Copyright © 2015 KSII

Security Analysis of the Lightweight Cryptosystem TWINE in the Internet of Things

Wei Li[1, 2, 3, 4], Wenwen Zhang[1], Dawu Gu[2], Zhi Tao[1], Zhihong Zhou[4, 5], Ya Liu[6, 2], Zhiqiang Liu[2]
[1]School of Computer Science and Technology, Donghua University
Shanghai, 201620, China
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University
Shanghai, 200240, China
[3]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences
Beijing, 100093, China
[4]Shanghai Key Laboratory of Integrate Administration Technologies for Information Security
Shanghai, 200240, China
[5]School of Information Security Engineering, Shanghai Jiao Tong University
Shanghai, 200240, China
[6]Department of Computer Science and Engineering, University of Shanghai for Science and Technology
Shanghai, 200093, China
[e-mail: liwei.cs.cn@gmail.com,zhouzhihong@sjtu.edu.cn]
*Corresponding author: Wei Li, Zhihong Zhou

Received August 20, 2014; revised November 21, 2014; accepted December 16, 2014; published February 28, 2015

Abstract

The TWINE is a new Generalized Feistel Structure (GFS) lightweight cryptosystem in the Internet of Things. It has 36 rounds and the key lengths support 80 bits and 128 bits, which are flexible to provide security for the RFID, smart cards and other highly-constrained devices. Due to the strong attacking ability, fast speed, simple implementation and other characteristics, the differential fault analysis has become an important method to evaluate the security of lightweight cryptosystems. On the basis of the 4-bit fault model and the differential analysis, we propose an effective differential fault attack on the TWINE cryptosystem. Mathematical analysis and simulating experiments show that the attack could recover its 80-bit and 128-bit secret keys by introducing 8 faulty ciphertexts and 18 faulty ciphertexts on average, respectively. The result in this study describes that the TWINE is vulnerable to differential fault analysis. It will be beneficial to the analysis of the same type of other iterated lightweight cryptosystems in the Internet of Things.

Keywords: Internet of Things, Cryptanalysis, Lightweight Cryptosystem, Differential Fault Analysis, TWINE

This work is supported by the National Natural Science Foundation of China under Grant No. 61003278, No. 61472250, No. 61202371, and No. 61402288, Shanghai Natural Science Foundation of under Grant No.15ZR1400300, Innovation Program of Shanghai Municipal Education Commission under Grant No. 14ZZ066, the open research fund of State Key Laboratory of Information Security, the Opening Project of Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, the Fundamental Research Funds China Postdoctoral Science Foundation under Grant No. 2012M521829, Shanghai Postdoctoral Research Funding Program under Grant No. 12R21414500, Plan of action for the innovation of science and technology of Shanghai Municipal Science and Technology Commission under Grant No.14511100300, Shanghai Engineering Research Center Project under Grant No.GCZX14014 and C14001, and the National Social Science Foundation of China under Grant No. 13CFX054.

http://dx.doi.org/10.3837/tiis.2015.02.018 **ISSN : 1976-7277**

1. Introduction

Today's information security engineers are facing with the problem of building a trustworthy system from untrustworthy components. Security experts claim that the only workable solutions demand some minimal number of trustworthy components to date. For ensuring the security of an overall system, these trustworthy components are required to provide some services such as authentication, encryption/decryption, cryptographic tokens and so on. Typically, the security of an overall system is provided at the level of softwares (cryptographic algorithms). Traditional cryptographic protocol designs assume that input and output messages are available to attackers, but other information about keys remains unknown. During the last two decades a new class of attacks against cryptographic devices have become public. These attacks exploit easily accessible side-channel information like input-output behavior under malfunctions, power consumption, running time, and can be mounted by anyone using low-cost equipments [1-4]. These side-channel attacks amplify and evaluate leaked information with the help of statistical methods, and are often much more powerful than classical cryptanalysis [5-9], especially in the Internet of Things. As a representative of the informationization tide, the Internet of Things has been playing a vital part in the daily activities of human society, such as intelligent transportation, modern logistics, food safety, environmental monitoring, etc. However, the traditional cryptographic algorithms are not suitable for solving the increasingly prominent security problems of the Internet of Things, since the application components used in the Internet of Things, such as RFID, smart cards and other highly-constrained devices, are mainly microprocessing equipments with weak computing ability and limited storage capacity. In this case, the lightweight cryptographic algorithms and the related side channel attacks have been widely applied to security analysis of the Internet of Things.

