-1}_{(j+a)\%n+n}, \Delta X^{i-1}_{(j+b)\%n+n}) = (1, 1),$$

the variables of the equation are the linear combination of subkey bits linear to $X^{i-1}_{(j+b)\%n+n}$ and subkey bits linear to $X^{i-1}_{(j+a)\%n+n}$ and the subkey bits to be guessed are those that influence

$$X^{i-2}_{(j+b+a)\%n+n}, X^{i-2}_{(j+b+b)\%n+n}, X^{i-2}_{(j+b+c)\%n+n}, X^{i-2}_{(j+b)\%n}, X^{i-2}_{(j+a+a)\%n+n},$$

$$X^{i-2}_{(j+a+c)\%n+n}, X^{i-2}_{(j+a)\%n}$$

and have not been deduced or guessed before.

For each text bit, we use a recursive algorithm to determine the subkeys bits that influence it and subkey bits that are linear to it. The pseudo code is in Algorithm 1.

Algorithm 1 Generate related key bits for X^i_j in encryption direction

Input Round i and bit position j
Output: $[Influen\_keys, Linear\_keys]$
function RelatedKeys(i, j)
    $Influent\_keys$= [ ], $Linear\_keys$=[ ]
    if $i = 0$ **then**
        return $[Influent\_keys, Linear\_keys]$
    else
        if $j < n$ **then**
            return RelatedKeys$(i-1, j+n)$
        else
            $[I_0, L_0]$=RelatedKeys$(i-1, (j+a)\%n+n)$
            $[I_1, L_1]$=RelatedKeys$(i-1, (j+b)\%n+n)$
            $[I_2, L_2]$=RelatedKeys$(i-1, (j+c)\%n+n)$
            $[I_3, L_3]$=RelatedKeys$(i-1, j\%n)$
            $Linear\_keys$=$L_2 \cup L_3 \cup K^{i-1}_{j\%n}$
            $Influent\_keys = I_0 \cup I_1 \cup I_2 \cup I_3 \cup K^{i-1}_{j\%n}$
            return $[Influent\_keys, Linear\_keys]$
        end if
    end if
end function

For sufficient key bits in the appended r_1 rounds, we process each bit in the decryption direction and give the formulas and pseudo code in Appendix A. After processing all sufficient bit conditions in the prefixed and appended rounds, we get a table of subkey bits variables corresponding to different parameter conditions for each sufficient bit condition (see Table 5 for example).

It can be seen that whether a specific subkey bit can be deduced in an equation corresponding to a sufficient bit condition depends on other 3 parameter bit differences. Some bit differences may act as parameter in more than one sufficient bit conditions and therefore we should process such sufficient bit conditions together. Specifically, we gather sufficient bit conditions with related parameters into one group and calculate the average number of subkey bits values for the group. In each round, suppose we put the original order of sufficient bit conditions in *Index_order* and the corresponding parameter sets in *Para_sets*, we use Algorithm 2 to group sufficient bit conditions.

Algorithm 2 Group sufficient bit conditions in one round

```
procedure GROUP(Index_order, Para_sets)
    Assert length(Index_order)=length(Para_sets)
    k=0
    while k <length(Index_order) do
        flag=0
        j=k+1
        while j <length(Index_order) do
            if Para_sets[j] ∩ Para_sets[k] is not empty then
                Index_order[k] = Index_order[k] ∪ Index_order[j]
                Remove Index_order[j] from Index_order
                Para_sets[k] = Para_sets[k] ∪ Para_sets[j]
                Remove Para_sets[j] from Para_sets
                flag=1
            else
                j++
            end if
        end while
        if flag=0 then
            k++
        end if
    end while
end procedure
```

In an actual attack, for each round, firstly guess the subkey bits to get the specific equations in this round. Then deduce the values of subkey bit variables in the equations according to parameter difference values group by group. In the j-th group, if we guess g_j subkey bits to get specific equations that totally involve k_j subkey bit variables and there are $t_{j,i}$ parameter conditions in each of which we correspondingly get $v_{j,i}$ values of subkey bit variables, the average number of values for the $(g_j + k_j)$ subkey bits in this group is $2^{g_j} \cdot \frac{\sum_i t_{j,i} v_{j,i}}{\sum_i t_{j,i}}$. For all groups, we get $\prod_j (2^{g_j} \cdot \frac{\sum_i t_{j,i} v_{j,i}}{\sum_i t_{j,i}})$ values of $\sum_j (g_j + k_j)$ subkey bits. For all extended rounds (or say groups), if the number of involved subkey bits (include the guessed ones and deduced ones) is less than the length of master key, we are able to launch an attack with time complexity less than exhaustive search.

