PAPER

Probabilistic Analysis of Differential Fault Attack on MIBS*

Yang GAO[†a)], **Yong-juan WANG**[†b)], ***Nonmembers***, **Qing-jun YUAN**[†c)], ***Member***, **Tao WANG**[†d)], ***and*** **Xiang-bin WANG**[†e)], ***Nonmembers***

SUMMARY We propose a new method of differential fault attack, which is based on the nibble-group differential diffusion property of the lightweight block cipher MIBS. On the basis of the statistical regularity of differential distribution of the S-box, we establish a statistical model and then analyze the relationship between the number of faults injections, the probability of attack success, and key recovering bits. Theoretically, time complexity of recovering the main key reduces to 2^2 when injecting 3 groups of faults (12 nibbles in total) in 30,31 and 32 rounds, which is the optimal condition. Furthermore, we calculate the expectation of the number of fault injection groups needed to recover 62 bits in main key, which is 3.87. Finally, experimental data verifies the correctness of the theoretical model.

key words: *lightweight block cipher, MIBS algorithm, differential fault attack, probabilistic model*

1. Introduction

With the advancement of network technology, the demand for people to communicate on the Internet is increasing, and cryptology has also been greatly developed. Considering the encryption environment and computing resources in real life, in order to achieve the standard of efficiency and security of the encryption algorithm, lightweight ciphers have attracted a lot of attention in recent years. For example, LBlock [1], Piccolo [2], LED [3], MIBS [4], HIGHT [5], PRESENT [6] and so on.

The MIBS [7] algorithm is a lightweight block cipher based on Feistel structure, which supports 64-bit and 80-bit key lengths, denoted as MIBS-64 and MIBS-80 respectively. Owing to required cost of hardware is only 1396GE(Gate Equivalent circuit) and 1530GE, MIBS is fully capable of being applied to micro-computing devices [7]. Researchers have used traditional methods to analyze MIBS algorithm since proposed due to the wide application prospects of this algorithm. So far, the analysis methods for MIBS include differential analysis [8], impossible differential analysis [9], linear analysis [10], [11], zero-correlation analysis [12], and integral analysis [13].

On the other hand, in 1996, fault attack was proposed by Boneh et al.[14] to analysis RSA signature algorithms implemented in CRT mode. In the following year, after the improvement of Biham in [15], the well-known differential fault attack(DFA) came into being. They successfully analyzed the block cipher DES algorithm in this method. The attack principle is injecting some faults in certain rounds in the encryption process. Then we can get the set of possible round keys from the fault cipher and differential equations. Finally, repeat the steps and filter the possible round keys then recover main key by key expansion algorithm. Differential fault attack combines side-channel attack with traditional cryptanalysis thoughts, which has a significant effect on hardware-based lightweight ciphers. After nearly two decades of development, researchers constantly put forward new methods of DFA, successfully analyzing SMS4[16], PRESENT [17], Keeloq [18], Camellia [19] and other lightweight ciphers.

Related work: Despite the fact that various analyses exist for MIBS, DFA on MIBS is rather limited. In 2011, Wang, Zhao, Wang *et al.* presented the first DFA on MIBS [20]. With 8 nibble faults injected at the 31st round and the 32nd round, almost 43 bits of the main key can be recovered. In 2016, Wang and Yan extended the fault injection to the round key [21]. Under the key schedule fault model, their attack only works when faults are injected at the round key in the 29th round to the 31st round. In 2018, Wang, Zhang, Wang *et al.* also proposed a fault attack when fixed faults are injected at the 31st round and the 32nd round [22]. Table 1 illustrates the previous DFA under different fault model on MIBS.

Table 1 Results of DFA on MIBS

Fault model	Nibble fault injection number	Complexity	Reference
Random fault model	16	$2^{21.7}$	[20]
Key schedule fault model	Approximately 10	2^2	[21]
Fixed fault model	32	2^{17}	[22]
Random fault model	Approximately 15.5	2^2	This paper

Manuscript received May 14, 2018.
Manuscript revised October 9, 2018.
Manuscript publicized November 16, 2018.

[†]The authors are with the State Key Laboratory for Mathematical Engineering and advanced computing, Zhengzhou, 450000 China.

*This work is supported in part by National Nature Science Foundation of China (Grant No. 61402522).

a) E-mail: gaoyang_1279@126.com
b) E-mail: pinkywyj@163.com
c) E-mail: gcxyuan@outlook.com
d) E-mail: wt107263@163.com
e) E-mail: wang_moony@163.com

DOI: 10.1587/transinf.2018EDP7168

Our contribution: In this paper, we analyze and study the DFA on MIBS-64. The main achievements and innovations are as follows:

a). We propose a new idea of fault injection based on the differential diffusion characteristics of MIBS: Totally 3 groups of faults need to be injected in the 30th, 31st, and 32nd rounds. In other words, 62 bits of key can be recovered with at least 12 faults injections, which is a great improvement over the width of fault injections and the success rate of recovering main key in [20] and [22].

b). On the basis of breaking MIBS, we introduce the probabilistic analysis method of DFA. By the lower bound of recovering round key we could derive the expectation of number of faults injection, which has certain guiding significance for the theoretical research of DFA and the hardware and software implementation of MIBS algorithm.

2. Preliminaries

2.1 Symbol Description

For the sake of follow-up discussion, Table 2 give the symbols and their corresponding meanings that will appear in this paper.

