Hindawi
Security and Communication Networks
Volume 2018, Article ID 5160237, 11 pages
https://doi.org/10.1155/2018/5160237

Research Article

Improved Integral Attacks on SIMON32 and SIMON48 with Dynamic Key-Guessing Techniques

Zhihui Chu,[1,2] **Huaifeng Chen**,[1,2] **Xiaoyun Wang**,[1,2,3] **Xiaoyang Dong**,[3] **and Lu Li**[1,2]

[1]*Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China*
[2]*School of Mathematics, Shandong University, Jinan 250100, China*
[3]*Institute for Advanced Study, Tsinghua University, Beijing 100084, China*

Correspondence should be addressed to Xiaoyun Wang; xiaoyunwang@mail.tsinghua.edu.cn

Received 17 July 2017; Accepted 3 January 2018; Published 19 February 2018

Academic Editor: Barbara Masucci

Dynamic key-guessing techniques, which exploit the property of AND operation, could improve the differential and linear cryptanalytic results by reducing the number of guessed subkey bits and lead to good cryptanalytic results for SIMON. They have only been applied in differential and linear attacks as far as we know. In this paper, dynamic key-guessing techniques are first introduced in integral cryptanalysis. According to the features of integral cryptanalysis, we extend dynamic key-guessing techniques and get better integral cryptanalysis results than before. As a result, we present integral attacks on 24-round SIMON32, 24-round SIMON48/72, and 25-round SIMON48/96. In terms of the number of attacked rounds, our attack on SIMON32 is better than any previously known attacks, and our attacks on SIMON48 are the same as the best attacks.

1. Introduction

The integral attack, proposed by Daemen et al. [1], is an important cryptanalytic technique for symmetric-key primitives. The integral distinguisher is based on the property that when some parts of the input (constant bits) of distinguishers are held constant whereas the other parts (active bits) vary through all possibilities, the sum of all the output values equals zero at some particular locations (balanced bits). In the key recovery, the sum is random if the guessed key is incorrect, while the sum is zero if the guessed key is correct. As a powerful class of cryptanalytic techniques, integral cryptanalysis has been applied to many block ciphers, especially the ones with low-degree round functions.

SIMON is a family of ten lightweight block ciphers designed by the US National Security Agency [2]. The SIMON2n/mn family of lightweight block ciphers have classical Feistel structures with 2n-bit block size and mn-bit key, where n is the word size.

SIMON has been extensively scrutinized [3–25]. As an ultralightweight primitive, SIMON is a very good target for integral cryptanalysis. In integral cryptanalysis, Wang et al. [21] experimentally found an integral distinguisher for 14 rounds of SIMON32 and mounted a key-recovery attack on 21-round SIMON32. At EUROCRYPT 2015, Todo proposed the division property [17], which is a generalized integral property. This new technique enables the cryptographers to propagate the integral property in a more precise manner. As a result, an 11-round integral distinguisher of SIMON48 was found. Subsequently, using the bit-based division property, Todo and Morii proved the 14-round distinguisher of SIMON32 theoretically in [18]. However, searching integral characteristics by the bit-based division property requires much time and memory complexity. In order to overcome the problem, Xiang et al. [23] proposed a state partition to achieve a trade-off between the accuracy of the integral distinguisher and the time-memory complexity. Accordingly, Todo's result was improved by one round for SIMON48. Afterwards, MILP

method was applied by Xiang et al. [22] to find integral characteristics of some lightweight block ciphers, including a 15-round integral distinguisher for SIMON48. At ACNS 2016, some integral distinguishers of SIMON-like ciphers were constructed by Kondo et al. [10]. However, the block size considered is only 32 bits. Later in [7], with the equivalent-subkey technique, Fu et al. presented integral attacks on 22-round SIMON32, 22-round SIMON48/72, and 23-round SIMON48/96. Good results [6, 13, 20] were achieved in differential and linear cryptanalysis, as well. The cryptanalytic results that attack the most rounds of SIMON were obtained in [6], and these results were achieved by linear hull cryptanalysis. The most efficient differential and linear attacks on SIMON were presented with the help of dynamic key-guessing techniques.

With regard to dynamic key-guessing techniques, they were initially proposed to improve the differential attacks on SIMON [20]. The techniques, which exploit the property of AND operation, help reduce the average number of guessed key bits significantly in differential cryptanalysis. Then they were applied to linear hull attacks on SIMON [6]. In both [6, 20], with the techniques above, the adversaries are able to extend previous differential (resp., linear hull) results on SIMON by 2 to 4 more rounds, using existing differential (resp., linear hull) distinguishers. Subsequently, Qiao et al. [13] released a tool, which provides the differential security evaluation of SIMON given differential distinguishers of high probability. Moreover, with newly proposed differentials [9], Qiao et al. improved differential attacks against SIMON, using the techniques. Also in the differential cryptanalysis and linear cryptanalysis of Simeck [26], good results [13, 27] have been obtained by using dynamic key-guessing techniques. Up to now, the dynamic key-guessing techniques have only been combined with linear and differential cryptanalysis methods. There is no attempt to combine the dynamic key-guessing techniques with integral attack so far.

Besides the above results under the single-key model, the security of SIMON has also been evaluated under the related-key [11] and known-key [8] models. In the related-key setting, Kondo et al. [11] constructed a 15-round related-key impossible differential distinguisher of SIMON32.

Our Contributions. In this paper, we first apply dynamic key-guessing techniques to integral attacks. In our improved integral cryptanalysis, we extend dynamic key-guessing techniques to compute the sum, which is in the form of $\sum_x f(x,k) \cdot V[x]$, where f is a nonlinear Boolean function and $V[x]$ are counters for x. The dynamic key-guessing techniques improve the time complexity of the computation significantly. Please see the following example. Suppose $f(x,k) = 1 \oplus f_1(x_1,k_1)\&f_2(x_2,k_2)$, where $x = x_1 \parallel x_2$, $k = k_1 \parallel k_2$, and f_1 and f_2 are two Boolean functions. We guess k_1 at first; then we split $x = x_1 \parallel x_2$ into two sets: $S_1 = \{x \mid f_1(x_1,k_1) = 0\}$ and $S_2 = \{x \mid f_1(x_1,k_1) = 1\}$. We continue to compute the sum for each set. For set S_1, there is no need to guess k_2 since $f(x,k) = 1$ when $x \in S_1$. Finally, we sum them up.

Using the dynamic key-guessing techniques, we present improved integral attacks on SIMON32 and SIMON48 in the single-key model. We present integral attacks on 24-round SIMON32, 24-round SIMON48/72, and 25-round SIMON48/96. In terms of the number of attacked rounds, our attack on SIMON32 is better than any previously known attacks, and our attacks on SIMON48 are the same as the best attacks. In order to verify the correctness of our approach, we implement the summation procedure of the integral attack on 22-round SIMON32. A summary of our results is given in Table 1.

Outline. This paper is structured as follows. Section 2 briefly describes the specification of SIMON and some integral distinguishers. In Section 3, we discuss the time reduction in integral cryptanalysis of bit-oriented block ciphers. In Section 4, we present improved integral attacks on SIMON32 and SIMON48. In Section 4.1, we give the experimental result. Finally, Section 5 draws conclusions.

2. Preliminaries

2.1. Notations

n: the word size

x_i: the ith bit of bit string x

$x_{[i-j]}$ (or $x_i - x_j$): the ith to the jth bits of bit string x

$x_{i_1,\ldots,i_n}$: the XOR sum of x_i, where $i = i_1,\ldots,i_n$, i.e., $\bigoplus_{i\in\{i_1,\ldots,i_n\}} x_i$

$x \parallel y$: concatenation of two bit strings x and y

X^r: the input of round r or output of round $(r-1)$

X_L^r, X_R^r: the left and right halves of X^r, that is, $X^r = X_L^r \parallel X_R^r$

$X_{L,i}^r$ (resp. $X_{R,i}^r$): the ith bit of bit string X_L^r(resp. X_R^r)

$X_{L,[i-j]}^r$ (or $X_{L,i}^r - X_{L,j}^r$): the ith to the jth bits of bit string X_L^r

$X_{R,[i-j]}^r$ (or $X_{R,i}^r - X_{R,j}^r$): the ith to the jth bits of bit string X_R^r

K^r: the subkey used in rth round

$k\backslash\{k_{i_1},\ldots,k_{i_n}\}$: a new bit string, of which bits are derived from bit string k, excluding $\{k_{i_1},\ldots,k_{i_n}\}$

$\oplus$: bitwise XOR

&: bitwise AND

$x \lll t$: a left circular shift of bit string x by t bits

$V[x], W[x], Y[x]$: counters for bit string x

$B^k(f)$: $B^k(f) = \sum_x f(x,k)\cdot V[x]$, where f is a Boolean function of x and k (actually, $B^k(f)$ are counters for k)

$F(x)$: $F(x) = [(x \lll 1)\&(x \lll 8)] \oplus (x \lll 2)$

$F(x)_i$: the ith bit of bit string $F(x)$

2.2. Description of SIMON2n/mn. SIMON2n/mn is a two-branch balanced Feistel network with $2n$-bit block size and mn-bit key, where n is the word size. There are 10 variants for SIMON. The parameters of SIMON32/64, SIMON48/72,

Table 1: Summary of some related results for SIMON32 and SIMON48.