Note that there are two types of repeats in subkey bit variables and guessed subkey bits when combing the numbers of values of subkey bits in all groups. The first one is due to that some subkey bits are variables of more than one group. The second one is that a linear combination of some subkey bits is a variable of an equation that may be deduced and then each of the subkey bits is again need to be guessed and thus one bit is repeated. When launching an actual attack, all these bits should be preserved as there are conditions that no specific value of the subkey bit variable is get from an equation. However, when calculating the complexity of the attack, we should eliminate the repeated bits as we take expected number of values of variables in each group.

3.4 Calculate Complexity of Attacks

Given the differential with high probability and number of rounds that we add before and after the differential, the program can give out the number of all subkey bits involved in the extended rounds $|sk|$ and the number of solutions to these subkey bits for each pair in T_1, say C_s in [15]. According to [15], a wrong subkey occurs with probability $p_e = \frac{C_s}{2^{|sk|}}$ and the expected count of a wrong subkey for all pairs in T_1 is $\lambda_e = N_r \times p_e$. Combining the complexity of searching subkey bits involved in the extended paths that get more than $s = \lfloor \lambda_r \rfloor$ hits and the complexity of tranverse the remaining subkey bits, the time complexity of the attack is domimated by

$$T_{es} = 2^{mn} \times (1 - Poisscdf(s, \lambda_e)), \tag{3}$$

where $Poissoncdf(x, y)$ is the cumulative distribution function of Poisson distribution with expectation y. The success probability is

$$1 - Poissoncdf(s, \lambda_r), \tag{4}$$

where $Poissoncdf(s, \lambda_r)$ denotes the probability that there is no subkey bits with more than s hits.

4 Differential Attacks on Simeck with Dynamic Key-guessing Techniques

4.1 A Differential of Simeck32/64

Though several differentials with high probability of Simeck family were given in [21], we want to get new differentials with lower hamming weight. Using automatic search method with MILP techniques [13,18,19,20], we find a 13-round differential characteristic of Simeck32/64 with probability 2^{-38} (see Table 2). Then we search all differential characteristics with the same input and output differences and with probability q such that $2^{-50} \leq q \leq 2^{-38}$. The distribution of the differential characteristics is given in Table 3. Combing all the differential characteristics we get that the probability of the differential $(0x0, 0x2) \rightarrow (0x2, 0x0)$ is about $2^{-29.64}$.

Table 2: A differential characteristic of 13-round Simeck32/64 with probability 2^{-38}

Rounds	The input differences
0(Input)	0000000000000000 0000000000000010
1	0000000000000010 0000000000000000
2	0000000000000100 0000000000000010
3	0000000000001010 0000000000000100
4	0000000000010000 0000000000001010
5	0000000000111010 0000000000010000
6	0000000000001100 0000000000111010
7	0000000000101010 0000000000001100
8	0000000000010000 0000000000101010
9	0000000000001010 0000000000010000
10	0000000000000100 0000000000001010
11	0000000000000010 0000000000000100
12	0000000000000000 0000000000000010
13(Output)	0000000000000010 0000000000000000

Table 3: The distribution of the characteristics of Simeck32 in the differential with input and output difference $(0000, 0002) \rightarrow (0002, 0000)$. The invalid characteristics is due to the special property of the dependent inputs of the AND operations in Simeck [1,19,20].

Prob.	2^{-38}	2^{-40}	2^{-41}	2^{-42}	2^{-43}	2^{-44}	2^{-45}	2^{-46}	2^{-47}	2^{-48}	2^{-49}	2^{-50}	Invalid
#Char.	4	62	52	427	637	2427	4384	12477	22742	48324	62039	50411	169458

4.2 Results on Simeck

We use differentials with high probability to evaluate the security of Simeck family regarding differential attacks with dynamic key-guessing techniques. The outputs of our program provide all information about the subkey bits corresponding to all sufficient bit conditions. Due to page limits, we give the details of dynamic key-guessing data in `http://pan.baidu.com/s/1jGyBwj0` and give basic information of the attacks in the following.