Table 2 Symbol description

Symbol	Description
x_i^r	4 bits input at position i in the r-th round
K	The main key
k_i	4 bits of key at position i
W_k^r	64 bits key generated by the main key in the r-th round
R_k^r	32 bits round key in the r-th round
$R_{k_i}^r$	4 bits key at position i in the r-th round
C_i	4 bits correct ciphertext at position i
C_i^*	4 bits fault ciphertext at position i
ΔC_i^*	Difference between the correct and wrong ciphertext at position i
f_i	4 bits fault at position i
$\#S$	Number of elements included in set S
p_l^r	Probability of recovering the r-th round key after injecting l group of faults when $k_i = 0x0000$

2.2 Introduction to MIBS Algorithm

MIBS is a lightweight block cipher algorithm proposed by M.Izadi in CANS2009. It adopts Feistel structure and has a 64-bit block size, which supports 64-bit and 80-bit key lengths, respectively referred to as MIBS-64 and MIBS-80 accordingly. And the number of iteration round is 32. All iterative operations in MIBS are based on nibbles(4 bits). The round function F of MIBS is SPN structured, including subkey XORs, S-boxes(4×4) and a linear layer P(branch number of which is 5). A round of MIBS is shown in Fig. 1:

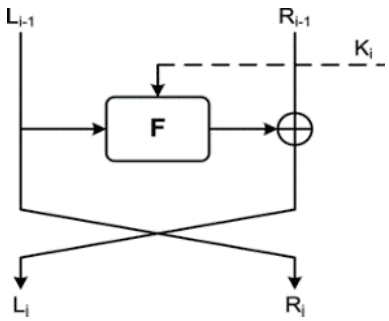


Fig. 1 Algorithm structure of MIBS

We know from Fig. 1 that round function of MIBS encryption is:

$$L_i = R_{i-1} \oplus F(L_{i-1}, K_i), R_i = L_{i-1}.$$

The round function $F(L_{i-1}, K_i)$ includes round key addition, S-box conversion, and mixed layer P (including linear confusion and byte permutation). The mixed layer P can be described as the following linear transformation and the values of S-box are shown in Table 3.

Table 3 S-box of MIBS

α	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(\alpha)$	4	F	3	8	D	A	C	0	B	5	7	E	2	6	1	9

$$\begin{aligned}
y'_0 &= y_0 \oplus y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \\
y'_1 &= y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \\
y'_2 &= y_0 \oplus y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7 \\
y'_3 &= y_1 \oplus y_2 \oplus y_3 \oplus y_6 \oplus y_7 \\
y'_4 &= y_0 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_7 \\
y'_5 &= y_0 \oplus y_1 \oplus y_3 \oplus y_4 \oplus y_5 \\
y'_6 &= y_0 \oplus y_1 \oplus y_2 \oplus y_5 \oplus y_6 \\
y'_7 &= y_0 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_7
\end{aligned}$$

The key expansion schemes of MIBS-64 and MIBS-80 are slightly distinct. Without loss of the generality, we only study the MIBS-64 algorithm. The 32 rounds of extended keys are generated by the following iteration function via the main key $K(k_0, k_1, \ldots, k_{63})$.

W_k^0=The main key
For(i=1;i<=32;i++)
{
$W_k^i = W_k^{i-1} >>> 15$
$W_k^i = S(W_k^i[63:60]) \| W_k^i[59:0]$
$W_k^i = W_k^i[63:16] \| W_k^i[15:11] \oplus (i-1) \| W_k^i[10:0]$
$R_k^i = W_k^i[63:32]$
}

3. Structure Properties of MIBS

3.1 Fault Diffusion Property of MIBS

First, we analyze the fault diffusion rule of MIBS by two consecutive rounds of inputs. Suppose inputs of the $r-1$ th round are $x_0^{r-1}, x_1^{r-1}, x_2^{r-1}, \ldots, x_{15}^{r-1}$, it is found that the fault injection at x_0^{r-1} can be diffused into $x_0^r, x_2^r, x_4^r, x_5^r, x_6^r, x_7^r$ in the r-th round. The process of faults diffusion in two rounds is shown in Fig. 2.

Furthermore, we inject nibble faults at each position in $x_0^r, x_1^r, x_2^r, \ldots, x_7^r$, then index all diffusion positions, as shown in Table 4.

3.2 Differential Distribution of S-Box in MIBS

In MIBS, both of the nonlinear transformation in round function and the complexity of the key expansion algorithm are based on the same 4-bit S-box. Accordingly, it is necessary to study the differential distribution properties of the S-box. Given $\alpha \in F_2^4, \beta \in F_2^4, m \in F_2^4$, satisfying $S(m \oplus \alpha) \oplus S(m) = \beta$, we call α the S-box input difference and β the S-box output difference. When α is fixed, the correspondence between β and input value m is shown in Table 5.

From S-box differential distribution table above, we can generalize the following properties:

Property 1: When m is fixed, there are 15 possible "input-output" differential pairs (α, β). Furthermore, α and β traverse from 0x1 to 0xF respectively.

Property 2: When $m \neq 0$, for 15 corresponding sets of (α, β), there are 3 sets of possible input values are recorded as $Y_i (i = 1, 2, 3)$, where $\#Y_i = 4$. And the remaining 12 sets of possible input values are recorded as sets $Z_i (i = 1, \ldots, 12)$, where $\#Z_i = 2$.

Property 3: $Y_1 = Y_2 = Y_3, Z_i \cap Z_j = m, Y_i \cap Z_j = m, (i = 1, 2, 3; j = 1, \ldots, 12)$.