Target	Rounds	Data	Time	Memory (bytes)	Success probability	Attack type	Source
SIMON32/64	21	2^{31}	2^{63}E	2^{54}	1	Integ.	[21]
	21	2^{31}	$2^{55.25}$E	-	51%	Diff.	[20]
	22	2^{31}	2^{63}E	$2^{55.8}$	1	Integ.	[7]
	22	2^{32}	$2^{58.76}$E	-	31.5%	Diff.	[13]
	23	$2^{31.19}$	$2^{57.19}$TWO+$2^{61.84}$A+2^{56}E	-	28%	Lin. hull	[6]
	24	2^{32}	2^{63}E	$2^{33.64}$	1	Integ.	Section 4.3
SIMON48/72	18	-	-	-	1	Integ.	[23]
	22	2^{47}	2^{71}E	2^{42}	1	Integ.	[7]
	23	2^{47}	$2^{63.25}$E	-	48%	Diff.	[20]
	24	$2^{47.92}$	$2^{69.92}$ONE+$2^{67.89}$A+2^{56}E	-	-	Lin. hull	[6]
	24	2^{48}	2^{71}E	2^{50}	1	Integ.	Appendix B.2
SIMON48/96	19	-	-	-	1	Integ.	[23]
	23	2^{47}	2^{95}E	2^{47}	1	Integ.	[7]
	24	2^{47}	$2^{87.25}$E	-	48%	Diff.	[20]
	24	2^{48}	$2^{78.99}$E	-	47.5%	Diff.	[13]
	25	$2^{47.92}$	$2^{91.92}$TWO+$2^{89.89}$A+2^{80}E	-	-	Lin. hull	[6]
	25	2^{48}	2^{95}E	2^{50}	1	Integ.	Appendix B.3

Note. This table summaries our results along with some previous major results of SIMON32 and SIMON48 in the single-key setting; E: encryption; A: addition; TWO: two rounds of encryption or decryption; ONE: one round of encryption or decryption.

Table 2: Parameters of SIMON32 and SIMON48.

Block size ($2n$)	Key size (mn)	Rounds
32 ($n = 16$)	64 ($m = 4$)	32
48 ($n = 24$)	72 ($m = 3$)	36
	96 ($m = 4$)	36

and SIMON48/96 are listed in Table 2, since only these three variants are considered in this paper. Let $X^i = X_L^i \parallel X_R^i$ denote the input of round i and $X^{i+1} = X_L^{i+1} \parallel X_R^{i+1}$ be the output of round i. The subkey used in round i is denoted by K^i. The ith round is as follows (also see Figure 1):

$$\begin{aligned} X_R^{i+1} &= X_L^i, \\ X_L^{i+1} &= F\left(X_L^i\right) \oplus X_R^i \oplus K^i, \end{aligned} \tag{1}$$

where the internal nonlinear function F is defined as

$$F\left(X_L^i\right) = \left[\left(X_L^i \lll 1\right) \& \left(X_L^i \lll 8\right)\right] \oplus \left(X_L^i \lll 2\right). \tag{2}$$

The key schedules are different depending on the key size. Please refer to [2] for more details.

2.3. Integral Distinguishers of SIMON32 and SIMON48. Attackers prepare a set of texts where some bits (constant bits) are fixed to same values and the other bits (active bits) range over all possible values. If some bits (balanced bits) in the encrypted texts sum to zero after $\overline{R}$ rounds encryption, the cipher has an $\overline{R}$-round integral distinguisher.

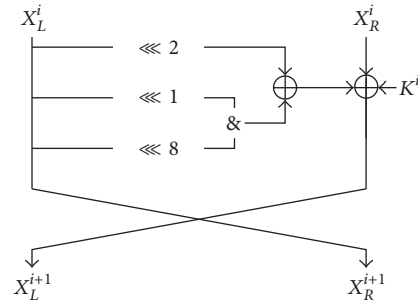

Figure 1: Round function of SIMON.

Wang et al. [21] found a 14-round integral distinguisher of SIMON32 experimentally. Later, Todo and Morii [18] proved the correctness of this distinguisher using division property. Also, Fu et al. [7] revealed this distinguisher from the view of degree of the Boolean function. Integral characteristics of SIMON32 and SIMON48 were found in [7, 18, 21, 22]. And we apply them to our attacks. The constant bit, active bit, balanced bit, and unknown bit are labeled as `c`, `a`, `b`, and `?`, respectively. The integral characteristics used in this literature are as follows.

(i) SIMON32's 14-round distinguisher:

Input: (`caaaaaaaaaaaaaaa`, `aaaaaaaaaaaaaaaa`)

Output: (????????????????, ?b??????b??????b)

(ii) SIMON48's 15-round distinguisher:

Input: (caaaaaaaaaaaaaaaaaaaaaaa, aaaaaaaaaaaaaaaaaaaaaaaa)

Output: (????????????????????????, bbbbbbbbbbbbbbbbbbbbbbbb)

3. Time Reduction in Integral Attacks on Bit-Oriented Block Ciphers

Suppose the input of the integral distinguisher is from the set

$$S_I = \left\{X^i = \left(X_L^i, X_R^i\right) \mid X_{L,0}^i = c, X_{L,[1-15]}^i \in \mathbb{F}_2^{n-1}, X_R^i \in \mathbb{F}_2^n\right\}. \tag{3}$$

After $\overline{R}$-round encryption, some bits of the output $X^{i+\overline{R}}$ are balanced. For simplicity, let the first bit of the right part, that is, $X_{R,0}^{i+\overline{R}}$, be balanced. We add α rounds before the distinguisher and append β rounds after it. Let the Boolean expressions of $X_{L,0}^i$ and $X_{R,0}^{i+\overline{R}}$ be functions represented as $f_{E_\alpha}(x_P, k_P)$ and $f_{D_\beta}(x_C, k_C)$, where x_P, x_C, k_P, and k_C are effective bit strings derived from the plaintext, the ciphertext, and involved subkeys.

We briefly outline the idea of our integral attacks on $\alpha + \overline{R} + \beta$ rounds of ciphers. Given the entire codebook, we guess some subkey bits and carry out the first α rounds' encryption. Then choose a set of states that form the input space S_I. For the corresponding ciphertexts, we guess the related subkey bits and decrypt the last β rounds to check if the target bit $X_{R,0}^{i+\overline{R}}$ is balanced.

In general, the time complexity of the integral attack is roughly $\mathcal{O}(2^l \cdot N)$, where l is the number of guessed subkey bits and N denotes the number of plaintext-ciphertext pairs. But we can optimize it with dynamic key-guessing techniques.

3.1. Find Collections of Ciphertexts. Let $V[x_P, x_C]$ denote the counters into which we store the frequency of (x_P, x_C). For each guessed k_P, we traverse the whole plaintext space and make partial encryptions. If $f_{E_\alpha}(x_P, k_P) = c$, we store the corresponding ciphertext. Thus, we generate new counters $W[k_P, x_C]$, which are defined as $\sum_{x_P, f_{E_\alpha}(x_P,k_P)=c} V[x_P, x_C]$.

Furthermore, if f_{E_α} is linear with some bit of k_P, say $k_{P,0}$, we let $f_{E_\alpha}(x_P, k_P) = k_{P,0} \oplus f'_{E_\alpha}(x_P, k'_P)$, where $k_P = k_{P,0} \parallel k'_P$. We now assign c the value $k_{P,0} \oplus 1$. Accordingly, $W[k'_P, x_C] = \sum_{x_P, f'_{E_\alpha}(x_P,k'_P)=1} V[x_P, x_C]$, which means that the condition $f'_{E_\alpha}(x_P, k'_P) = 1$ can be transformed to a coefficient. Therefore, it is sufficient to calculate $W[k'_P, x_C] = \sum_{x_p} f'_E(x_P, k'_P) \cdot V[x_P, x_C]$.