For Simeck32/64, we adapt two differentials. The first one is $(0x8000, 0x4011) \to (0x4000, 0x0)$ that covers 13 rounds with probability $2^{-27.28}$ in [21]. We prefix 3 rounds and append 5 rounds to the differential. Building 2^{14} structures with 2^{16} plaintexts in each structure we are expect to get $2^{31.2}$ pairs in T_1 and finally 3.29 right pairs. In the dynamic key-guessing procedure we are expect to get $2^{19.11}$ values of 53 subkey bits. According to the calculation method in Section 3.4, the time complexity and success probability of the attack are $2^{52.34}$ and 41.7%. The extended differential path of the 21-round Simeck32/64 is in Table 4. We demonstrate the solutions of subkey bits in Round 2 in Table 5.

Table 4: Sufficient Conditions of Extended Differential Path of 21-round Simeck32/64

Rounds	Input Differences of Each Round
0	$1,\ast,0,0,0,\ast,\ast,\ast,0,\ast,\ast,\ast,\ast,1,\ast,\ast,\ast,\ast,\ast,0,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast$
1	$\mathbf{0},\ast,\mathbf{0},0,\mathbf{0},\mathbf{0},\ast,\mathbf{0},\mathbf{0},\mathbf{0},\ast,\ast,\ast,\mathbf{0},\mathbf{1},\ast,1,\ast,0,0,0,\ast,\ast,\ast,0,\ast,\ast,\ast,\ast,1,\ast,\ast$
2	$0,\mathbf{1},0,0,0,\mathbf{0},\mathbf{0},\mathbf{0},0,\mathbf{0},\mathbf{0},\mathbf{1},\mathbf{0},0,\mathbf{0},\mathbf{1},0,\ast,0,0,0,0,\ast,0,0,0,\ast,\ast,\ast,0,1,\ast$
3	$1,\mathbf{0},0,0,0,0,\mathbf{0},0,0,0,\mathbf{0},\mathbf{0},\mathbf{0},0,0,\mathbf{0},0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1$
3→16	13-round differential
16	$0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,\mathbf{0},0,0,0,0,0,0,0,0,0,0,\mathbf{0},0,0,0$
17	$1,\ast,0,0,0,0,0,0,0,0,0,0,\ast,0,0,0,\mathbf{0},\mathbf{1},0,0,0,0,0,\mathbf{0},0,0,0,\mathbf{0},\mathbf{0},0,0,0$
18	$\ast,\ast,0,0,0,0,0,\ast,0,0,0,\ast,\ast,0,0,1,\mathbf{1},\ast,\mathbf{0},0,0,0,\mathbf{0},\mathbf{0},0,0,\mathbf{0},\mathbf{0},\ast,0,0,\mathbf{0}$
19	$\ast,\ast,\ast,0,0,0,\ast,\ast,0,0,\ast,\ast,\ast,0,1,\ast,\ast,\ast,\mathbf{0},0,0,\mathbf{0},\mathbf{0},\ast,0,\mathbf{0},\mathbf{0},\ast,\ast,\mathbf{0},\mathbf{0},\mathbf{1}$
20	$\ast,\ast,\ast,0,0,\ast,\ast,\ast,0,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,0,\mathbf{0},\mathbf{0},\ast,\ast,\mathbf{0},\mathbf{0},\ast,\ast,\ast,\mathbf{0},\mathbf{1},\ast$
21	$\ast,\ast,\ast,0,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\ast,\mathbf{0},\mathbf{0},\ast,\ast,\ast,\mathbf{0},\ast,\ast,\ast,\ast,\ast,\ast,\ast$