Property 4: When $m = 0$, for 15 (α, β), the corresponding sets of possible input values are recorded as $P_i (i = 1, 2, \ldots, 15)$, where $\#P_i = 4$.

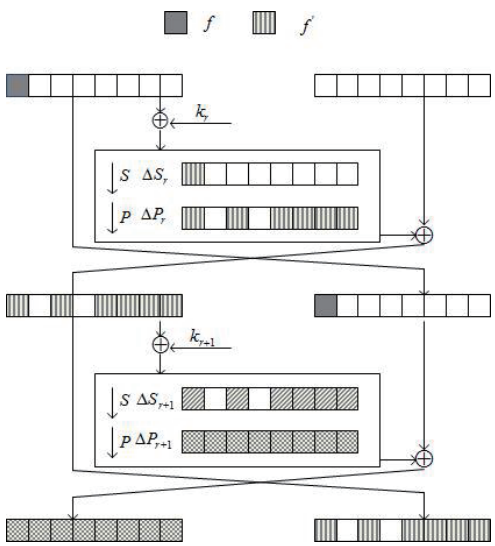


Fig. 2 Fault diffusion property in MIBS

Table 4 Fault diffusion index

Fault injection positions	Diffusion positions
x_0^{r-1}	$x_0^r, x_2^r, x_4^r, x_5^r, x_6^r, x_7^r$
x_1^{r-1}	$x_0^r, x_1^r, x_2^r, x_3^r, x_5^r, x_6^r$
x_2^{r-1}	$x_1^r, x_2^r, x_3^r, x_4^r, x_6^r, x_7^r$
x_3^{r-1}	$x_0^r, x_1^r, x_3^r, x_4^r, x_5^r, x_7^r$
x_4^{r-1}	$x_0^r, x_1^r, x_2^r, x_4^r, x_5^r$
x_5^{r-1}	$x_1^r, x_2^r, x_5^r, x_6^r, x_7^r$
x_6^{r-1}	$x_0^r, x_1^r, x_3^r, x_6^r, x_7^r$
x_7^{r-1}	$x_0^r, x_2^r, x_3^r, x_6^r, x_7^r$

Table 5 Differential distribution of S-box in MIBS

α		Relationship between β and m when α is fixed						
1	β	4	7	8	9	B	C	E
	m	C,D	4,5	E,F	A,B	0,1,2,3	6,7	8,9
2	β	1	3	7	A	B	C	F
	m	4,6	C,E	0,1,2,3	5,7	9,B	8,A	D,F
3	β	2	5	6	7	B	C	D
	m	9,A	8,B	5,6	D,E	C,F	0,1,2,3	4,7
4	β	3	5	6	7	8	9	F
	m	9,D	1,5	A,E	B,F	3,7	0,4,8,C	2,6
5	β	2	3	4	7	D	E	F
	m	1,4	2,7	3,6	9,C	8,D	0,5,A,F	B,E
6	β	2	5	8	A	C	E	F
	m	3,5	A,C	0,6,B,D	8,E	9,F	2,4	1,7
7	β	1	2	3	4	5	9	C
	m	A,D	8,F	1,6	0,7,9,E	3,4	2,5	B,C
8	β	4	6	9	A	C	D	F
	m	2,A	3,B	7,F	1,9	5,D	6,E	0,4,8,C
9	β	1	4	5	8	B	D	F
	m	0,7,9,E	1,8	6,F	5,C	4,D	2,B	3,A
A	β	1	3	6	8	C	D	E
	m	1,B	0,5,A,F	7,D	2,8	4,E	3,9	6,C
B	β	2	3	4	6	8	A	B
	m	7,C	3,8	4,F	2,9	1,A	0,6,B,D	5,E
C	β	1	2	6	9	B	E	F
	m	3,F	2,E	0,4,8,C	1,D	6,A	7,B	5,9
D	β	1	2	7	8	9	A	B
	m	5,8	0,6,B,D	7,A	4,9	3,E	2,F	1,C
E	β	1	4	5	6	7	A	E
	m	2,C	5,B	0,7,9,E	1,F	6,8	4,A	3,D
F	β	3	5	9	A	B	D	E
	m	4,B	2,D	6,9	3,C	7,8	0,5,A,F	1,E

4. DFA on MIBS Algorithm

Before the DFA process, it's necessary to select suitable fault injection method, which determines the specific attack assumptions. Fault injection is usually executed by three means: clock glitch [23], voltage glitch [24] and laser [25].

Clock glitch: The principle is that the register is prematurely taken by the rising edge of the clock, and it often stores the intermediate value of the circuit operation process. The error value is often a (pseudo) random number, so it can be regarded as a random fault injection.

Voltage glitch: Because the voltage is low at a certain moment, the combined circuit operation (transmission) speed is slow. When the normal clock rising edge comes, the circuit operation is not finished and the random result is taken. So this error type is the same as the clock glitch, also a random fault injection.

Laser: When laser is applied to the same register in the same chip. Some bit positions can always be 0, and others always be 1. Thus this method can be considered as a fixed fault injection.

In above three methods, the laser often needs to destroy the physical structure of the chip, while clock glitch and voltage glitch do not need to do so. That is to say, random fault injection requires lower cost of attack. Accordingly, we consider applying the random fault injection model. Moreover, because the data processing basic units of S-box and encryption components in MIBS are nibble, the DFA methods in this paper are based on model of random nibble faults injection.