The TWINE is a new lightweight cryptosystem to provide security for the highly--constrained devices in the Internet of Things [10]. Its block size is 64 bits, and the supported key sizes are 80 bits in the TWINE-80 and 128 bits in the TWINE-128, respectively. It repeats 36 rounds and every round consists of the 4-bit to 4-bit S-boxes and the linear transformations. Since its introduction, the TWINE has been the target of classical cryptanalytic efforts. The designers of the TWINE, focus on the impossible differential and saturation attacks, which are regarded as the most critical attacks to the TWINE[10]. Then the biclique cryptanalysis of the TWINE has been proposed in [11-12]. Later M. Coban et al make use of the slow diffusion of both the encryption and the key schedule to describe the mutidimensional meet-in-the-middle attacks on the reduced round of the TWINE [13].

Different from the classical cryptanalysis, differential fault analysis, also called DFA, is one type of side-channel attacks in the Internet of Things [14-15]. It was proposed on the DES cryptosystem by E. Biham and A. Shamir for its strong attacking ability, fast speed, simple implementation and other characteristics [16]. The similar attacks have been applied to AES [17-21], IDEA [22], and ARIA [23] etc. The DFA attack is based on deriving information about the secret key by examining the differences between a cipher resulting from a correct operation and a cipher of the same initial message resulting from a faulty operation.

In the literature, little research has been devoted to the security of the TWINE against the DFA. The TWINE takes the generalized Feistel-based structure, and has neither a bit permutation nor a Galois-Field matrix**.** Its basic components include the 4-bit S-boxes, the XOR and the 4-bit-wise permutation (shuffle). And the diffusion layer is smaller than other

ciphers with the same structure. Thus, it has a rather long diffusion path and maximizes the avalanche effect of the linear transformation. This kind of designing strengthens the security of the TWINE against the DFA attack. It makes the path of fault propagation more difficult to be analyzed. Thus, it is difficult to derive the relationship between the fault and the secret key of the TWINE.

We thus propose an effective DFA method to recover the secret key of the TWINE. It adopts the 4-bit fault model. In the DFA attack, the attackers could induce a random error into the 64-bit layer of the encryption, and thus obtain a faulty ciphertext. By differential analysis, the last subkey could be recovered. Then the attackers could decrypt the right ciphertext to obtain the input of the last round, which is the output of the penultimate round. They repeat the above procedure to recover more subkeys until the secret key is obtained by the key schedule. On this fault model and attacking procedure, our method can recover the secret key of the TWINE. The experiments show that using 8 errors and 18 faults on average could recover the 80-bit and 128-bit secret keys, respectively. To the best of our knowledge, it is the first work that a differential fault attack on the TWINE has been successfully put into practice. Compared with the classical cryptanalysis, the differential fault attack on the TWINE has a good performance in data complexity, time complexity and memory complexity, as **Table 1** shows.

Table 1. Cryptanalysis of the TWINE

Method	Paper	Complexity in the TWINE-80			Complexity in the TWINE-128		
		data	time	memory	data	time	memory
Impossible differential attack	[10]	$2^{61.55}$	$2^{77.04}$	2^{74}	$2^{52.21}$	$2^{115.10}$	2^{118}
Saturation attack	[10]	2^{62}	$2^{68.43}$	2^{67}	$2^{62.81}$	$2^{106.14}$	2^{103}
Biclique cryptanalysis	[11, 12]	2^{60}	$2^{79.10}$	2^{8}	2^{60}	$2^{126.82}$	2^{8}
Mutidimensional meet-in-the-middle attack	[13]	\	\	\	2^{48}	2^{122}	2^{124}
Differential fault attack	This paper	$2^{3.81}$	$2^{16.86}$	$2^{16.01}$	$2^{4.71}$	$2^{16.65}$	$2^{8.98}$

This paper is organized as follows. Section 2 briefly introduces the TWINE. The next section describes the fault model and basic assumption, and proposes our DFA analysis to recover the secret key. Section 4 and 5 summarize the attacking complexity and the experimental results. Finally section 6 concludes the paper.

2. Description of the TWINE

The TWINE is a 64-bit lightweight block cipher with two primary instances taking 80-bit and 128-bit secret keys. It has 36 rounds and is composed of the encryption, the decryption and the key schedule as **Fig. 1** shows [10].

2.1 Encryption

The encryption algorithm of the TWINE-80 and the TWINE-128 is described as **Table 2** shows.