Table 5: Solutions of Subkey Bits in Round 2 of 21-round Simeck32/64

Rounds	Bit Conditions	Solutions of Key Bits to Equations	Conditions Leading to Solutions	Pr	$\Pr^F$
		Discard the pair	$(\Delta X_{17}^1, \Delta X_{22}^1, \Delta X_{17}^0) = (0,0,0)$		$\frac{1}{8}$
	$\Delta X_{17}^2 = 1 \Leftrightarrow$	$*$	$(\Delta X_{17}^1, \Delta X_{22}^1, \Delta X_{17}^0) = (0,0,1)$	$\frac{1}{8}$	
	$\Delta(X_{17}^1 \wedge X_{22}^1)$	K_1^0	$(\Delta X_{17}^1, \Delta X_{22}^1) = (0,1)$	$\frac{1}{4}$	
	$\oplus \Delta X_{17}^0 = 1$	K_6^0	$(\Delta X_{17}^1, \Delta X_{22}^1) = (1,0)$	$\frac{1}{4}$	
		$k_1^0 \oplus K_6^0$	$(\Delta X_{17}^1, \Delta X_{22}^1) = (1,1)$	$\frac{1}{4}$	
	$\Delta X_{27}^2 = 1 \Leftrightarrow$	Discard the pair	$(\Delta X_{27}^1, \Delta X_{28}^1 \oplus \Delta X_{27}^0) = (0,0)$		$\frac{1}{4}$
	$\Delta X_{27}^1 \wedge X_{16}^1$	$*$	$(\Delta X_{27}^1, \Delta X_{28}^1 \oplus \Delta X_{27}^0) = (0,1)$	$\frac{1}{4}$	
	$\oplus \Delta X_{28}^1 \oplus \Delta X_{27}^0 = 1$	K_0^0	$\Delta X_{27}^1 = 1$	$\frac{1}{2}$	
		Discard the pair	$(\Delta X_{28}^1, \Delta X_{17}^1, \Delta X_{28}^0) = (0,0,1)$		$\frac{1}{8}$
	$\Delta X_{28}^2 = 0 \Leftrightarrow$	$*$	$(\Delta X_{28}^1, \Delta X_{17}^1, \Delta X_{28}^0) = (0,0,1)$	$\frac{1}{8}$	
	$\Delta(X_{28}^1 \wedge X_{17}^1)$	K_{12}^0	$(\Delta X_{28}^1, \Delta X_{17}^1) = (0,1)$	$\frac{1}{4}$	
	$\oplus \Delta X_{22}^0 = 0$	K_1^0	$(\Delta X_{28}^1, \Delta X_{17}^1) = (1,0)$	$\frac{1}{4}$	
		$K_1^0 \oplus K_{12}^0$	$(\Delta X_{28}^1, \Delta X_{17}^1) = (1,1)$	$\frac{1}{4}$	
		Discard the pair	$(\Delta X_{22}^1, \Delta X_{27}^1, \Delta X_{22}^0) = (0,0,1)$		$\frac{1}{8}$
	$\Delta X_{22}^2 = 0 \Leftrightarrow$	$*$	$(\Delta X_{22}^1, \Delta X_{27}^1, \Delta X_{22}^0) = (0,0,0)$	$\frac{1}{8}$	
	$\Delta(X_{22}^1 \wedge X_{27}^1)$	K_6^0	$(\Delta X_{22}^1, \Delta X_{27}^1) = (0,1)$	$\frac{1}{4}$	
2(10)	$\oplus \Delta X_{22}^0 = 0$	K_{11}^0	$(\Delta X_{22}^1, \Delta X_{27}^1) = (1,0)$	$\frac{1}{4}$	
		$K_6^0 \oplus K_{11}^0$	$(\Delta X_{22}^1, \Delta X_{27}^1) = (1,1)$	$\frac{1}{4}$	
	$\Delta X_{23}^2 = 0 \Leftrightarrow$	Discard the pair	$(\Delta X_{28}^1, \Delta X_{23}^0) = (0,1)$		$\frac{1}{4}$
	$\Delta X_{28}^1 \wedge X_{23}^1$	$*$	$(\Delta X_{28}^1, \Delta X_{23}^0) = (0,0)$	$\frac{1}{4}$	
	$\oplus \Delta X_{23}^0 = 0$	K_7^0	$\Delta X_{28}^1 = 1$	$\frac{1}{2}$	
		Discard the pair	$(\Delta X_{26}^1, \Delta X_{31}^1, \Delta X_{27}^1 \oplus \Delta X_{26}^0) = (0,0,1)$		$\frac{1}{8}$
	$\Delta X_{26}^2 = 0 \Leftrightarrow$	$*$	$(\Delta X_{26}^1, \Delta X_{31}^1, \Delta X_{27}^1 \oplus \Delta X_{26}^0) = (0,0,1)$	$\frac{1}{8}$	
	$\Delta(X_{26}^1 \wedge X_{31}^1)$	K_{10}^0	$(\Delta X_{26}^1, \Delta X_{31}^1) = (0,1)$	$\frac{1}{4}$	
	$\oplus \Delta X_{27}^1 \oplus \Delta X_{26}^0 = 0$	K_{15}^0	$(\Delta X_{26}^1, \Delta X_{31}^1) = (1,0)$	$\frac{1}{4}$	
		$K_{10}^0 \oplus K_{15}^0$	$(\Delta X_{26}^1, \Delta X_{31}^1) = (1,1)$	$\frac{1}{4}$	
	$\Delta X_{21}^2 = 0 \Leftrightarrow$	Discard th pair	$(\Delta X_{26}^1, \Delta X_{22}^1 \oplus \Delta X_{21}^0) = (0,1)$		$\frac{1}{4}$
	$\Delta X_{26}^1 \wedge X_{21}^1$	$*$	$(\Delta X_{26}^1, \Delta X_{22}^1 \oplus \Delta X_{21}^0) = (0,0)$	$\frac{1}{4}$	
	$\oplus \Delta X_{22}^1 \oplus \Delta X_{21}^0 = 0$	K_5^0	$\Delta X_{26}^1 = 1$	$\frac{1}{2}$	
	$\Delta X_{31}^2 = 1 \Leftrightarrow$	Discard th pair	$(\Delta X_{31}^1, \Delta X_{31}^0) = (0,0)$		$\frac{1}{4}$
	$\Delta X_{31}^1 \wedge X_{20}^1$	$*$	$(\Delta X_{31}^1, \Delta X_{31}^0) = (0,1)$	$\frac{1}{4}$	
	$\oplus \Delta X_{31}^0 = 1$	K_4^0	$\Delta X_{31}^1 = 1$	$\frac{1}{2}$	
	$\Delta X_{25}^2 = 0 \Leftrightarrow$				
	X_{25}^1	K_9^0		1	
	$\oplus X_{26}^1 \oplus \Delta X_{25}^0 = 0$				
	$\Delta X_{30}^2 = 0 \Leftrightarrow$				
	X_{19}^1	K_3^0		1	
	$\oplus X_{31}^1 \oplus \Delta X_{30}^0 = 0$				