3.2. Compute $\sum_x f(x,k) \cdot V[x]$ with Dynamic Key-Guessing Techniques. As described above, the modeling to find the collections of ciphertexts can be converted into the task of computing another counter $W[k]$ which is defined as

$$W[k] = \sum_x f(x,k) \cdot V[x], \tag{4}$$

where f is a Boolean function of x and k, and $V[x]$ denotes the number of x. Let x be a l_1-bit value and k be a l_2-bit value. In a naive way, it needs $\mathcal{O}(2^{l_1+l_2})$ calculations of f to get the counters $W[k]$. Using dynamic key-guessing techniques, the calculation can be done with improved time complexity. The basic idea is as follows.

Let $k = k^G \parallel k^A \parallel k^B \parallel k^C$, where k^G, k^A, k^B, and k^C are l_2^G, l_2^A, l_2^B, and l_2^C bits. After guessing k^G, the set of x can be split into two sets S^A and S^B with N^A and N^B elements, respectively. For values in S^A, f is independent of k^B. Similarly, for values in S^B, f is independent of k^A. Thus, $\sum_x f(x,k) \cdot V[x] = \sum_{x \in S^A} f(x, k^G \parallel k^A \parallel k^C) \cdot V[x] + \sum_{x \in S^B} f(x, k^G \parallel k^B \parallel k^C) \cdot V[x]$. We compute the sum for each set, then we sum them up. Therefore, using dynamic key-guessing techniques, the improved time complexity becomes $\mathcal{O}(N^A \cdot 2^{l_2^G + l_2^A + l_2^C} + N^B \cdot 2^{l_2^G + l_2^B + l_2^C})$.

Again, we provide a toy example that illustrates the idea behind the improvement. Let $x \in \mathbb{F}_2^3$, $k \in \mathbb{F}_2^3$, and $f(x,k) = x_0 \oplus k_0 \oplus ((x_1 \oplus k_1) \& (x_2 \oplus k_2))$. Firstly, we guess k_1. Then,

$$\begin{aligned} &\sum_x f(x,k) \cdot V[x] \\ &\quad = \sum_{x_1 = k_1, x_0 \in \mathbb{F}_2, x_2 \in \mathbb{F}_2} (x_0 \oplus k_0) \cdot V[x] \\ &\qquad + \sum_{x_1 = k_1 \oplus 1, x_0 \in \mathbb{F}_2, x_2 \in \mathbb{F}_2} (x_{0,2} \oplus k_{0,2}) \cdot V[x]. \end{aligned} \tag{5}$$

Next, we create four counters $T_{0,0}$, $T_{0,1}$, $T_{1,0}$, and $T_{1,1}$ and assign some values to them: $T_{0,0} = V[0 \parallel k_1 \parallel 0] + V[0 \parallel k_1 \parallel 1]$, $T_{0,1} = V[1 \parallel k_1 \parallel 0] + V[1 \parallel k_1 \parallel 1]$, $T_{1,0} = V[0 \parallel (k_1 \oplus 1) \parallel 0] + V[1 \parallel (k_1 \oplus 1) \parallel 1]$, and $T_{1,1} = V[0 \parallel (k_1 \oplus 1) \parallel 1] + V[1 \parallel (k_1 \oplus 1) \parallel 0]$. Finally, $W[k] = T_{0,k_0 \oplus 1} + T_{1,k_0 \oplus k_2 \oplus 1}$. Thus, the calculation of $W[k]$ essentially requires $2 \times (2 + 2 + 2^2) = 2^4$ additions, while it takes 2^6 operations in a straightforward method.

See Appendix A for more information on the time complexity of the calculation of $\sum_x f(x,k) \cdot V[x]$.

3.3. Compute the XOR Sum of the Recovered Bit. Assume now that we obtain the new counters $W[k'_P, x_C]$. For a fixed k'_P, we guess k_C and partially decrypt each effective bit string x_C to get the value of the target bit, that is, $f_{D_\beta}(x_C, k_C)$. Then, check whether the XOR sum of the recovered bit is zero. Note that the XOR sum amounts to the parity of $\sum_{x_C} f_{D_\beta}(x_C, k_C) \cdot W[k'_P, x_C]$. For simplicity, let $f_{D_\beta}(x_C, k_C)$ be $k_{C,0} \oplus f'_{D_\beta}(x_C, k'_C)$, where $k_{C,0}$ is the first bit of k_C, $k_C = k_{C,0} \parallel k'_C$. We can omit $k_{C,0}$ since it does not affect the XOR sum. Hence, the XOR sum essentially equals the parity of new counters $Y[k'_P, k'_C]$ which is defined as $\sum_{x_C} f'_{D_\beta}(x_C, k'_C) \cdot W[k'_P, x_C]$. Also, dynamic key-guessing techniques can be applied in the last β rounds to improve the time complexity.

Table 3: Each effective bit of the Boolean expression of $X^i_{L,15}$.

x_i	Representation of x_i	k_i	Representation of k_i
x_0	$X^{i-4}_{L,7} \oplus (X^{i-4}_{L,8} \& X^{i-4}_{L,1}) \oplus X^{i-4}_{R,9} \oplus X^{i-4}_{L,11} \oplus X^{i-4}_{L,15}$	k_0	$K^{i-4}_9 \oplus K^{i-3}_{11} \oplus K^{i-3}_{15} \oplus K^{i-2}_{13} \oplus K^{i-1}_{15}$
x_1	$X^{i-4}_{L,8} \oplus (X^{i-4}_{L,9} \& X^{i-4}_{L,2}) \oplus X^{i-4}_{R,10}$	k_1	K^{i-4}_{10}
x_2	$X^{i-4}_{L,1} \oplus (X^{i-4}_{L,2} \& X^{i-4}_{L,11}) \oplus X^{i-4}_{R,3}$	k_2	K^{i-4}_3
x_3	$X^{i-4}_{L,12} \oplus (X^{i-4}_{L,13} \& X^{i-4}_{L,6}) \oplus X^{i-4}_{R,14}$	k_3	K^{i-4}_{14}
x_4	$X^{i-4}_{L,5} \oplus (X^{i-4}_{L,6} \& X^{i-4}_{L,15}) \oplus X^{i-4}_{R,7}$	k_4	K^{i-4}_7
x_5	$X^{i-4}_{L,8} \oplus (X^{i-4}_{L,9} \& X^{i-4}_{L,2}) \oplus X^{i-4}_{R,10} \oplus X^{i-4}_{L,12}$	k_5	$K^{i-4}_{10} \oplus K^{i-3}_{12}$
x_6	$X^{i-4}_{L,9} \oplus (X^{i-4}_{L,10} \& X^{i-4}_{L,3}) \oplus X^{i-4}_{R,11}$	k_6	K^{i-4}_{11}
x_7	$X^{i-4}_{L,2} \oplus (X^{i-4}_{L,3} \& X^{i-4}_{L,12}) \oplus X^{i-4}_{R,4}$	k_7	K^{i-4}_4
x_8	$X^{i-4}_{L,1} \oplus (X^{i-4}_{L,2} \& X^{i-4}_{L,11}) \oplus X^{i-4}_{R,3} \oplus X^{i-4}_{L,5}$	k_8	$K^{i-4}_3 \oplus K^{i-3}_5$
x_9	$X^{i-4}_{L,11} \oplus (X^{i-4}_{L,12} \& X^{i-4}_{L,5}) \oplus X^{i-4}_{R,13}$	k_9	K^{i-4}_{13}
x_{10}	$x_3 \oplus x_5$	k_{10}	$k_3 \oplus k_5 \oplus K^{i-2}_{14}$
x_{11}	$X^{i-4}_{L,9} \oplus (X^{i-4}_{L,10} \& X^{i-4}_{L,3}) \oplus X^{i-4}_{R,11} \oplus X^{i-4}_{L,13}$	k_{11}	$K^{i-4}_{11} \oplus K^{i-3}_{13}$
x_{12}	$X^{i-4}_{L,10} \oplus (X^{i-4}_{L,11} \& X^{i-4}_{L,4}) \oplus X^{i-4}_{R,12}$	k_{12}	K^{i-4}_{12}
x_{13}	$X^{i-4}_{L,3} \oplus (X^{i-4}_{L,4} \& X^{i-4}_{L,13}) \oplus X^{i-4}_{R,5}$	k_{13}	K^{i-4}_5
x_{14}	$X^{i-4}_{L,2} \oplus (X^{i-4}_{L,3} \& X^{i-4}_{L,12}) \oplus X^{i-4}_{R,4} \oplus X^{i-4}_{L,6}$	k_{14}	$K^{i-4}_4 \oplus K^{i-3}_6$
x_{15}	$x_4 \oplus x_8$	k_{15}	$k_4 \oplus k_8 \oplus K^{i-2}_7$
x_{16}	$X^{i-4}_{L,11} \oplus (X^{i-4}_{L,12} \& X^{i-4}_{L,5}) \oplus X^{i-4}_{R,13} \oplus X^{i-4}_{L,15}$	k_{16}	$K^{i-4}_{13} \oplus K^{i-3}_{15}$

4. Integral Attacks on SIMON32 and SIMON48

4.1. Integral Attack on 22-Round SIMON32. We start with a key-recovery attack over four rounds of partial encryption and four rounds of partial decryption, exploiting the 14-round integral characteristic. Any of balanced bits can be taken as the target bit. Here, we pick $X^{i+14}_{R,0}$. In the attack, we compress each plaintext-ciphertext pair into counters. Then we apply the approach given above to the reduced SIMON32.