4.1 Attack Conditions and Specific Assumptions

a). The intruder fully understands the cryptographic devices. Anytime and anywhere he can inject random nibble faults in the encryption process [26], but the exact value of the fault is unknown.

b). The intruder can repeatedly inject random faults multiple times in the same place.

c). The intruder can repeatedly restart the cryptographic device, and encrypt the same plaintext with the same main key.

4.2 Attack Model

Take the fault position x_0^{32} as an example. We know from Sect. 3.1 that the fault in x_0^{32} can diffuse to the ciphertext $C_0, C_2, C_4, C_5, C_6, C_7$. With regard to the position C_0, the following formula can be obtained from the round function:

$$\begin{aligned} C_0 &= x_8^{32} \oplus (y_0 \oplus y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7) \\ &= x_8^{32} \oplus (S(x_0^{32} \oplus R_{k_0}^{32}) \oplus S(x_1^{32} \oplus R_{k_1}^{32}) \oplus S(x_3^{32} \oplus R_{k_3}^{32}) \\ &\quad \oplus S(x_4^{32} \oplus R_{k_4}^{32}) \oplus S(x_6^{32} \oplus R_{k_6}^{32}) \oplus S(x_7^{32} \oplus R_{k_7}^{32}) \\ &= x_8^{32} \oplus (S(C_8 \oplus R_{k_0}^{32}) \oplus S(C_9 \oplus R_{k_1}^{32}) \oplus S(C_{11} \oplus R_{k_3}^{32}) \\ &\quad \oplus S(C_{12} \oplus R_{k_4}^{32}) \oplus S(C_{14} \oplus R_{k_6}^{32}) \oplus S(C_{15} \oplus R_{k_7}^{32}) \end{aligned}$$

Then inject fault at x_0^{32}, we can get

$$\begin{aligned} C_0^* = x_8^{32} &\oplus (S(C_8^* \oplus R_{k_0}^{32}) \oplus S(C_9^* \oplus R_{k_1}^{32}) \oplus S(C_{11}^* \oplus R_{k_3}^{32}) \\ &\oplus S(C_{12}^* \oplus R_{k_4}^{32}) \oplus S(C_{14}^* \oplus R_{k_6}^{32}) \oplus S(C_{15}^* \oplus R_{k_7}^{32}) \end{aligned}$$

from false ciphertext. Afterwards we get a differential equation by uniting two formulas above

$$\begin{aligned} \Delta C_0 = &S(C_8^* \oplus R_{k_0}^{32}) \oplus S(C_8 \oplus R_{k_0}^{32}) \oplus S(C_9 \oplus R_{k_1}^{32}) \oplus \\ &S(C_9^* \oplus R_{k_1}^{32}) \oplus S(C_{11} \oplus R_{k_3}^{32}) \oplus S(C_{11}^* \oplus R_{k_3}^{32}) \oplus \\ &S(C_{12} \oplus R_{k_4}^{32}) \oplus S(C_{12}^* \oplus R_{k_4}^{32}) \oplus S(C_{14} \oplus R_{k_6}^{32}) \oplus \\ &S(C_{14}^* \oplus R_{k_6}^{32}) \oplus S(C_{15} \oplus R_{k_7}^{32}) \oplus S(C_{15}^* \oplus R_{k_7}^{32}) \end{aligned}$$

In this equation, $R_{k_0}^{32}, R_{k_1}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_6}^{32}, R_{k_7}^{32}$ are unknown. Similarly, we can get 6 equations about $\Delta C_0, \Delta C_1, \Delta C_2, \Delta C_3, \Delta C_5, \Delta C_6$ when injecting faults at x_1^{32}. According to Sect. 3.1, the effect of fault diffusion at different positions is distinct. Thus, the number of differential equations obtained by injecting faults at different positions is not the same either. In order to improve the efficiency of solving differential equations, the amount of equations is as much as possible. We inject faults at $x_0^{32}, x_1^{32}, x_2^{32}, x_3^{32}$ as a group, because the number of differential equations reaches maximum, totally $4 \times 6 = 24$ equations are obtained. The unknown nibbles of round key contained in each equation are listed in Table 6.

From Table 4, it can be seen that when injecting faults at $x_0^{32}, x_1^{32}, x_2^{32}, x_3^{32}$, all differential equations $\Delta C_i (i = 0, 1, \ldots, 7)$ appear three times. We simplify each equation to the form of $S(R_{k_i}^{32} \oplus \alpha) \oplus S(R_{k_i}^{32}) = \beta, (i = 0, 1, \ldots, 7)$. For certain differential equations including $R_{k_i}^{32}$, first we search Table 5 to find a set of key candidate satisfying the equation. And then we select possible keys in the intersections obtained from three equations. By this means $R_{k_i}^{32}$ can be recovered with a rather high probability.

Table 6 Nibbles of round key contained in differential equations

Nibbles of ciphertext difference	Nibbles of round key
ΔC_0	$R_{k_0}^{32}, R_{k_1}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_6}^{32}, R_{k_7}^{32}$
ΔC_1	$R_{k_1}^{32}, R_{k_2}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_5}^{32}, R_{k_6}^{32}$
ΔC_2	$R_{k_0}^{32}, R_{k_1}^{32}, R_{k_2}^{32}, R_{k_4}^{32}, R_{k_5}^{32}, R_{k_7}^{32}$
ΔC_3	$R_{k_1}^{32}, R_{k_2}^{32}, R_{k_3}^{32}, R_{k_6}^{32}, R_{k_7}^{32}$
ΔC_4	$R_{k_0}^{32}, R_{k_2}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_7}^{32}$
ΔC_5	$R_{k_0}^{32}, R_{k_1}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_5}^{32}$
ΔC_6	$R_{k_0}^{32}, R_{k_1}^{32}, R_{k_2}^{32}, R_{k_5}^{32}, R_{k_6}^{32}$
ΔC_7	$R_{k_0}^{32}, R_{k_2}^{32}, R_{k_3}^{32}, R_{k_5}^{32}, R_{k_6}^{32}, R_{k_7}^{32}$