In the third column of Table 5, $*$ means the variables in this equation can take both values (0 and 1) and a specific subkey bit means this bit take a definite value. The bold lines are group split lines.

The second differential we use is the one from Section 4.1. We add 4 rounds on the top and 5 rounds at the bottom. With 2^{18} structures containing 2^{14} plaintexts each, we are expected to get $2^{31.9}$ pairs in T_1 and finally 2.56 right pairs. We are expect to get $2^{21.09}$ values of 54 subkey bits in dynamic key-guessing procedure. The time complexity and success probability are $2^{57.88}$ and 47.1%. The extended differential path of 22-round Simeck32/64 is in Table 7 in Appendix B.

For Simeck48/96, we use the differential $(0x400000, 0xe00000) \rightarrow (0x400000, 0x200000)$ in [21] that covers 20 rounds with probability $2^{-43.65}$. We append 4 rounds on top and 4 rounds at bottom. With 2^{18} structures with 2^{28} plaintexts in each, we are expected to get $2^{50.46}$ plaintext pairs in T_1 and finally 2.54 right pairs. There are $2^{32.89}$ values of 75 subkey bits in dynamic key-guessing procedure and the time complexity and success probability are $2^{68.31}$ and 46.8%. The extended differential path of the 28-round Simeck48/96 is in Table 8 in Appendix B.

For Simeck64/128, we use the differential $(0x0, 0x4400000) \rightarrow (0x8800000, 0x400000)$ in [21] that covers 26 rounds with probability $2^{-60.02}$. We append 4 rounds on top and 4 rounds at bottom. With 2^{42} structures with 2^{21} plaintexts in each, we are expected to get $2^{38.59}$ plaintext pairs in T_1 and finally 3.94 right pairs. There are $2^{41.72}$ values of 82 subkey bits in dynamic key-guessing procedure and the time complexity and success probability are $2^{116.27}$ and 55.5%. If we add one more round on top, we are able to attack 35-round Simeck64/128 with the same data and time complexity and success probability. The difference is that we choose 2^{31} structures of 2^{32} plaintexts and we get $2^{67.26}$ values of 118 subkey bits in the dynamic key guessing procedure. The extended differential path of the 35-round Simeck64/128 is in Table 9 in Appendix B.