The Boolean expression of the constant bit $X^i_{L,15}$ has the same general form as that of the balanced bit $X^{i+14}_{R,0}$. The general form is shown in (6). The specific information on each bit is listed in Tables 3 and 4. In the tables, X^{i-4} and X^{i+18}, respectively, denote the plaintext and the ciphertext.

$$\begin{aligned} f(x,k) = {} & x_0 \oplus k_0 \oplus ((x_1 \oplus k_1) \& (x_2 \oplus k_2)) \oplus ((x_3 \\ & \oplus k_3) \& (x_4 \oplus k_4)) \oplus [(x_5 \oplus k_5 \oplus ((x_6 \oplus k_6) \\ & \& (x_7 \oplus k_7))) \& (x_8 \oplus k_8 \oplus ((x_9 \oplus k_9) \\ & \& (x_7 \oplus k_7)))] \oplus \{(x_{10} \oplus k_{10} \oplus ((x_6 \oplus k_6) \\ & \& (x_7 \oplus k_7)) \\ & \oplus [(x_{11} \oplus k_{11} \oplus ((x_{12} \oplus k_{12}) \& (x_{13} \oplus k_{13}))) \\ & \& (x_{14} \oplus k_{14} \oplus ((x_3 \oplus k_3) \& (x_{13} \oplus k_{13})))]) \& (x_{15} \\ & \oplus k_{15} \oplus ((x_7 \oplus k_7) \& (x_9 \oplus k_9)) \\ & \oplus [(x_{14} \oplus k_{14} \oplus ((x_{13} \oplus k_{13}) \& (x_3 \oplus k_3))) \\ & \& (x_{16} \oplus k_{16} \oplus ((x_3 \oplus k_3) \& (x_4 \oplus k_4)))])\}. \end{aligned} \tag{6}$$

During the computation of $Y[k'_P, k'_C]$, we first guess k'_P; then we guess k'_C. Since there is no difference between the first and the second halves of the computation, in the following, we mainly discuss the first half, that is, the computation of

$$W\left[k'_P, x_C\right] = \sum_{x_P} f'_E\left(x_P, k'_P\right) \cdot V\left[x_P, x_C\right]. \tag{7}$$

To describe our procedure in a convenient way, we simplify our modeling. We give a brief description of the modeling. We aim to compute another counter $B^{k'}(f')$, which is defined as $\sum_x f'(x,k') \cdot V[x]$, where $k = k_0 \parallel k'$ and $f(x,k) = k_0 \oplus f'(x,k')$. Our approach is as follows.

(a) Guess k_1, k_3, k_7 and then split the texts into 8 sets according to the value $(x_1 \oplus k_1, x_3 \oplus k_3, x_7 \oplus k_7)$. Table 5 shows corresponding variants of the Boolean function $f'(x,k')$. Accordingly, we have

$$\begin{aligned} f_{000} = {} & x_0 \oplus [(x_5 \oplus k_5) \& (x_8 \oplus k_8)] \oplus \{(x_{10} \oplus k_{10} \\ & \oplus [(x_{11} \oplus k_{11} \oplus ((x_{12} \oplus k_{12}) \& (x_{13} \oplus k_{13}))) \\ & \& (x_{14} \oplus k_{14})]) \& (x_{15} \oplus k_{15} \oplus [(x_{14} \oplus k_{14}) \\ & \& (x_{16} \oplus k_{16})])\}, \\ & \vdots \\ f_{111} = {} & x_{0,2,4} \oplus k_{2,4} \oplus [(x_{5,6} \oplus k_{5,6}) \& (x_{8,9} \oplus k_{8,9})] \\ & \oplus \{(x_{6,10} \oplus k_{6,10} \\ & \oplus [(x_{11} \oplus k_{11} \oplus ((x_{12} \oplus k_{12}) \& (x_{13} \oplus k_{13})))] \end{aligned}$$

Table 4: Each effective bit of the Boolean expression of $X_{R,0}^{i+14}$.

x_i	Representation of x_i	k_i	Representation of k_i
x_0	$X_{R,8}^{i+18} \oplus (X_{R,9}^{i+18} \& X_{R,2}^{i+18}) \oplus X_{L,10}^{i+18} \oplus X_{R,12}^{i+18} \oplus X_{R,0}^{i+18}$	k_0	$K_{10}^{i+17} \oplus K_{12}^{i+16} \oplus K_0^{i+16} \oplus K_{14}^{i+15} \oplus K_0^{i+14}$
x_1	$X_{R,9}^{i+18} \oplus (X_{R,10}^{i+18} \& X_{R,3}^{i+18}) \oplus X_{L,11}^{i+18}$	k_1	K_{11}^{i+17}
x_2	$X_{R,2}^{i+18} \oplus (X_{R,3}^{i+18} \& X_{R,12}^{i+18}) \oplus X_{L,4}^{i+18}$	k_2	K_4^{i+17}
x_3	$X_{R,13}^{i+18} \oplus (X_{R,14}^{i+18} \& X_{R,7}^{i+18}) \oplus X_{L,15}^{i+18}$	k_3	K_{15}^{i+17}
x_4	$X_{R,6}^{i+18} \oplus (X_{R,7}^{i+18} \& X_{R,0}^{i+18}) \oplus X_{L,8}^{i+18}$	k_4	K_8^{i+17}
x_5	$X_{R,9}^{i+18} \oplus (X_{R,10}^{i+18} \& X_{R,3}^{i+18}) \oplus X_{L,11}^{i+18} \oplus X_{R,13}^{i+18}$	k_5	$K_{11}^{i+17} \oplus K_{13}^{i+16}$
x_6	$X_{R,10}^{i+18} \oplus (X_{R,11}^{i+18} \& X_{R,4}^{i+18}) \oplus X_{L,12}^{i+18}$	k_6	K_{12}^{i+17}
x_7	$X_{R,3}^{i+18} \oplus (X_{R,4}^{i+18} \& X_{R,13}^{i+18}) \oplus X_{L,5}^{i+18}$	k_7	K_5^{i+17}
x_8	$X_{R,2}^{i+18} \oplus (X_{R,3}^{i+18} \& X_{R,12}^{i+18}) \oplus X_{L,4}^{i+18} \oplus X_{R,6}^{i+18}$	k_8	$K_4^{i+17} \oplus K_6^{i+16}$
x_9	$X_{R,12}^{i+18} \oplus (X_{R,13}^{i+18} \& X_{R,6}^{i+18}) \oplus X_{L,14}^{i+18}$	k_9	K_{14}^{i+17}
x_{10}	$x_3 \oplus x_5$	k_{10}	$k_3 \oplus k_5 \oplus K_{15}^{i+15}$
x_{11}	$X_{R,10}^{i+18} \oplus (X_{R,11}^{i+18} \& X_{R,4}^{i+18}) \oplus X_{L,12}^{i+18} \oplus X_{R,14}^{i+18}$	k_{11}	$K_{12}^{i+17} \oplus K_{14}^{i+16}$
x_{12}	$X_{R,11}^{i+18} \oplus (X_{R,12}^{i+18} \& X_{R,5}^{i+18}) \oplus X_{L,13}^{i+18}$	k_{12}	K_{13}^{i+17}
x_{13}	$X_{R,4}^{i+18} \oplus (X_{R,5}^{i+18} \& X_{R,14}^{i+18}) \oplus X_{L,6}^{i+18}$	k_{13}	K_6^{i+17}
x_{14}	$X_{R,3}^{i+18} \oplus (X_{R,4}^{i+18} \& X_{R,13}^{i+18}) \oplus X_{L,5}^{i+18} \oplus X_{R,7}^{i+18}$	k_{14}	$K_5^{i+17} \oplus K_7^{i+16}$
x_{15}	$x_4 \oplus x_8$	k_{15}	$k_4 \oplus k_8 \oplus K_8^{i+15}$
x_{16}	$X_{R,12}^{i+18} \oplus (X_{R,13}^{i+18} \& X_{R,6}^{i+18}) \oplus X_{L,14}^{i+18} \oplus X_{R,0}^{i+18}$	k_{16}	$K_{14}^{i+17} \oplus K_0^{i+16}$

Table 5: Variants of the Boolean function $f'(x, k')$.