Let $P \in (\{0,1\}^4)^{16}$ be the plaintext and $C \in (\{0,1\}^4)^{16}$ be the ciphertext, respectively. Let $RK \in (\{0,1\}^{32})^{36}$ represent the concentration of 36 subkeys, and $RK^i \in (\{0,1\}^4)^8$ represent the

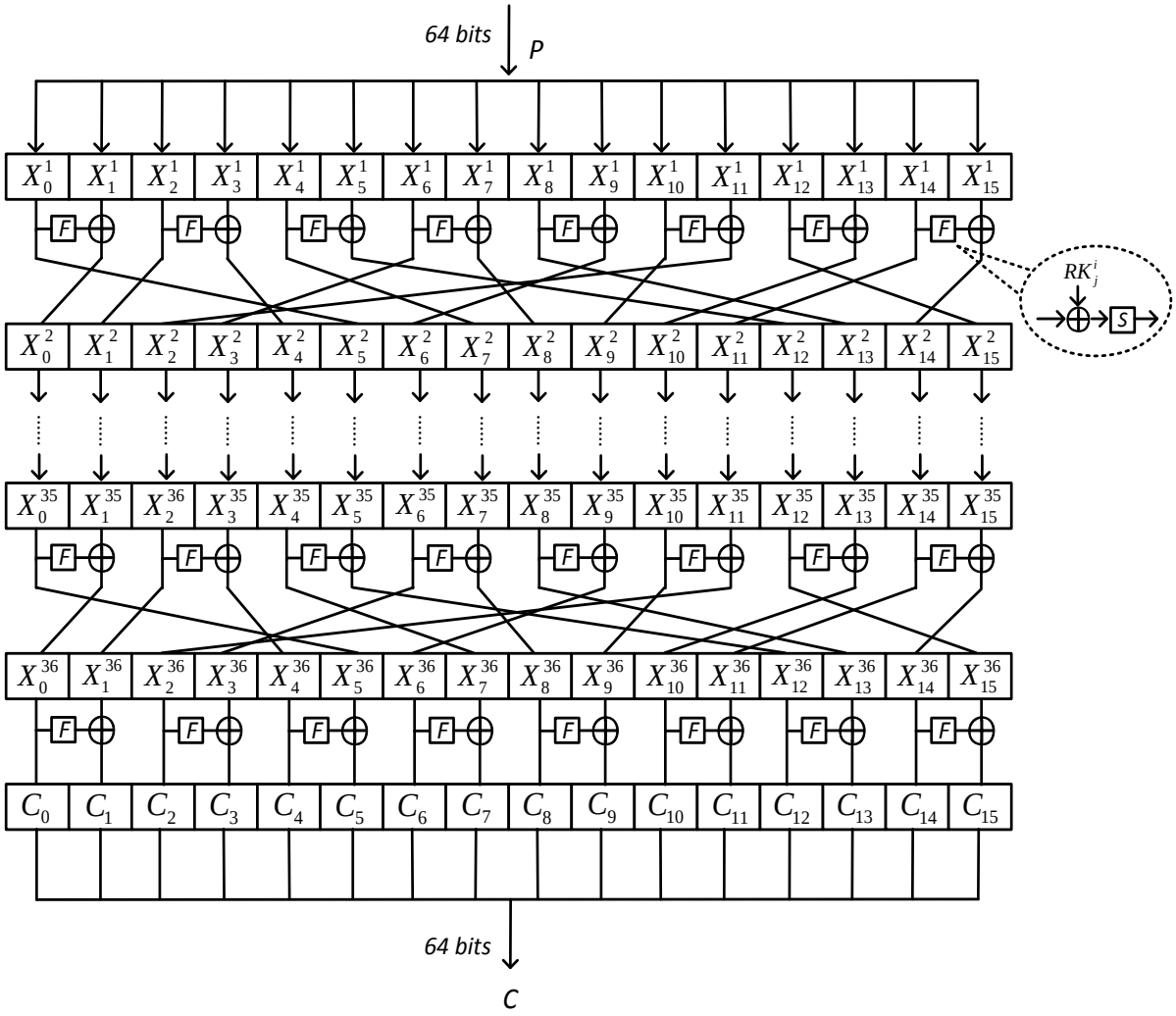

Fig. 1. The structure of the TWINE

i-th subkey from the secret key *K*, with $1 \le i \le 36$, respectively. Let $X_j^i \in \{0,1\}^4$ denote the *j*-th 4-bit input value in the *i*-th round, with $1 \le i \le 36$ and $0 \le j \le 15$.

The round function consists of a substitution layer *S* using 8 4×4 S-boxes and a diffusion layer π permuting 16 blocks.

2.2 Decryption

The decryption is the same as the encryption, including the subkeys with the reverse order.

2.3 Key schedule

The secret key *K* is the input of a key schedule to produce the subkeys for every round. It divides an 80-bit secret key and a 128-bit secret key into 36 subkeys as **Table 3** and **Table 4** show, where CON^r and $\overline{CON^r}$ represent 3-bit constants with $1 \le r \le 35$.