We conclude the attacks on reduced versions of Simeck in Table 6.

Table 6: Differential Attacks on Reduced Simeck

Versions	Attacked Rounds	$\|sk\|$	λ_e	λ_r	Chosen Count	Data Complexity	Time Complexity	Success Prob.
Simeck32/64	21	53	$2^{-2.678}$	3.29	4	2^{30}	$2^{48.52}$	41.7%
Simeck32/64	22	54	2^{-1}	2.56	3	2^{32}	$2^{57.88}$	47.1%
Simeck48/96	28	75	$2^{-8.365}$	2.54	3	2^{46}	$2^{68.31}$	46.8%
Simeck64/128	34	82	$2^{-1.678}$	3.94	4	2^{63}	$2^{116.34}$	55.5%
Simeck64/128	35	118	$2^{-1.678}$	3.94	4	2^{63}	$2^{116.34}$	55.5%

5 Conclusion

In this paper, we apply Wang *et al.*'s dynamic key-guessing techniques to a new lightweight block cipher family Simeck and give cryptanalysis results on it. The differentials we use include ones in references and also the one we get using MILP based method. We implement the dynamic key-guessing technique in a program and in some way it can help to automatically give the security estimation of SIMON and Simeck like block ciphers regarding differential attacks. As far as we are concerned, the results on Simeck in this paper are the best ones in terms of rounds attacked. Future work includes finding differentials with lower harming weight that is more adaptable to dynamic key-guessing techniques and expand the dynamic key-guessing technique to block ciphers of other structures.

References

1. Alex Biryukov, Arnab Roy and Vesselin Velichkov: Differential analysis of block ciphers SIMON and SPECK. In: Fast Software Encryption. Springer (2014)
2. Bagheri, N.: Linear Cryptanalysis of Reduced-Round SIMECK Variants. Cryptology ePrint Archive, Report 2015/716 (2015), http://eprint.iacr.org/2015/716
3. Christophe De Cannière and Christian Rechberger: Finding SHA-1 Characteristics: General Results and Applications. In: Advances in Cryptology–ASIACRYPT 2006, pp. 1–20. Springer (2006)
4. Danping Shi, Lei Hu, Siwei Sun, Ling Song, Kexin Qiao and Xiaoshuang Ma: Improved Linear (hull) Cryptanalysis of Round-reduced Versions of SIMON. Cryptology ePrint Archive, Report 2014/973 (2014), http://eprint.iacr.org/2014/973
5. Eli Biham, Alex Biryukov and Adi Shamir: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials (1999)
6. Eli Biham and Adi Shamir: Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology 4(1), 3–72 (1991)
7. Farzaneh Abed, Eik List, Stefan Lucks and Jakob Wenzel: Differential and linear cryptanalysis of reduced-round SIMON. IACR Cryptology ePrint Archive, Report 2013/526 (2013), http://eprint.iacr.org/2013/526
8. Florian Mendel, Tomislav Nad and Martin Schläffer: Finding SHA-2 Characteristics: Searching Through a Minefield of Contradictions. In: Advances in Cryptology–ASIACRYPT 2011, pp. 288–307. Springer (2011)
9. Gaëtan Leurent: Construction of Differential Characteristics in ARX Designs Application to Skein. In: Advances in Cryptology–CRYPTO 2013, pp. 241–258. Springer (2013)
10. Gangqiang Yang, Bo Zhu, Valentin Suder, Mark D. Aagaard and Guang Gong: The Simeck Family of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2015/612 (2015), http://eprint.iacr.org/2015/612
11. Hoda A. Alkhzaimi and Martin M. Lauridsen: Cryptanalysis of the SIMON family of block ciphers. IACR Cryptology ePrint Archive, Report 2013/543 (2013), http://eprint.iacr.org/2013/543
12. Javad Alizadeh, Nasour Bagheri, P.G.A.K., Sanadhya, S.K.: Linear cryptanalysis of round reduced SIMON. IACR Cryptology ePrint Archive, Report 2013/663 (2013), http://eprint.iacr.org/2013/663