Guess	$x_1 \oplus k_1, x_3 \oplus k_3, x_7 \oplus k_7$	f'
k_1, k_3, k_7	0, 0, 0	f_{000}
	0, 0, 1	f_{001}
	0, 1, 0	f_{010}
	0, 1, 1	f_{011}
	1, 0, 0	f_{100}
	1, 0, 1	f_{101}
	1, 1, 0	f_{110}
	1, 1, 1	f_{111}

$$\& (x_{13,14} \oplus k_{13,14})]) \& (x_{9,15} \oplus k_{9,15} \oplus [(x_{13,14} \oplus k_{13,14}) \& (x_{4,16} \oplus k_{4,16})])\}. \quad (8)$$

Then we create new counters for the next step. For example, if $(x_1 \oplus k_1, x_3 \oplus k_3, x_7 \oplus k_7) = (1, 1, 1)$, f' is equal to f_{111}. Thus, we compress corresponding counters into new counters V_{111}, where

$$V_{111}[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}] \quad (9)$$

is initialized to

$$\sum_{x_1=k_1\oplus 1, x_3=k_3\oplus 1, x_7=k_7\oplus 1, x_2\in\mathbb{F}_2, x_5\in\mathbb{F}_2, x_8\in\mathbb{F}_2} V[x]. \quad (10)$$

Due to $x_{10} = x_3 \oplus x_5$, $x_{6,10}$ is uniquely determined by $x_{5,6}$. Besides, 3-bit information is independent of the value $[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}]$. Consequently, the creation in this example costs $2^9 \times 7$ additions.

Table 6: Variants of the Boolean function f_{111}.

Guess	$x_{5,6} \oplus k_{5,6}, x_{13,14} \oplus k_{13,14}$	f_{111}
$k_{5,6}, k_{13,14}$	0, 0	f_{111}^{00}
	0, 1	f_{111}^{01}
	1, 0	f_{111}^{10}
	1, 1	f_{111}^{11}

(b) For each set of texts and corresponding Boolean function, we compute $B^{k'\setminus\{k_1,k_3,k_7\}}(f')$. We take f_{111} as an example when $(x_1 \oplus k_1, x_3 \oplus k_3, x_7 \oplus k_7) = (1, 1, 1)$.

(1) The next guesses, $k_{5,6}$ and $k_{13,14}$, are constrained by the simplified Boolean function f_{111}. The corresponding texts are split into four sets. The Boolean functions simplified even further are shown in Table 6.

(i) $(x_{5,6} \oplus k_{5,6}, x_{13,14} \oplus k_{13,14}) = (0, 0)$.

The new counters V_{111}^{00} are created. They are given by

$$V_{111}^{00}[x_{0,2,4}, x_{6,10}, x_{9,15}] = \sum_{x_{5,6}=k_{5,6}, x_{13,14}=k_{13,14}} V_{111}[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}]. \quad (11)$$

The creation of new counters takes $2^2 \times (2^5 - 1)$ addition operations. Accordingly, we have

$$f_{111}^{00} = x_{0,2,4} \oplus k_{2,4} \oplus ((x_{6,10} \oplus k_{6,10}) \& (x_{9,15} \oplus k_{9,15})). \quad (12)$$

In Appendix A, the time complexity of computing $B^{k_{2,4},k_{6,10},k_{9,15}}(f_{111}^{00})$ (Case 2 in Appendix A) is estimated. The calculation of $B^{k_{2,4},k_{6,10},k_{9,15}}(f_{111}^{00})$ requires 2^4 additions.

(ii) $(x_{5,6} \oplus k_{5,6}, x_{13,14} \oplus k_{13,14}) = (0, 1)$.

Similarly,

$$V_{111}^{01}[x_{0,2,4}, x_{6,10,11}, x_{12}, x_{13}, x_{4,9,15,16}] = \sum_{x_{5,6}=k_{5,6}, x_{13,14}=k_{13,14}\oplus 1} V_{111}[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}], \quad (13)$$

$$f_{111}^{01} = x_{0,2,4} \oplus k_{2,4} \oplus [(x_{6,10,11} \oplus k_{6,10,11} \oplus ((x_{12} \oplus k_{12}) \& (x_{13} \oplus k_{13}))) \& (x_{4,9,15,16} \oplus k_{4,9,15,16})].$$

The creation of new counters takes $2^5 \times (2^2 - 1) = 2^7 - 2^5$ addition operations. And the calculation of $B^{k_{2,4},k_{6,10,11},k_{12},k_{13},k_{4,9,15,16}}(f_{111}^{01})$ costs $2^{6.75}$ additions.

(iii) $(x_{5,6} \oplus k_{5,6}, x_{13,14} \oplus k_{13,14}) = (1, 0)$.

$$V_{111}^{10}[x_{0,2,4,8,9}, x_{6,10}, x_{9,15}] = \sum_{x_{5,6}=k_{5,6}\oplus 1, x_{13,14}=k_{13,14}} V_{111}[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}], \quad (14)$$

$$f_{111}^{10} = x_{0,2,4,8,9} \oplus k_{2,4,8,9} \oplus (x_{6,10} \oplus k_{6,10}) \& (x_{9,15} \oplus k_{9,15}).$$

The creation of new counters takes $2^2 \times (2^5 - 1) = 2^7 - 2^2$ addition operations. And the calculation of $B^{k_{2,4,8,9},k_{6,10},k_{9,15}}(f_{111}^{01})$ costs 2^4 additions.

(iv) $(x_{5,6} \oplus k_{5,6}, x_{13,14} \oplus k_{13,14}) = (1, 1)$.

$$V_{111}^{11}[x_{0,2,4,8,9}, x_{6,10,11}, x_{12}, x_{13}, x_{4,9,15,16}] = \sum_{x_{5,6}=k_{5,6}\oplus 1, x_{13,14}=k_{13,14}\oplus 1} V_{111}[x_{0,2,4}, x_{5,6}, x_{8,9}, x_{6,10}, x_{11}, x_{12}, x_{13}, x_{13,14}, x_{9,15}, x_{4,16}], \quad (15)$$

$$f_{111}^{11} = x_{0,2,4,8,9} \oplus k_{2,4,8,9} \oplus [(x_{6,10,11} \oplus k_{6,10,11} \oplus ((x_{12} \oplus k_{12}) \& (x_{13} \oplus k_{13}))) \& (x_{4,9,15,16} \oplus k_{4,9,15,16})].$$

The creation of new counters takes $2^5 \times (2^2 - 1) = 2^7 - 2^5$ addition operations. And the calculation of $B^{k_{2,4,8,9},k_{6,10,11},k_{12},k_{13},k_{4,9,15,16}}(f_{111}^{11})$ costs $2^{6.75}$ additions.

(2) After this, we sum the four temporary variables up; namely,

$$B^{k_{2,4},k_{5,6},k_{8,9},k_{6,10},k_{11}-k_{14},k_{9,15},k_{4,16}}(f_{111}) = \left(B^{k_{2,4},k_{6,10},k_{9,15}}\left(f_{111}^{00}\right) + B^{k_{2,4,8,9},k_{6,10},k_{9,15}}\left(f_{111}^{10}\right)\right) + \left(B^{k_{2,4},k_{6,10,11},k_{12},k_{13},k_{4,9,15,16}}\left(f_{111}^{01}\right) + B^{k_{2,4,8,9},k_{6,10,11},k_{12},k_{13},k_{4,9,15,16}}\left(f_{111}^{11}\right)\right). \quad (16)$$

Thus, the time complexity of the summation requires no more than $2^8 \times 3 = 2^{8.58}$ additions, for each $k_{5,6}, k_{13,14}$.

In this example, it takes $2^2 \times ((2^7 - 2^2 + 2^4 + 2^7 - 2^5 + 2^{6.75}) \times 2 + 2^{8.58}) = 2^{12.06}$ additions to compute $B^{k'\setminus\{k_1,k_3,k_7\}}(f')$.