Table 2. The encryption of the TWINE

Input: P, RK Output: C

$(X_0^1 \parallel X_1^1 \parallel \cdots \parallel X_{15}^1) = P$
$(RK^1 \parallel RK^2 \parallel \cdots \parallel RK^{36}) = RK$
for *i*=1 to 35 do
$(RK_0^i \parallel RK_1^i \parallel \cdots \parallel RK_7^i) = RK^i$
for *j*=0 to 7 do $X_{2j+1}^i = S(X_{2j}^i \oplus RK_j^i) \oplus X_{2j+1}^i$
for *l*=0 to 15 do $X_{\pi(l)}^{i+1} = X_l^i$
for *j*=0 to 7 do
$X_{2j+1}^i = S(X_{2j}^i \oplus RK_j^i) \oplus X_{2j+1}^i$
$C = (C_0 \parallel C_1 \parallel \cdots \parallel C_{15}) = (X_0^{36} \parallel X_1^{36} \parallel \cdots \parallel X_{15}^{36})$

Table 3. The key schedule for the TWINE-80

Input: K Output: RK
$(WK_0 \parallel \cdots \parallel WK_{19}) = K$ **for r=1 to 35 do** $RK^r = WK_1 \parallel WK_3 \parallel WK_4 \parallel WK_6 \parallel WK_{13} \parallel WK_{14} \parallel WK_{15} \parallel WK_{16}$ $WK_1 = WK_1 \oplus S(WK_0)$, $WK_4 = WK_4 \oplus S(WK_{16})$ $WK_7 = WK_7 \oplus 0 \parallel CON^r$, $WK_{19} = WK_{19} \oplus 0 \parallel \overline{CON^r}$ $WK_0 \parallel \cdots \parallel WK_3 = (WK_0 \parallel \cdots \parallel WK_3) <<< 4$ $WK_0 \parallel \cdots \parallel WK_{19} = (WK_0 \parallel \cdots \parallel WK_{19}) <<< 16$ $RK^{36} = WK_1 \parallel WK_3 \parallel WK_4 \parallel WK_6 \parallel WK_{13} \parallel WK_{14} \parallel WK_{15} \parallel WK_{16}$ $RK = RK^1 \parallel RK^2 \parallel \cdots \; RK^{35} \parallel RK^{36}$

Table 4. The key schedule for the TWINE-128

Input: K Output: RK
$(WK_0 \parallel \cdots \parallel WK_{31}) = K$ **for r=1 to 35 do** $RK^r = WK_2 \parallel WK_3 \parallel WK_{12} \parallel WK_{15} \parallel WK_{17} \parallel WK_{18} \parallel WK_{28} \parallel WK_{31}$ $WK_1 = WK_1 \oplus S(WK_0)$, $WK_4 = WK_4 \oplus S(WK_{16})$ $WK_{23} = WK_{23} \oplus S(WK_{30})$, $WK_7 = WK_7 \oplus 0 \parallel CON^r$ $WK_{19} = WK_{19} \oplus 0 \parallel \overline{CON^r}$ $WK_0 \parallel \cdots \parallel WK_3 = (WK_0 \parallel \cdots \parallel WK_3) <<< 4$ $WK_0 \parallel \cdots \parallel WK_{31} = (WK_0 \parallel \cdots \parallel WK_{31}) <<< 16$ $RK^{36} = WK_2 \parallel WK_3 \parallel WK_{12} \parallel WK_{15} \parallel WK_{17} \parallel WK_{18} \parallel WK_{28} \parallel WK_{31}$ $RK = RK^1 \parallel RK^2 \parallel \cdots \; RK^{35} \parallel RK^{36}$

3. Differential Fault Analysis on the TWINE

3.1 Notations

The following notations are used to describe the TWINE and its analysis.

Let $C^* \in (\{0,1\}^4)^{16}$ and $\Delta C \in (\{0,1\}^4)^{16}$ be the faulty ciphertext and the ciphertext difference,

respectively.

Let $\Delta X_j^i \in \{0,1\}^4$ represent the *j*-th 4-bit input difference in the *i*-th round, with $1 \le i \le 36$ and $0 \le j \le 15$.

Let A_j^i, B_j^i, ΔA_j^i and ΔB_j^i denote the *j*-th 4-bit input, output, input difference and output difference of the substitution layer in the *i*-th round with $1 \le i \le 36$ and $0 \le j \le 7$, respectively.

For the substitution layers, the relationships between the input differences and output differences of the S-box layers are defined as follows:

$$SS(\Delta A_j^i, \Delta B_j^i) = \{A_j^i \mid A_j^i \in \{0,1\}^4, S(A_j^i) \oplus S(A_j^i \oplus \Delta A_j^i) = \Delta B_j^i\},$$

where $1 \le i \le 36$ and $0 \le j \le 7$.

3.2 Fault model and basic assumption

The DFA analysis exploits the differences between a normal ciphertext and a faulty ciphertext stemming from encryptions of the same plaintext. Our proposed fault model includes the following two assumptions: the attackers have the capability to choose one plaintext to encrypt and obtain the corresponding right and faulty ciphertexts (Chosen Plaintext Attack, CPA). Furthermore, the attackers could induce a 4-bit error to one round of the encryption. In fact, the attackers could assume that the error is one bit or half a byte, and one bit is a special case of half a byte. It does not influence the attacking procedure. Both the value and the location of the error in this round are random. As for the attack, they could analyze a fault occurring near the end of the algorithm and assume the general random fault model where the fault modifies the processed data in a random way.