4.3 Attack Process

1). Inject faults

a). Select plaintext P arbitrarily, then encrypt it with the main key K. After that we correctly obtain the correct ciphertext $C = C_0 \| C_1 \| C_2 \| C_3 \| C_4 \| \cdots \| C_{14} \| C_{15}$.

b). Use the same key to encrypt the same plaintext. Then inject a nibble of fault at x_0^{32} in the 32th round of the encrypt operation. Next, we record incorrect ciphertext $C^* = C_0^* \| C_1^* \| C_2^* \| C_3^* \| C_4^* \| \cdots \| C_{14}^* \| C_{15}^*$.

c). Alter x_0^{32} to $x_1^{32}, x_2^{32}, x_3^{32}$, then repeat step b) for three times, and get the remaining three groups of false ciphertext.

2). List the differential equations of S-boxes

Six differential equations are obtained from a set of ciphertext difference. So we could list a total of 24 differential equations, and simplify each equation to the simple form $S(R_{k_j}^{32} \oplus \alpha_j) \oplus S(R_{k_i}^{32}) = \beta_j, (i = 0, 1, \ldots, 7, j = 1, 2, 3)$.

3). Filter the correct key

a). For three differential equations $S(R_{k_0}^{32} \oplus \alpha_j) \oplus S(R_{k_0}^{32}) = \beta_j, (j = 1, 2, 3)$ included $R_{k_0}^{32}$ that we obtained in step 2), look up Table 4 to select the key candidate set N_j that satisfies equations above. Then the intersection of N_1, N_2, N_3 is the correct value of $R_{k_0}^{32}$.

b). Repeat step a) and find the correct value of $R_{k_1}^{32}, R_{k_2}^{32}, R_{k_3}^{32}, R_{k_4}^{32}, R_{k_5}^{32}, R_{k_6}^{32}, R_{k_7}^{32}$, then recover complete R_k^{32}.

4). Recover R_k^{31} and R_k^{30}

a). Use R_k^{32} to decrypt ciphertext and we obtain intermediate output x^{32} of the 31th round. When the algorithm runs to the 31st round, we introduce nibbles of faults at $x_i^{31} (i = 0, 1, 2, 3)$ successively. Then repeat the steps above and recover 32 bit R_k^{31}.

b). Decrypt x^{32} with R_k^{31} to get the intermediate output x^{31} of the 30th round. Rerun the algorithm and inject nibbles of random faults at $x_i^{30} (i = 0, 1, 2, 3)$. Repeat the steps above to recover 32bit R_k^{30}.

5). Retrodict the main key K according to the key expansion scheme

It is easily seen from the key expansion algorithm that W_k^{30} can be expressed by those $R_k^r (r = 30, 31, 32)$, i.e.

$$W_k^{30} = (R_k^{30}[0:31] \| x \| y \| S^{-1}(R_k^{32}[0:3]) \oplus (\text{0x111}) \| R_k^{32}[4:14] \| S^{-1}(R_k^{31}[0:3]) \| R_k^{31}[4:14])$$

Those R_k^r are all known and x, y can be recovered by exhaustion. Finally we can recover 64 bits initial main key by key expansion algorithm.

4.4 Complexity Analysis

For the convenience of discussion, the following analysis of the attack complexity is aimed at recovering the last three round keys. Given the complexity of exhausting the unknown two bits can be ignored, the probability of recovering the main key K is approximately equal to that of recovering the last three round keys. In addition, differential equations in the previous section are denoted as $S(m \oplus \alpha) \oplus S(m) = \beta$, where m is the unknown input of S-box, α a random fault, and β is the output difference. The correspondence between input difference, output difference and possible input value is depicted in Table 5.

According to the specific attack process in Sect. 4.3, only $4 + 4 + 4 = 12$ faults need to be injected, and then the main key K can be completely recovered by exhausting the two unknown bits. Due to the uneven differential distribution of S-box, after injecting some faults, the element in the intersection of key candidate sets N_j may not be unique for a certain differential equation. At this time the corresponding nibble of round key cannot be determined.

According to the **Property 3** and **Property 4**, if and only if the possible input sets of two equations are identical, the number of solutions of the two differential equations is 4. In other cases, the input value m of the S-box can be determined. When $m \neq 0$, the probability that two possible input sets obtained from (α, β) and (α', β') are identical is $3/15 = 0.2$; When $m = 0$, the probability above decreases to $(3/15)^2 = 0.04$. That is to say, when the number of faults injection are identical, the probability that the equation having a solution is lower in the case of $m = 0$. Therefore, we can obtain a lower bound of the probability of recovering R_k by considering the special input, where $m = 0$.