(c) For each k_1, k_3, k_7, we sum the eight temporary variables up. The summation yields a time complexity of $2^{13} \times 7$ addition operations.

Thus, for each x_C, the time complexity of computing $W[k'_P, x_C]$ is approximately $2^{19.87}$ additions. The details are given in Table 7. T_1 denotes the time complexity of creating new counters according to guessed key bits. T_2 denotes the time complexity of computing the sum for each set. T_3 denotes the time complexity of summing them up.

Let us review the procedure proc_simon_32_bit_cond used to compute $Y[k'_P, k'_C]$ and the key-recovery attack on 22-round SIMON32. The procedure is as follows.

(1) For each of $2^{15} x_C$, we compute $W[k'_P, x_C]$.

(2) For each of $2^{16} k'_P$, we compute $Y[k'_P, k'_C]$.

The time complexity of proc_simon_32_bit_cond procedure is $2^{15} \times 2^{19.87} + 2^{16} \times 2^{19.87} = 2^{36.45}$ additions.

The attack works as follows.

(1) Compress the whole plaintext-ciphertext pairs into 2^{30} counters $V[x_P, x_C]$.

(2) Call proc_simon_32_bit_cond.

(3) Check the parity of $Y[k'_P, k'_C]$. If the parity is odd, discard the 32-bit subkey guess. Otherwise, use the key schedule to recover 32 bits of the master key and then exhaustively search for the remaining 32-bit keys.

It is noted that there is one AND operation and three XOR operations in one round of SIMON. In our analysis, we approximate them as four XOR operations. The time complexity of step 1 is 2^{32} compressions, which is equivalent to about $2^{32} \times (104/(4 \times 16 \times 22)) = 2^{28.24}$ encryptions. Since we care about the parity of $Y[k'_P, k'_C]$, all counters can be taken modulo 2. The addition is actually the bitwise XOR operation in the calculation of $Y[k'_P, k'_C]$. Thus, the time complexity of step 2 is equivalent to about $2^{36.45} \times (1/(4 \times 16 \times 22)) = 2^{26}$ encryptions. The time complexity of step 3 is 2^{63} encryptions. Hence, the proposed attack on 22-round SIMON32 requires 2^{32} known plaintexts and has a total time complexity equivalent to about 2^{63} encryptions.

We have implemented the calculation of $Y[k'_P, k'_C]$. The experiment was performed on Intel Core i7-4790 with 8 GBytes of DDR3 memory. The experimental result confirmed the correctness of our technique.

4.2. Integral Attack on 23-Round SIMON32. In this section, we extend the 22-round attack by one round. The improved attack is as follows. Guess 13 bits' subkey k_α and partially encrypt plaintexts, where $k_\alpha = K_1^{i-5} - K_6^{i-5} \parallel K_8^{i-5} - K_{13}^{i-5} \parallel K_{15}^{i-5}$. Then carry out the 22-round attack.

We briefly explain why there is no need to guess K_7^{i-5}. Let the first bit of x_P (resp., k_P) be $x_{P,0}$ (resp., $k_{P,0}$). In our attack,

Table 7: Time complexity of calculating $W[k'_P, x_C]$ with a fixed x_C.

Guess	$x_1 \oplus k_1, x_3 \oplus k_3, x_7 \oplus k_7$	f'	Time		
			T_1	T_2	T_3
k_1, k_3, k_7	0, 0, 0	f_{000}	$2^9 \times 7$	$2^{12.06}$	$2^{13} \times 7$
	0, 0, 1	f_{001}	$2^9 \times 7$	$2^{12.06}$	
	0, 1, 0	f_{010}	$2^9 \times 7$	$2^{12.06}$	
	0, 1, 1	f_{011}	$2^9 \times 7$	$2^{12.06}$	
	1, 0, 0	f_{100}	$2^9 \times 7$	$2^{12.06}$	
	1, 0, 1	f_{101}	$2^9 \times 7$	$2^{12.06}$	
	1, 1, 0	f_{110}	$2^9 \times 7$	$2^{12.06}$	
	1, 1, 1	f_{111}	$2^9 \times 7$	$2^{12.06}$	
Total time			$((2^9 \times 7 + 2^{12.06}) \times 8 + 2^{13} \times 7) \times 2^3 = 2^{19.87}$		

we make the redefinitions, $x_{P,0} = K_7^{i-5} \oplus X_{L,7}^{i-4} \oplus (X_{L,8}^{i-4} \& X_{L,1}^{i-4}) \oplus X_{R,9}^{i-4} \oplus X_{L,11}^{i-4} \oplus X_{L,15}^{i-4}$ and $k_{P,0} = K_7^{i-5} \oplus K_9^{i-4} \oplus K_{11}^{i-3} \oplus K_{15}^{i-3} \oplus K_{13}^{i-2} \oplus K_{15}^{i-1}$. It is evident that $K_7^{i-5} \oplus X_{L,7}^{i-4} \oplus (X_{L,8}^{i-4} \& X_{L,1}^{i-4}) \oplus X_{R,9}^{i-4} \oplus X_{L,11}^{i-4} \oplus X_{L,15}^{i-4}$ can be obtained after guessing 13 bits' subkey k_α. Consequently, we still have $f'_E(x_P, k'_P) = 1$.

The 23-round attack has a data complexity of 2^{32} known plaintexts and a time complexity of about 2^{63} encryptions.

4.3. Integral Attack on 24-Round SIMON32. The 22-round attack can be extended by one round in forward and one round in backward direction in a straightforward way. The improved attack proceeds as follows. Guess 26 bits subkey $k_\alpha \parallel k_\beta$, where $k_\beta = K_0^{i+18} \parallel K_2^{i+18} - K_7^{i+18} \parallel K_9^{i+18} - K_{14}^{i+18}$. Partially encrypt plaintexts and partially decrypt corresponding ciphertexts. Then carry out the 22-round attack presented above. It should be noted that we do not guess K_8^{i+18}. The reason is essentially the same as the case mentioned above.

In this attack, the dominant part of the time complexity is still exhaustively searching half of the key space. The total time complexity of our attack is about 2^{63} encryptions. The number of the required known plaintexts is 2^{32}. The success probability of our attack is 100%.

The total memory complexity of our attack is determined by the size of the entire SIMON32 codebook, $V[x_P, x_C]$ and $W[k'_P, x_C]$. This corresponds to a memory requirement of about $2^{33.64}$ bytes. Note that we can only store $F(X_R^{i+19})_0 \oplus X_{L,0}^{i+19} \parallel (F(X_R^{i+19}) \oplus X_L^{i+19})_2 - (F(X_R^{i+19}) \oplus X_L^{i+19})_{14} \parallel X_{R,4}^{i+19} - X_{R,6}^{i+19} \parallel X_{R,8}^{i+19} \parallel X_{R,10}^{i+19} - X_{R,15}^{i+19}$ for each ciphertext. In addition, there is no need to store $Y[k'_P, k'_C]$. The elements of it can be computed on-the-fly. As soon as a value of $Y[k'_P, k'_C]$ is computed, the bit condition is checked. If the condition is satisfied, then exhaustively search for the remaining 6-bit key.

4.4. Improved Integral Attacks on SIMON48. We can improve the integral attacks on SIMON48/72 and SIMON48/96, using dynamic key-guessing techniques. Since the attack procedures for them are similar, we present these integral attacks in Appendix B. The results are summarized in Table 1.

5. Conclusion

In this paper, dynamic key-guessing techniques are first introduced in integral cryptanalysis, and we extend dynamic key-guessing techniques to fit our needs. Dynamic key-guessing techniques significantly improve the complexity of calculating $\sum_x f(x, k) \cdot V[x]$. Using dynamic key-guessing techniques, we can attack two more rounds than previously known integral attacks on SIMON32 and SIMON48.

Appendix

A. Time Complexities under Some Variations of Boolean Functions

In this section, we estimate the time complexity of calculating $B^k(f)$, which is defined as $\sum_x f(x, k) \cdot V[x]$, under some variations of Boolean functions. Let Guess denote the bit guessed at first. Let $x_i \oplus k_i$ denote the set of texts, where the value of x_i is the same. Let f_i be the simplified Boolean function after guessing. In addition, the form of f_i is the same as that of f_i*. And the new counters are created with a time complexity of T_1 additions. Computing the sum for each set costs T_2 addition operations. It takes T_3 additions to sum all of them up. Moreover, total time denotes the overall time complexity.