The attacking procedure is as follows: the right ciphertext is obtained when a plaintext is encrypted with a secret key. The attackers induce a random error in some round of the encryption and thus obtain a faulty ciphertext. The faults could be injected by either using the simulation in software implementation, or using radiation, X-ray and micro-probe in hardware implementation. By differential fault analysis, the value of the last subkey can be recovered. Then the attackers could decrypt the right ciphertext to obtain the input of the last round, which is the output of the penultimate round. At last they repeat the above procedure to deduce more subkeys until the secret key is obtained by the key schedule.

3.3 Attacking Procedure

In this subsection, we apply the above basic idea and propose a novel differential fault analysis to recover the secret keys of the TWINE-80 and the TWINE-128. The analysis is split into the following successive steps for the TWINE.

Step 1. A ciphertext *C* is derived when an arbitrary plaintext *P* is encrypted with a secret key *K*.

Step 2. This step aims at recovering the last subkey RK^{36}. A fault may be induced on either the input or the output of *F* function in the 31st round whereas the approach is identical in

either case. Assume that the error is induced in the first F function in this round. Note that any modification of one 4-bit error provokes the XOR-differences ΔX_0^{32} on X_0^{32}, ΔX_0^{33} on X_0^{33}, ΔX_5^{33} on X_5^{33}, ΔX_0^{34} on X_0^{34}, ΔX_5^{34} on X_5^{34}, ΔX_{12}^{34} on X_{12}^{34}, ΔX_0^{35} on X_0^{35}, ΔX_5^{35} on X_5^{35}, ΔX_{10}^{35} on X_{10}^{35}, ΔX_{12}^{35} on X_{12}^{35}, ΔX_{15}^{35} on X_{15}^{35}, ΔX_0^{36} on X_0^{36}, ΔX_2^{36} on X_2^{36}, ΔX_5^{36} on X_5^{36}, ΔX_9^{36} on X_9^{36}, ΔX_{10}^{36} on X_{10}^{36}, ΔX_{12}^{36} on X_{12}^{36}, ΔX_{14}^{36} on X_{14}^{36} and ΔX_{15}^{36} on X_{15}^{36}. These alter the original ciphertext C into the faulty ciphertext C^*. There is no diffusion layer in the last round, so the input and output difference of the S-boxes in this round can be calculated as follows:

$$\Delta A_j^{36} = \Delta X_{2j}^{36} \oplus \Delta RK_j^{36} = \Delta X_{2j}^{36} = \Delta C_{2j},$$
$$\Delta B_j^{36} = \Delta C_{2j+1} \oplus \Delta X_{2j}^{36} = \Delta C_{2j+1},$$

where $j \in \{0,1,5,6\}$. The above equations, in conjunction with a pair of right faulty ciphertexts, allow to infer the relationship between input differences and output differences of the S-boxes. Thus, the differential transformation of S-boxes in the last round has

$$SS(\Delta A_j^{36}, \Delta B_j^{36}) = \{A_j^{36} \mid A_j^{36} \in \{0,1\}^4, S(A_j^{36}) \oplus S(A_j^{36} \oplus \Delta A_j^{36}) = \Delta B_j^{36}\},$$

where $j \in \{0,1,5,6\}$. The j-th 4-bit value of A^{36} satisfies

$$A_j^{36} \in SS(\Delta A_j^{36}, \Delta B_j^{36}).$$

Then the attackers do brute-force search for the value of ΔA_j^{36} to deduce A_j^{36}, where $j \in \{0,1,5,6\}$. This procedure leads to a list of candidates A_j^{36}, and the value of RK_j^{36} could be deduced as follows:

$$RK_j^{36} = A_j^{36} \oplus C_{2j},$$

where $j \in \{0,1,5,6\}$. Then the attackers could choose the inputs of other F functions in the 31st round to induce errors. So other values of the last subkey could be deduced by the similar method. **Table 5** lists the relationship between the fault locations of the j'-th F function in the 31st round and the affected j-th 4-bit values in the last two subkeys, with $0 \le j' \le 7$ and $0 \le j \le 7$.