13. Kexin Qiao, Lei Hu, Siwei Sun, Xiaoshuang Ma and Haibin Kan: Improved MILP Modeling for Automatic Security Evaluation and Application to FOX. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E98-A(1), 72–80 (2015)
14. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Advances in Cryptology–EUROCRYPT 1993. pp. 386–397. Springer (1994)
15. Ning Wang, Xiaoyun Wang, Keting Jia and Jingyuan Zhao: Differential Attacks on Reduced SIMON Versions with Dynamic Key-guessing Techniques. Cryptology ePrint Archive, Report 2014/448 (2014), http://eprint.iacr.org/2014/448
16. Ning Wang, Xiaoyun Wang, Keting Jia and Jingyuan Zhao: Improved differential attacks on reduced SIMON versions. IACR Cryptology ePrint Archive, Report 2014/448 (2014), http://eprint.iacr.org/2014/448
17. Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks and Louis Wingers: The SIMON and SPECK families of lightweight block ciphers. IACR Cryptology ePrint Archive, Report 2013/404 (2013), http://eprint.iacr.org/2013/404
18. Siwei Sun, Lei Hu, Ling Song, Yonghong Xie and Peng Wang: Automatic security evaluation of block ciphers with S-bP structures against related-key differential attacks. In: Inscrypt 2013 (2014)
19. Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song and Kai Fu: Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. Cryptology ePrint Archive, Report 2014/747 (2014), http://eprint.iacr.org/2014/747
20. Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma and Ling Song: Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-oriented Block Ciphers. In: Advances in Cryptology–ASIACRYPT 2014 (2014)
21. Stefan Klbl and Arnab Roy: A Brief Comparison of Simon and Simeck. Cryptology ePrint Archive, Report 2015/706 (2015), http://eprint.iacr.org/2015/706
22. Theobald, T.: How to break Shamir's asymmetic basis. In: Advances in Cryptology–CRYPTO 1995, pp. 136–147. Springer (1995)
23. Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu: Finding Collisions in the Full SHA-1. In: Advances in Cryptology–CRYPTO 2005, pp. 17–36. Springer (2005)

A Related Keys in Decryption Direction

For sufficient bit condition $\Delta X_j^i = 0$ or 1 and $j \in [0, n-1]$, in decrypt direction we have

$$\begin{aligned} \Delta X_j^i =& \Delta X_{(j+b)\%n}^{i+1} \wedge X_{(j+a)\%n}^{i+1} \oplus \Delta X_{(j+a)\%n}^{i+1} \wedge X_{(j+b)\%n}^{i+1} \oplus \Delta X_{j+b}^{i+1} \wedge \Delta X_{(j+a)\%n}^{i+1} \\ & \oplus \Delta X_{(j+c)\%n}^{i+1} \oplus \Delta X_j^{i+2}, \end{aligned} \tag{5}$$

where

$$\begin{aligned} X_{(j+a)\%n}^{i+1} =& X_{(j+a+b)\%n}^{i+2} \wedge X_{(j+a+a)\%n}^{i+2} \oplus X_{(j+a+c)\%n}^{i+2} \oplus \\ & X_{(j+a)\%n}^{i+3} \oplus K_{(j+a)\%n}^{i+1}, \\ X_{(j+b)\%n}^{i+1} =& X_{(j+b+b)\%n}^{i+2} \wedge X_{(j+b+a)\%n}^{i+2} \oplus X_{(j+b+c)\%n}^{i+2} \oplus \\ & X_{(j+b)\%n}^{i+3} \oplus K_{(j+b)\%n}^{i+1}. \end{aligned} \tag{6}$$

Algorithm 3 demonstrates how to get subkey bits that influence X_j^i and that are linear to X_j^i.