All the cases are similar, so we focus on Case 3 in the following. f_3 is a Boolean function of 5-bit x and 4-bit k, where $x = x_0 \parallel \cdots \parallel x_4$ and $k = k_1 \parallel \cdots \parallel k_4$. First, we guess k_1 and the texts are split into two sets. One set contains the texts where $x_1 = k_1$, and the other contains the texts where $x_1 = k_1 \oplus 1$. Obviously, we obtain the corresponding simplified Boolean functions, x_0 and $x_{0,2} \oplus k_2 \oplus (x_3 \oplus k_3) \& (x_4 \oplus k_4)$. The new counters can be created according to simplified Boolean functions. Note that when $x_1 = k_1$, we only need to compute $V_1[x_0 = 1] = \sum_{x_0=1, x_1=k_1} V[x]$. Next, we try to compute the corresponding temporary variables. When $x_1 = k_1 \oplus 1$, we obtain $B^{k_2,k_3,k_4}(x_{0,2} \oplus k_2 \oplus (x_3 \oplus k_3) \& (x_4 \oplus k_4))$, referring to Case 2. And when $x_1 = k_1$, $B(x_0) = V_1[x_0 = 1]$. Finally, we sum them up and get $B^k(f)$.

Table 8: Time complexity of the calculation f_1.

Guess	$x_1 \oplus k_1$	f_1	T_1	T_2	T_3
k_1	0	0	1	0	2
	1	0	2	0	
Total time		$2 \times (1 + 2 + 2) = 10$			

Table 9: Time complexity of the calculation f_3.

Guess	$x_1 \oplus k_1$	f_3	T_1	T_2	T_3
k_1	0	0	$2^3 - 1$	0	2^3
	1	f_2*	$2^3 \times 1$	2^4	
Total time		$2 \times (2^3 - 1 + 2^3 + 2^4 + 2^3) = 2^{6.29}$			

Table 10: Time complexity of the calculation f_2.

Guess	$x_1 \oplus k_1$	f_2	T_1	T_2	T_3
k_1	0	0	2×1	0	2^2
	1	0	2×1	0	
Total time		$2 \times (2 + 2 + 2^2) = 2^4$			

Table 11: Time complexity of the calculation f_4.

Guess	$x_1 \oplus k_1$	f_4	T_1	T_2	T_3
k_1	0	0	$2 \times (2^3 - 1)$	0	2^4
	1	f_2*	$2^3 \times 1$	2^4	
Total time		$2 \times (2 \times (2^3 - 1) + 2^3 + 2^4 + 2^4) = 2^{6.75}$			

Case 1. $f_1 = x_0 \oplus ((x_1 \oplus k_1)\&(x_2 \oplus k_2))$ (see Table 8).

Case 2. $f_2 = x_0 \oplus k_0 \oplus ((x_1 \oplus k_1)\&(x_2 \oplus k_2))$ (see Table 10).

Case 3. $f_3 = x_0 \oplus (x_1 \oplus k_1)\&((x_2 \oplus k_2) \oplus (x_3 \oplus k_3)\&(x_4 \oplus k_4))$ (see Table 9).

Case 4. $f_4 = x_0 \oplus k_0 \oplus (x_1 \oplus k_1)\&((x_2 \oplus k_2) \oplus (x_3 \oplus k_3)\&(x_4 \oplus k_4))$ (see Table 11).

B. Improved Integral Attacks on SIMON48

B.1. Integral Attack on 23-Round SIMON48. Using the 15-round integral characteristic, we provide a key-recovery attack (procedure proc_attack_simon_48) over four rounds of partial encryption and four rounds of partial decryption. The constant bit is $X^{1}_{L,23}$ and we take the balanced bit $X^{i+15}_{R,23}$ as the bit condition. It is obvious that the Boolean expressions of $X^{i}_{L,23}$ and $X^{i+15}_{R,23}$ have the same general form, as follows.

$$\begin{aligned} f(x,k) = {} & x_0 \oplus k_0 \oplus ((x_1 \oplus k_1)\,\&\,(x_2 \oplus k_2)) \oplus ((x_3 \\ & \oplus k_3)\,\&\,(x_4 \oplus k_4)) \oplus [(x_5 \oplus k_5 \oplus ((x_6 \oplus k_6) \\ & \&\,(x_7 \oplus k_7)))\,\&\,(x_8 \oplus k_8 \oplus ((x_9 \oplus k_9) \\ & \&\,(x_7 \oplus k_7)))] \oplus \{(x_{10} \oplus k_{10} \oplus ((x_6 \oplus k_6) \\ & \&\,(x_7 \oplus k_7)) \\ & \oplus [(x_{11} \oplus k_{11} \oplus ((x_{12} \oplus k_{12})\,\&\,(x_{13} \oplus k_{13}))) \\ & \&\,(x_{14} \oplus k_{14} \oplus ((x_{17} \oplus k_{17})\,\&\,(x_{13} \oplus k_{13})))]) \\ & \&\,(x_{15} \oplus k_{15} \oplus ((x_7 \oplus k_7)\,\&\,(x_9 \oplus k_9)) \\ & \oplus [(x_{14} \oplus k_{14} \oplus ((x_{13} \oplus k_{13})\,\&\,(x_{17} \oplus k_{17}))) \\ & \&\,(x_{16} \oplus k_{16} \oplus ((x_{17} \oplus k_{17})\,\&\,(x_{18} \oplus k_{18})))])\}. \end{aligned} \tag{B.1}$$

In the above function, $x_1 \oplus x_5 = x_{10}$ and $x_2 \oplus x_8 = x_{15}$.

Procedure proc_simon_48_comp_w/proc_simon_48_comp_y

(1) Guess k_1, k_3 and create corresponding counters.

(2) When $x_1 \oplus k_1 = 1$ and $x_3 \oplus k_3 = 1$, guess k_7, k_{17} and create corresponding counters. The other situations can be treated in a similar way.

(3) When $x_1 \oplus k_1 = 1$, $x_3 \oplus k_3 = 1$, $x_7 \oplus k_7 = 1$, and $x_{17} \oplus k_{17} = 1$, guess $k_{8,9}, k_{13,14}$ and create corresponding counters. The other situations can be treated in a similar way.

(4) Compute temporary variables and sum them up.

Time complexity evaluation: Steps (3)-(4) cost $2^2 \times [(2^7 - 2^5 + 2^{6.75}) \times 4 + 2^8 \times 3] = 2^{12.63}$ bitwise XOR operations; Steps (2)–(4) cost $2^2 \times [(2^9 \times 7 + 2^{12.63}) \times 4 + 2^{13} \times 3] = 2^{17.97}$ bitwise XOR operations; Steps (1)–(4) cost $2^2 \times [(2^{14} + 2^{17.97}) \times 4 + 2^{16} \times 3] = 2^{22.29}$ bitwise XOR operations.

Procedure proc_simon_48_bit_cond

(1) For each of $2^{17} x_C$, call proc_simon_48_comp_w.

(2) For each of $2^{18} k'_P$, call proc_simon_48_comp_y.

Time complexity evaluation: $2^{17} \times 2^{22.29} + 2^{18} \times 2^{22.29} = 2^{40.87}$ bitwise XOR operations.

Procedure proc_attack_simon_48

(1) Compress the whole plaintext-ciphertext pairs into 2^{34} counters.

(2) Call proc_simon_48_bit_cond.

(3) Check the bit condition. If the condition is satisfied, use the key schedule to recover 36 bits of the master key; then exhaustively search for the remaining key bits. Otherwise, discard the 36-bit subkey guess.

Complexity evaluation includes 2^{48} known plaintexts, $2^{71}/2^{95}$ encryptions, and 2^{33} bytes.

In the key-recovery attack (procedure proc_attack_simon_48), the whole plaintext-ciphertext pairs are compressed into counters. Thus, the memory complexity of our attack is only determined by the size of counters used in the attack. This corresponds to a memory requirement of about 2^{33} bytes. Note that there is no need to store $Y[k'_P, k'_C]$, since we can compute the elements of $Y[k'_P, k'_C]$ on-the-fly, similar to the integral attack on 24-round SIMON32.

B.2. Integral Attack on 24-Round SIMON48/72

Procedure proc_attack_simon_48_72_24

(1) Guess 18 bits' subkey k_α and partially encrypt plaintexts, where $k_\alpha = K_3^{i-5} - K_5^{i-5} \parallel K_7^{i-5} \parallel K_9^{i-5} - K_{22}^{i-5}$.

(2) Call proc_attack_simon_48.