Step 3. In this step, there are no errors induced. The attackers could make advantage of the errors in the previous step to deduce the penultimate subkey. They could decrypt the right ciphertext by the last subkey to obtain the output of the penultimate round. The input and output differences of the S-boxes in the penultimate round could be derived as follows:

$$\Delta A_j^{35} = \Delta X_{2j}^{35} \oplus \Delta RK_j^{35} = \Delta X_{2j}^{35} = \Delta X_{\pi(2j)}^{36} = \Delta C_{\pi(2j)},$$
$$\Delta B_j^{35} = \Delta X_{\pi(2j+1)}^{36} \oplus \Delta X_{2j+1}^{35} = \Delta X_{\pi(2j+1)}^{36} = \Delta C_{\pi(2j+1)},$$

and the value of A_j^{35} satisfies

$$A_j^{35} \in SS(\Delta A_j^{35}, \Delta B_j^{35}),$$

where $j \in \{0,5,6\}$. The attackers do brute-force search for the value of ΔA_j^{35} to deduce the input A_j^{35} of S-boxes in the penultimate round. The values of RK_j^{35} could be deduced as below:

$$RK_j^{35} = A_j^{35} \oplus X_{2j}^{35} = A_j^{35} \oplus X_{\pi(2j)}^{36} = A_j^{35} \oplus C_{\pi(2j)},$$

where $j \in \{0,5,6\}$. So other values of the penultimate subkeys could be deduced by the similar method. **Table 5** lists the relationship between the fault locations of the j'-th F function in the 31st round and the affected j-th 4-bit values in the penultimate subkey with $0 \le j' \le 7$ and $0 \le j \le 7$. If the key size is 80 bits, then the attacking procedure jumps step 5; else it jumps step 4.

Table 5. The relationship between the fault locations and the affected 4-bit values in the last two rounds.

The fault locations in the j-th F	The j-th 4-bit value of the last subkey	The j-th 4-bit value of the penultimate subkey
0	0, 1, 5, 6	0, 5, 6
1	0, 1, 3, 7	0, 4, 5
2	2, 3, 4, 7	1, 3, 7
3	1, 2, 3, 5	4, 5, 6
4	2, 4, 5, 6	1, 2, 3
5	0, 3, 4, 5	0, 4, 6
6	0, 4, 5, 6	2, 3, 7
7	1, 2, 6, 7	1, 2, 7

Step 4. The attackers could make advantage of the previous two steps to derive the output of the 34th round, and induce faults the similar locations into the 29th round. After computing the input differences and output differences of the S-boxes, all bytes of RK^{34} and RK^{33} could be deduced.

Step 5. As for the TWINE-80, the last two subkeys could be recovered as follows:

$$RK^{36}=WK_1||WK_3||WK_4||WK_6||WK_{13}||WK_{14}||WK_{15}||WK_{16},$$
$$RK^{35}=WK_0||WK_2||WK_9||WK_{10}||WK_{11}||WK_{12}||WK_{18}||WK_{19}.$$

So the remaining 16 bits of $WK_5||WK_7||WK_8||WK_{17}$ could be derived by the brute-force search. The 80-bit secret key is calculated by

$$K=WK_0||WK_1||\ldots||WK_{19}.$$

As for the TWINE-128, the attackers could derive the last four subkeys as follows:

$$RK^{36}=WK_2||WK_3||WK_{12}||WK_{17}||WK_{18}||WK_{28}||WK_{31},$$
$$RK^{35}=WK_8||WK_{11}||WK_{13}||WK_{14}||WK_{24}||WK_{27}||WK_{29}||WK_{30},$$
$$RK^{34}=WK_4||WK_7||WK_9||WK_{10}||WK_{20}||WK_{23}||WK_{25}||WK_{26},$$
$$RK^{33}=WK_0||WK_5||WK_6||WK_{16}||WK_{19}||WK_{21}||WK_{22}.$$

So the remaining 8 bits of $WK_1||WK_{15}$ could only be derived by the brute-force search. Eventually the 128-bit secret key is calculated by

$$K=WK_0||WK_1||\ldots||WK_{31}.$$

4. Attacking Complexity

We summarize the attacking procedure to select subkey candidates for a secret key. The time complexity of brute-force search for one fault injection is

$$\mu = 2^{2\cdot\omega} \cdot \left\lceil \frac{\sigma}{\omega} \right\rceil,$$

where σ denotes the size of the substitution layer and ω denotes the input size of one S-box. In addition, an estimation of the number of faults necessary for the attack to be successful is vital. In the attacking procedure, the number of faulty ciphertexts to recover a subkey depends on the fault location and the fault model.

We take the derivation of RK^{36} as an example. On the definition of an S-function, if A^{36} is a candidate, $A^{36} \oplus \Delta A^{36}$ may be another subkey candidate. In other words, if the input candidates set of S-boxes is not null, the input A^{36} may have several candidates. It indicates that RK^{36} may have some possible elements.

In the fault model, a random error could be induced at any round of the encryption. If the fault occurs in the last round, only 4 bits in the input of the S-box will change, which could recover at most 4 bits of the last subkey by DFA. To recover the last subkey, it is necessary to induce many errors into different F functions.