Theorem 1: In the fault injection model in Sect. 4.2, the lower bound of the probability of recovering R_k after injecting l groups ($4l$ times) of faults is

$$p_l^r = 1 - \sum_{i=1}^{8} \frac{(-1)^{i+1} \cdot C_8^i}{5^{(3l-1)i}}.$$

Proof: First consider the situation of $l = 1$, which is shown in Fig. 3. In this case, we can obtain a total of 24 differential equations, of which each $F_i (i = 1, 2, \ldots, 8)$ appear 3 times. For any F_i among them, if the set of possible input values are identical after solving three equations, there are 4 elements in each possible input set. Thus, the probability that those F_i not having unique solution is $\frac{3^2 \times 15^{14}}{15^{16}}$. Moreover, whether each differential equation having a unique solution is mutual independence. The probability that $F_i, F_j (i, j = 1, 2, \ldots, 8, i \neq j)$ not having unique solution is $\frac{3^4 \times 15^{12}}{15^{16}}$. By parity of reasoning, the probability of at least one differential equation having a non-unique solution by Inclusion-exclusion principle is:

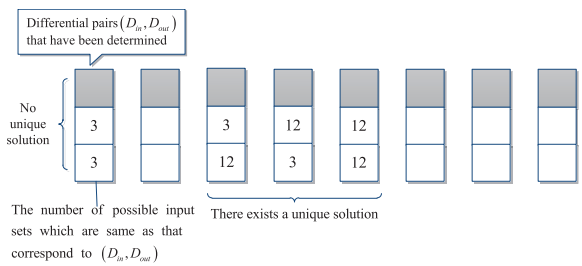


Fig. 3 Status of differential equation solutions when $l = 1$

$$\sum_{i=1}^{8} \frac{C_8^i \cdot 3^{2i} \cdot 15^{16-2i}}{15^{16}} \cdot (-1)^{i+1} = 0.2786$$

Therefore, the probability of recovering R_k^r after injecting l groups of faults is at least $p_3^r = (1 - 0.2786) \times 100\% = 72.14\%$. Similarly, the probability of at least one differential equation having a non-unique solution after l groups ($4l$ times) injection is:

$$\sum_{i=1}^{8} \frac{C_8^i \cdot 3^{(3l-1)i} \cdot 15^{8\cdot(3l-1)-(3l-1)\cdot i}}{15^{8\cdot(3l-1)}} \cdot (-1)^{i+1} = \sum_{i=1}^{8} \frac{-1^{i+1} \cdot C_8^i}{5^{3l-1} i}$$

Its complementary events, that is, the probability of recovering R_k^r after l groups of fault injection is

$$p_l^r = 1 - \sum_{i=1}^{8} \frac{(-1)^{i+1} \cdot C_8^i}{5^{(3l-1)i}},$$

where the specific value of R_k^r is 0x00000000. From the analysis above, when taking other value, the probability of R_k^r being recovered should be greater than p_l^r. □

The specific results are shown in Table 7.

In order to obtain the lower bound of the probability of recovering the main key K, we need to know the probability of recovering R_k^r after exactly injecting l groups of faults, denoted as $p_1^{'r}$. Obviously we have $p_1^{'r} = p_1^r$. When $l \geq 2$, there is $p_1^{'r} = p_l^r - p_{l-1}^r$ by the exclusivity of the event. Moreover, we know p_1^r is greater than 99.99% and the increase becomes flat when $l \geq 4$ in a single round, so the probability of recovering K can be regarded as 100%. Therefore, when calculating the lower bound of probability, only $l < 4$ is considered.

Theorem 2: In the fault injection model in Sect. 4.2, the lower bound of the probability of recovering K after exactly injecting $L(L = 3, 4, \ldots, 9)$ groups of faults is

$$p_L = \sum_{l_1+l_2+l_3=L} p_{l_1}^{'30} \cdot p_{l_2}^{'31} \cdot p_{l_3}^{'32},$$

where $l_1, l_2, l_3 \in \{1, 2, 3\}$.

Proof: First calculate $p_1^{'r}$ by the data in Table 7, which is shown in the following Table 8:

According to the fault injection model in Sect. 4.2, the main key K can be derivated by recovering round keys of 30, 31, and 32 round. Therefore, the total number of fault injections is equal to the sum of that of recovering $R_k^r(r = 30, 31, 32)$. Using multiplication principle and additive principle, the probability of recovering K after injecting L groups of faults when R_k^r is 0x00000000 is denoted as

$$p_L = \sum_{l_1+l_2+l_3=L} p_{l_1}^{'30} \cdot p_{l_2}^{'31} \cdot p_{l_3}^{'32},$$

which is also the lower bound of probability. □

In addition, we denote the lower bound of probability of recovering K after injecting $L(L = 3, 4, \ldots, 9)$ groups of faults as

$$\hat{p}_L = \sum_{l_1+l_2+l_3\leq L} p_{l_1}^{'30} \cdot p_{l_2}^{'31} \cdot p_{l_3}^{'32}.$$

The specific value of p_L and $\hat{p}_L$ are shown in Fig. 4 and Fig. 5.

It can be seen from Fig. 4 that P_L reaches the peak when $L = 4$. On the other hand, when $L > 6$, P_L is rather small and negligible. Therefore, we can use the lower bound of probability in Theorem 2 to calculate the expectation of faults injection number when recovering the main key K, which is $E(L) = \sum_{L=3}^{9} L \cdot p_L = 3.87$.