Complexity evaluation includes 2^{48} known plaintexts, 2^{71} encryptions, and 2^{50} bytes.

In this attack, the dominant part of the memory complexity is the size of the entire SIMON48 codebook. This corresponds to a memory requirement of about 2^{50} bytes. It is noted that we can only store $(F(X_R^{i+20}) \oplus X_L^{i+20})_3 - (F(X_R^{i+20}) \oplus X_L^{i+20})_5 \parallel F(X_R^{i+20})_7 \oplus X_{L,7}^{i+20} \parallel (F(X_R^{i+20}) \oplus X_L^{i+20})_9 - (F(X_R^{i+20}) \oplus X_L^{i+20})_{23} \parallel X_{R,5}^{i+20} \parallel X_{R,6}^{i+20} \parallel X_{R,11}^{i+20} - X_{R,13}^{i+20} \parallel X_{R,15}^{i+20} \parallel X_{R,17}^{i+20} - X_{R,20}^{i+20} \parallel X_{R,22}^{i+20} \parallel X_{R,23}^{i+20}$ for each ciphertext.

B.3. Integral Attack on 25-Round SIMON48/96

Procedure proc_attack_simon_48_96_25

(1) Guess 36 bits' subkey $k_\alpha \parallel k_\beta$, where $k_\alpha = K_3^{i-5} - K_5^{i-5} \parallel K_7^{i-5} \parallel K_9^{i-5} - K_{22}^{i-5}$ and $k_\beta = K_3^{i+19} - K_5^{i+19} \parallel K_7^{i+19} \parallel K_9^{i+19} - K_{22}^{i+19}$. Partially encrypt plaintexts and partially decrypt corresponding ciphertexts.

(2) Call proc_attack_simon_48.

Complexity evaluation includes 2^{48} known plaintexts, 2^{95} encryptions, and 2^{50} bytes.

Similar to the integral attack on 24-round SIMON48/72, the dominant part of the memory complexity is the size of the entire SIMON48 codebook.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work is supported by China's 973 Program (no. 2013CB834205).

References

[1] J. Daemen, L. R. Knudsen, and V. Rijmen, "The block cipher square," in *Fse*, vol. 97, pp. 149–165, Springer, 1997.

[2] B. Ray, S. Douglas, S. Jason, T. Stefan, W. Bryan, and W. Louis, "The simon and speck families of lightweight block ciphers," Cryptology ePrint Archive Report 2013/404, 2013.

[3] M. A. Abdelraheem, J. Alizadeh, H. A. Alkhzaimi, M. R. Aref, N. Bagheri, and P. Gauravaram, "Improved linear cryptanalysis of reduced-round simon-32 and simon-48," in *Proceedings of the International Conference in Cryptology in India*, vol. 9462 of *Lecture Notes in Comput. Sci.*, pp. 153–179, Springer, Cham, Germany, December 2015.

[4] P. Ahir, M. Mozaffari-Kermani, and R. Azarderakhsh, "Lightweight architectures for reliable and fault detection Simon and Speck cryptographic algorithms on FPGA," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 4, article no. 109, 2017.

[5] A. Aysu, E. Gulcan, and P. Schaumont, "SIMON says: Break area records of block ciphers on FPGAs," *IEEE Embedded Systems Letters*, vol. 6, no. 2, pp. 37–40, 2014.

[6] H. Chen and X. Wang, "Improved linear hull attack on round-reduced SIMON with dynamic key-guessing techniques," in *International Conference on Fast Software Encryption*, vol. 9783, pp. 428–449, Springer, Berlin, Heidelberg, 2016.

[7] K. Fu, L. Sun, and M. Wang, "New integral attacks on SIMON," *IET Information Security*, vol. 11, no. 5, pp. 277–286, 2017.

[8] Y. Hao and W. Meier, "Truncated differential based known-key attacks on round-reduced simon," *Designs, Codes and Cryptography*, vol. 83, no. 2, pp. 467–492, 2017.

[9] S. Kölbl, G. Leander, and T. Tiessen, "Observations on the simon block cipher family," in *Annual Cryptology Conference*, vol. 9215 of *Lecture Notes in Comput. Sci.*, pp. 161–185, Springer, Heidelberg, Germany, 2015.

[10] K. Kondo, Y. Sasaki, and T. Iwata, "On the design rationale of SIMON block cipher: Integral attacks and impossible differential attacks against SIMON variants," in *International Conference on Applied Cryptography and Network Security*, vol. 9696, pp. 518–536, Springer, Cham, Switzerland, 2016.

[11] K. Kondo, Y. Sasaki, Y. Todo, and T. Iwata, "Analyzing key schedule of SIMON: Iterative key differences and application to related-key impossible differentials," in *International Workshop on Security*, vol. 10418, pp. 141–158, Springer, Cham, Switzerland, 2017.

[12] R. Nithya and D. S. Kumar, "Where aes is for internet, simon could be for iot," *Procedia Technology*, vol. 25, pp. 302–309, 2016.

[13] K. Qiao, L. Hu, and S. Sun, "Differential analysis on simeck and simon with dynamic key-guessing techniques," in *Proceedings of the International Conference on Information Systems Security and Privacy*, pp. 64–85, Springer, 2016.

[14] A. Shahverdi, M. Taha, and T. Eisenbarth, "Lightweight Side Channel Resistance: Threshold Implementations of Simon," *IEEE Transactions on Computers*, vol. 66, no. 4, pp. 661–671, 2017.

[15] D. Shi, L. Hu, S. Sun, L. Song, K. Qiao, and X. Ma, "Improved linear (hull) cryptanalysis of round-reduced versions of SIMON," *Science China Information Sciences*, vol. 60, no. 3, Article ID 39101, 2017.

[16] P. Suil, P. Sepehrdad, S. Vaudenay, N. Courtois, and P. Sušil, "On selection of samples in algebraic attacks and a new technique to find hidden low degree equations," *International Journal of Information Security*, vol. 15, no. 1, pp. 51–65, 2016.

[17] Y. Todo, "Structural evaluation by generalized integral property," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 287–314, Springer, 2015.

[18] Y. Todo and M. Morii, "Bit-based division property and application to SIMON family," in *International Conference on Fast Software Encryption*, vol. 9783, pp. 357–377, Springer, Berlin, Heidelberg, 2016.

[19] G. Wang, N. Gan, and Y. Li, "Improved differential attack on 30-round simon64," *Wuhan University Journal of Natural Sciences*, vol. 21, no. 1, pp. 75–83, 2016.

[20] N. Wang, X. Wang, K. Jia, and J. Zhao, "Differential attacks on reduced simon versions with dynamic key-guessing techniques," Cryptology ePrint Archive Report 2014/448, 2014.

[21] Q. Wang, Z. Liu, K. Varıcı, Y. Sasaki, V. Rijmen, and Y. Todo, "Cryptanalysis of reduced-round SIMON32 and SIMON48," in *International Conference in Cryptology in India*, vol. 8885, pp. 143–160, Springer, Cham, Switzerland, 2014.

[22] Z. Xiang, W. Zhang, Z. Bao, and D. Lin, "Applying milp method to searching integral distinguishers based on division property for 6 lightweight block ciphers," in *Advances in Cryptology–ASIACRYPT*, vol. 10031 of *Lecture Notes in Comput. Sci.*, pp. 648–678, Springer, Berlin, Germany, 2016.

[23] Z. Xiang, W. Zhang, and D. Lin, "On the division property of SIMON48 and SIMON64," in *International Workshop on Security*, vol. 9836, pp. 147–163, Springer, Cham, Switzerland, 2016.

[24] X.-L. Yu, W.-L. Wu, Z.-Q. Shi, J. Zhang, L. Zhang, and Y.-F. Wang, "Zero-correlation linear cryptanalysis of reduced-round simon," *Journal of Computer Science and Technology*, vol. 30, no. 6, pp. 1358–1369, 2015.

[25] P. Zajac, "Upper bounds on the complexity of algebraic cryptanalysis of ciphers with a low multiplicative complexity," *Designs, Codes and Cryptography. An International Journal*, vol. 82, no. 1-2, pp. 43–56, 2017.

[26] G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, "The simeck family of lightweight block ciphers," in *International Workshop on Cryptographic Hardware and Embedded Systems*, vol. 9293, pp. 307–329, Springer, Berlin, Heidelberg, 2015.

[27] L. Qin, H. Chen, and X. Wang, "Linear hull attack on round-reduced simeck with dynamic key-guessing techniques," in *Australasian Conference on Information Security and Privacy*, vol. 9723, pp. 409–424, Springer, Cham, Switzerland, 2016.