If the fault is induced at an ideal location before the last round, then the input difference and output difference of the S-boxes in this round contain only one nonzero 4-bit value. However, the output difference of π has multibytes owing to the diffusion of linear transformation. Thus, the input difference of the S-boxes in the last round contains multibytes after the computation of the last several rounds. The above idea is applied in the attacking procedure to improve the efficiency of fault injection.

Since at least two faults can make one element in the intersection of RK^{36}, the attackers continue deriving intersection of subkey candidates sets until the intersection has only one element. Thus, at least two faulty ciphertexts are required to derive multibytes of one subkey. The theoretical minimum number of faulty ciphertexts to recover one subkey is defined as

$$\phi = \begin{cases} 0 & \text{if } \tau=0 \\ \left\lceil \frac{2\cdot\sigma}{\tau} \right\rceil & \text{if } 1 \le \tau \le \sigma \end{cases},$$

where σ represents the size of the substitution layer, and τ represents the maximum number of bits in a subkey derived by two faulty ciphertexts. To derive the subkey, the value of τ equals the number of bits in the nonzero output difference of the nonlinear transformation in this round. If $\tau = 0$, then there is no bits of a subkey derived and thus $\phi = 0$.

To recover a secret key, the time complexity is

$$\frac{u\cdot\phi\cdot\delta}{2} + 2^{v-\eta} = \begin{cases} 2^{v} & \text{if } \tau=0 \\ 2^{2\cdot\omega} \cdot \left\lceil \frac{\sigma^2\cdot\delta}{\omega\cdot\tau} \right\rceil + 2^{v-\eta} & \text{if } 1 \le \tau \le \sigma \end{cases},$$

the data complexity is

$$\frac{\phi\cdot\delta}{2}+1=\begin{cases}1 & \text{if } \tau=0\\ \left\lceil\frac{\sigma\cdot\delta}{\tau}\right\rceil+1 & \text{if } 1\le\tau\le\sigma\end{cases},$$

chosen plaintext-ciphertext pairs, and the memory complexity is

$$64\cdot 2+\eta+2^{v-\eta},$$

where δ denotes the number of subkeys to recover a secret key, σ represents the size of the substitution layer, ω denotes the input size of one S-box, τ represents the maximum number of bits in a subkey derived by two faulty ciphertexts, v denotes the size of the secret key, and η represents the number of bits in the secret key derived by the DFA. If $\tau=0$, then there is no bits of a subkey derived and thus $\eta=0$.

To recover the 80-bit secret key in theory, the time complexity is $2^{16.59}$, the data complexity is $2^{3.17}$ chosen plaintext-ciphertext pairs, and the memory complexity is $2^{16.01}$, where ω=4, σ=64, τ=16, δ=2, η=64, v=80 and ϕ=8. To recover the 128-bit secret key in theory, the time complexity is $2^{16.01}$, the data complexity is $2^{4.09}$ chosen plaintext-ciphertext pairs, and the memory complexity is $2^{8.98}$, where ω=4, σ=64, τ=16, δ=4, η=120, v=128 and ϕ=8.

5. Experimental Results

We implemented our attack on a PC using Visual C++ 8.0 Compiler on a 2.53 GHz celeron with 2GB memory. The fault induction was simulated by computer software. In this situation, we ran the attacking algorithm to 1000 encryption units with different random generated keys. The experiments are divided as 5 groups in average, denoted as G_1, G_2, G_3, G_4 and G_5.

Fig. 2 and **Fig. 3** show the number of bits recovered in the 80-bit and 128-bit versions in intersections of candidates to recover the secret keys. The x-coordinate represents the number of experiments and the y-coordinate represents the number of the recovered bits of the secret key. In **Fig. 2**, the colored lines denote the number of the recovered bits of the secret key in the 3rd, 6th, 9th and 12th intersections of TWINE-80, respectively. In **Fig. 3**, the colored lines denote the number of the recovered bits of the secret key in the 3rd, 6th, 9th, 12th, 15th, 18th, 21st and 24th intersections of TWINE-128, respectively. We define accuracy, reliability and latency for evaluating the experimental results in detail.

Accuracy is a metric that defines how close the number of the secret key is to the true number of subkey candidates. Basically, the closer the experimental number of the secret key candidates is to the true number, the more accurate the experiment is. Thus, we consider the Root Mean-Square Error (RMSE) to measure the accuracy, where RMSE is given by

$$RMSE=\sqrt{\frac{1}{N}\sum_{e=1}^{N}(h_{true}-h_{measured}(e))},$$

where N denotes the number of experiments in a set, e represents the index of the experiment, h_{true} denotes the number of bits in the secret key, and $h_{measured}$ represents the number of bits