Table 7 Relationship between l and p_l^r

l (Groups)	1	2	3
$p_l^{30}, p_l^{31}, p_l^{32}$	72.14%	98.73%	99.99%

Table 8 Relationship between l and $p_l^{'r}$

$p_l^{'r}$	$l = 1$	$l = 2$	$l = 3$
$r = 30, 31, 32$	72.14%	26.59%	1.26%

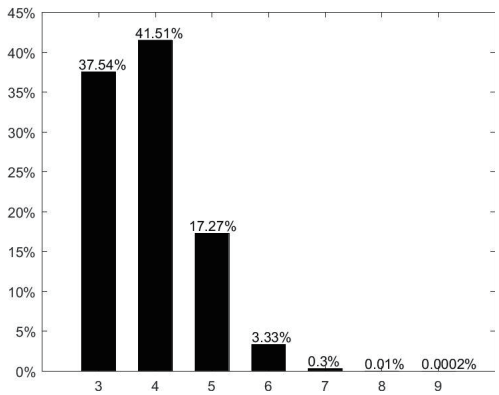


Fig. 4 Relationship between L and p_L

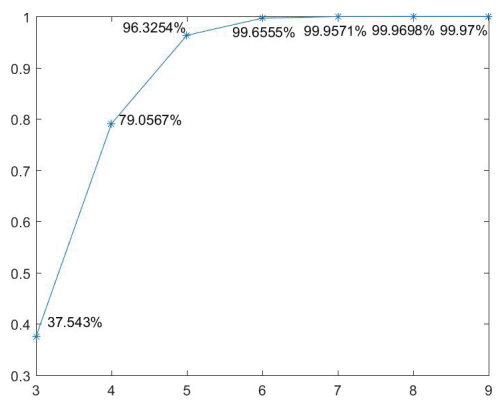


Fig. 5 Relationship between L and $\hat{p}_L$

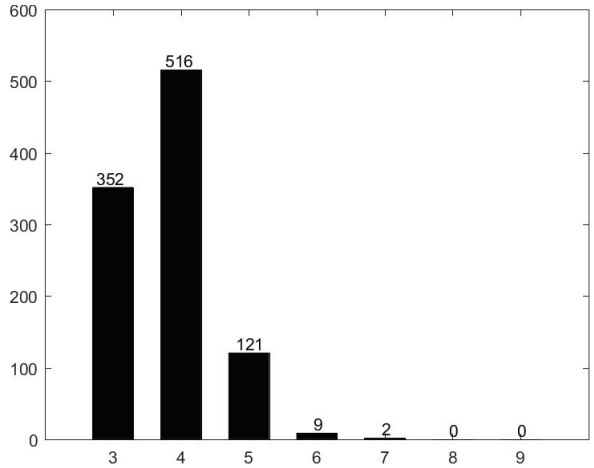


Fig. 6 The number of group of experiments corresponding to total number of fault injections

5. Experimental Simulation Results and Analysis

5.1 Experimental Environment

The hardware is configured as a PC with CPU of Intel CoreTM i5-4200M 2.5GHz, 64-bit operating system and 4GB of memory. The programming environment is Visual C++ for Microsoft Visual Studio 2012 and business mathematics software MATLAB R2016a.

5.2 Analysis of Experimental Results

Set the plaintext as 123456789ABCDEF0 and randomly select the 64bit main key. In order to eliminate the artificial intervention in the process of experiment, we use lightweight stream cipher Trivium to generate 3000 bits pseudo-random data stream. Then we randomly truncate nibbles as fault value per round. After we carry out 10000 trials, the final results of experiments are dipicted in Fig. 6. After calculation, the average groups of fault injections needed to recover the main key K(62 bits) is 3.79, which is close to the theoretical expectation.

6. Conclusion

Based on the differential diffusion properties of MIBS algorithm, we propose a fault injection attack strategy for nibbles and give a method to reduce the number of fault injections. In terms of attack complexity, the lower bound probability of recovering R_k^r can be calculated through related probability knowledge, and the expectation of fault injection groups is 3.87, which is a greater improvement than previous results. At the same time, we use software to verify the simulation process and the results of experiment are close to the expected theoretical ones.

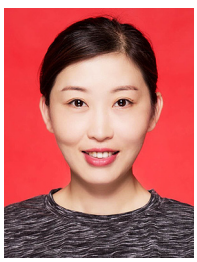
In this paper, we propose the complexity analysis and success rate verification of the DFA method. More importantly, it provides the idea for other fault attack of lightweight ciphers in the future study. Moreover, the next step we will carry out similar discussions and researches on other lightweight block ciphers. On the other hand, we continue to study the method of reducing the width of the fault, and consider decreasing number of fault injection groups so as to improve the efficiency and feasibility of attack. At last, researchers should also pay attention to DFA resilience of lightweight ciphers.