Algorithm 3 Generate related key bits for X_j^i in decryption direction

Input: Round i and bit position j
Output: $[Influen\_keys, Linear\_keys]$
function RelatedKeys(i, j)
  $Influent\_keys$= [ ], $Linear\_keys$=[ ]
  if $i = r_0 + R + r_1$ **then**
    return $[Influent\_keys, Linear\_keys]$
  else
    if $j \geq n$ **then**
      return RelatedKeys$(i+1, j\%n)$
    else
      $[I_0, L_0]$=RelatedKeys$(i, (j+a)\%n + n)$
      $[I_1, L_1]$=RelatedKeys$(i, (j+b)\%n + n)$
      $[I_2, L_2]$=RelatedKeys$(i, (j+c)\%n + n)$
      $[I_3, L_3]$=RelatedKeys$(i+1, j+n)$
      $Linear_keys$=$L_2 \cup L_3 \cup$ K_j^i
      $Influent\_keys = I_0 \cup I_1 \cup I_2 \cup I_3 \cup K_j^i$
      return $[Influent\_keys, Linear\_keys]$
    end if
  end if
end function

B Sufficient Conditions of Extended Differential Path

In the following, we provide the sufficient conditions of extended differential paths of 22-round Simeck32/64, 28-round Simeck48/96 and 35-round Simeck64/128.

Table 7: Sufficient Conditions of Extended Differential Path of 22-round Simeck32/64

Rounds	Input Differences of Each Round
0	0, 0, 0, *, *, 0, 0, *, *, *, 0, 1, *, *, *, *, 0, 0, *, *, *, 0, *, *, *, *, *, *, *, *, *, *
1	0, 0, **0**, **0**, *, 0, **0**, **0**, *, *, **0**, **0**, **1**, *, *, **0**, 0, 0, 0, *, *, 0, 0, *, *, *, 0, 1, *, *, *, *
2	0, 0, 0, **0**, **0**, 0, 0, **0**, **0**, *, 0, 0, **0**, **1**, *, **0**, 0, 0, 0, 0, *, 0, 0, 0, *, *, 0, 0, 1, *, *, 0
3	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, 1, *, 0
4	0, 0, 0, 0, 0, 0, 0, 0, 0, **0**, 0, 0, 0, 0, **0**, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
4→17	13-round differential
17	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**, 0
18	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
19	0, 0, 0, 0, *, 0, 0, 0, *, *, 0, 0, 1, *, *, 0, 0, 0, 0, **0**, **0**, 0, 0, **0**, **0**, *, 0, 0, **0**, **1**, *, **0**
20	0, 0, 0, *, *, 0, 0, *, *, *, 0, 1, *, *, *, *, 0, 0, **0**, **0**, *, 0, **0**, **0**, *, *, **0**, **0**, **1**, *, *, **0**
21	0, 0, *, *, *, 0, *, *, *, *, *, *, *, *, *, *, 0, **0**, **0**, *, *, **0**, **0**, *, *, *, **0**, **1**, *, *, *, *
22	0, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, **0**, **0**, *, *, *, **0**, *, *, *, *, *, *, *, *, *, *

Table 8: Sufficient Conditions of Extended Differential Path of 28-round Simeck48/96

Rounds	Input Differences of Each Round
0	***000000***0**************0***0****************
1	***00000000000***0****1****000000***0***********
2	***0000000000000000***01***00000000000***0****1*
3	111000000000000000000000***0000000000000000***01
4	010000000000000000000000111000000000000000000000
4→24	20-round differential
24	010000000000000000000000001000000000000000000000
25	1*100000000000000000*000010000000000000000000000
26	***000000000000*000***011*100000000000000000*000
27	***0000000*000***0****1****000000000000*000***01
28	***00*000***0**************0000000*000***0****1*

Table 9: Sufficient Conditions of Extended Differential Path of 34-round Simeck64/128

Rounds	Input Differences of Each Round
0	**********0000000*000**00***0*************00*000**00***0********
1	*0****1***000000000000*000**00************0000000*000**00***0***
2	*00***01**00000000000000000*000**0****1***000000000000*000**00**
3	*000**001*0000000000000000000000*00***01**00000000000000000*000*
4	00000100010000000000000000000000*000**001*0000000000000000000000
5	0000000000000000000000000000000000000100010000000000000000000000
5→31	26-round differential
31	0000100010000000000000000000000000000000100000000000000000000000
32	000**001*1000000000000000000000*00001000100000000000000000000000
33	00***01***0000000000000000*000**000**001*1000000000000000000000*
34	0****1****00000000000*000**00***00***01***0000000000000000*000**
35	**********000000*000**00***0****0****1****00000000000*000**00***