References

[1] W. Wu and L. Zhang, "LBlock: A lightweight block cipher," Lect. Notes Comput. Sci., vol.6715, Berlin: Springer-Verlag, pp.327–344, 2011.
[2] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: An ultra-lightweight block cipher," Int. Conf. Cryptographic Hardware and Embedded Systems, Springer-Verlag, vol.6917, pp.342–357, 2011.
[3] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw, "The LED block cipher," Int. Conf. Cryptographic Hardware and Embedded Systems, Springer-Verlag, vol.6917, pp.326–341, 2011.
[4] T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, "TWINE: A Lightweight Block Cipher for Multiple Platforms," Int. Conf. Selected Areas in Cryptography, Springer, Berlin, Heidelberg, vol.7707, pp.339–354, 2012.
[5] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee, "HIGHT: A new block cipher suitable for low-resource device," Int. Conf. Cryptographic Hardware and Embedded Systems, Springer-Verlag, vol.4249, pp.46–59, 2006.
[6] A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An Ultra-Lightweight Block Cipher," International Workshop on Cryptographic Hardware and Embedded Systems, Springer, Berlin, Heidelberg, vol.4727, pp.450–466, 2007.
[7] M. Izadi, B. Sadeghiyan, S.S. Sadeghian, and H.A. Khanooki, "MIBS: A new lightweight block cipher," Int. Conf. Cryptology and Network Security, Springer, Berlin, Heidelberg, vol.5888, pp.334–348, 2009.
[8] X. Ma, L. Hu, S. Sun, K. Qiao, and J. Shan, "Tighter Security Bound of MIBS Block Cipher against Differential Attack," Int. Conf. Network and System Security, Springer, Cham, vol.8792, pp.518–525, 2014.
[9] L. Fu and C. Jin, "Impossible Differential Cryptanalysis on 13-round MIBS-80," J. Electronics & Information Technology, vol.38, no.4, pp.848–855, 2016.
[10] A. Bay, J. Huang, and S. Vaudenay, "Improved Linear Cryptanalysis of Reduced-Round MIBS," Proc. IWSEC 2014, LNCS, vol.8639, pp.204–220, Springer-Verlag, Switzerland, 2014.
[11] Q.-C. Liu, Y.-Q. Zhao, M. Ma, and F.-M. Liu, "Related-key Invariant Bias Linear Cryptanalysis on MIBS Block Cipher," J. Cryptologic Research, vol.3, no.4, pp.352–360, 2016.
[12] X. Su and J. Guan, "Zero Correlation Linear Cryptanalysis of Lightweight Block Cipher MIBS," J. Information Engineering University, vol.16, no.1, pp.20–24, 2015.
[13] W. Yi, L. Lu, and S. Chen, "Integral and Zero-correlation Linear Cryptanalysis of Lightweight Block Cipher MIBS," J. Electronics & Information Technology, vol.38, no.4, pp.819–826, 2016.
[14] D. Boneh, R.A. DeMillo, and R.J. Lipton, "On the Importance of Checking Cryptographic Protocols for Faults," Int. Conf. Theory and Application of Cryptographic Techniques, vol.1233, pp.37–51, Springer-Verlag, 1997.
[15] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," International Cryptology Conference, vol.1294, pp.513–525, Springer, Berlin, Heidelberg, 1997.

[16] L. Zhang and W. Wu, "Differential fault analysis on SMS4," Chinese Journal of Computers, 2006.
[17] N. Bagheri, R. Ebrahimpour, and N. Ghaedi, "New differential fault analysis on PRESENT," Eurasip Journal on Advances in Signal Processing, vol.2013, no.1, pp.1–10, Sept. 2013.
[18] C. Paar, T. Eisenbarth, M. Kasper, T. Kasper, and A. Moradi, "KeeLoq and Side-Channel Analysis-Evolution of an Attack," Fault Diagnosis and Tolerance in Cryptography, IEEE, pp.65–69, 2009.
[19] W. Li, D. Gu, J. Li, Z. Liu, and Y. Liu, "Differential fault analysis on Camellia," J. Systems & Software, vol.83, no.5, pp.844–851, May 2010.
[20] S. Wang, X. Zhao, T. Wang, et al, "Wide Differential Fault Analysis on MIBS," Computer Science, vol.38, no.4, pp.122–124, 2011.
[21] Z. Wang and Y. Yan, "Differential fault analysis of MIBS cipher by inducing faults to key schedule," Computer Engineering & Design, vol.37, no.6, pp.1435–1439, 2016.
[22] Y.-j. Wang, S.-y. Zhang, T. Wang, and Y. Gao,, "Differential Fault Attack on Block Cipher MIBS," J. University of Electronic Science and Technology of China, vol.47, no.4, pp.601–605, 2018.
[23] S. Endo, T. Sugawara, N. Homma, T. Aoki, and A. Satoh, "An on-chip glitchy-clock generator for testing fault injection attacks," J. Cryptographic Engineering, vol.1, no.4, pp.265–270, 2011.
[24] A.G. Yanci, S. Pickles, and T. Arslan, "Detecting Voltage Glitch Attacks on Secure Devices," Bio-Inspired, Learning and Intelligent Systems for Security, IEEE Computer Society, pp.75–80, 2008.
[25] M. Agoyan, J.-M. Dutertre, A.-P. Mirbaha, D. Naccache, A.-L. Ribotta, and A. Tria, "Single-bit DFA using multiple-byte laser fault injection," IEEE Int. Conf. Technologies for Homeland Security, IEEE, pp.113–119, 2010.
[26] T.G. Malkin, F.-X. Standaert, and M. Yung, "A Comparative Cost/Security Analysis of Fault Attack Countermeasures," Int. Conf. Fault Diagnosis and Tolerance in Cryptography, vol.4236, pp.159–172, Springer-Verlag, 2006.

Yang Gao was born in Henan, Luoyang. He received the B.S. degree in Cryptology from Information Engineering University. He is a Master. His research interests include design and analysis of symmetric cipher algorithm.



Yong-juan Wang was born in Henan, Kaifeng. She received the Ph.D. degree in Cryptology from Information Engineering University. She is a professor of Cryptology. Her research interests include analysis of symmetric cipher algorithm and network cryptography.

Qing-jun Yuan was born in Hebei, Hengshui. He received the M.S. degree in Cryptology from Information Engineering University. He is a Master. His research interests include design and analysis of symmetric cipher algorithm.

Tao Wang was born in Shandong, Linyi. He received the B.S. degree in Cryptology from Information Engineering University. He is a Master. His research interests include design and analysis of symmetric cipher algorithm.

Xiang-bin Wang was born in Henan, Nanyang. He is a senior student in Information Engineering Unversity. His research interests include design and anaiysis of symmetric cipher algorithm.