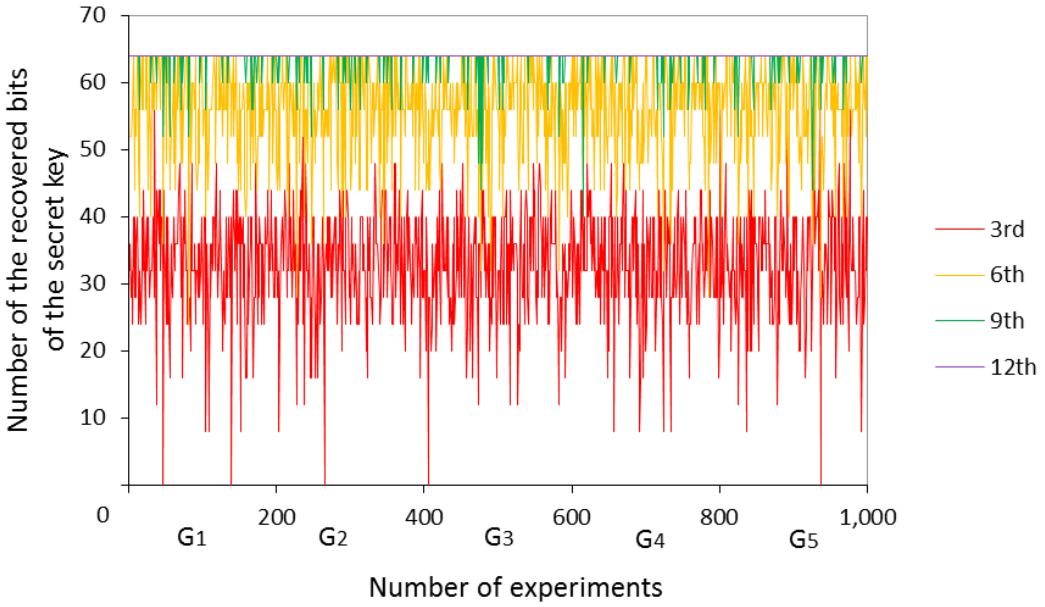

Fig. 2. Number of bits recovered in TWINE–80

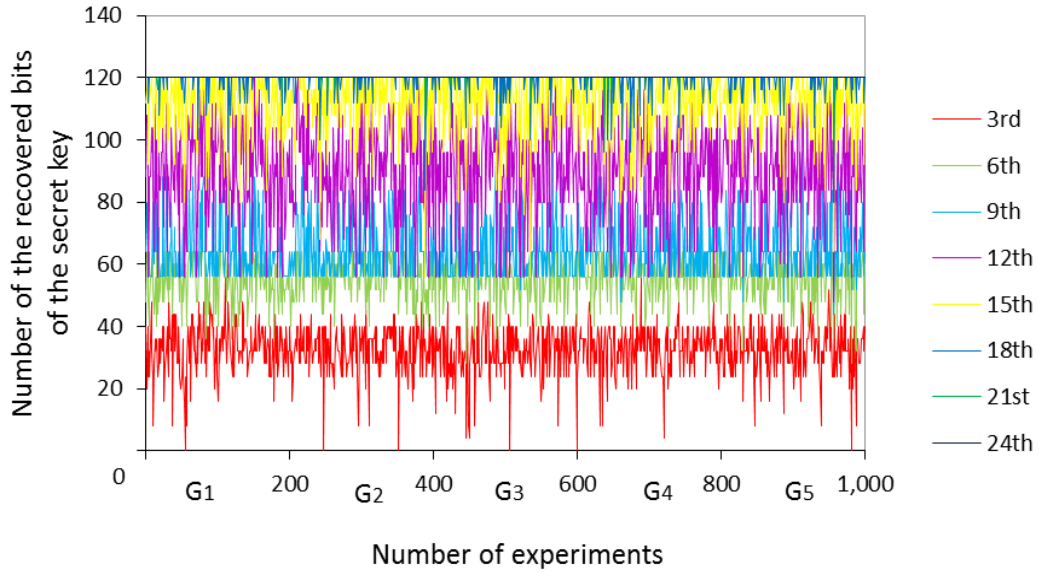

Fig. 3. Number of bits recovered in TWINE-128

recovered in the secret key candidates. As we know, the values of h_{true} are 64 bits for the 80-bit version and 120 bits for the 128-bit version. The closer the RMSE value is to 0, the more accurate the experiments are. The RMSE values for every intersections of subkey candidates are shown in **Table 6** and **Table 7**, where N=200, $e \in \{1,\cdots,1000\}$ and $h_{true} \in \{64,120\}$. For example, to compute the RMSE value of G_1 in the 3rd intersection, it is observed that $h_{\text{measured}}(e)$ represents the values of the red line shown in **Fig. 2**, N=200, $e \in \{1,\cdots,200\}$, and

$h_{true} = 64$. Thus， the RMSE value of G_1 in the 3rd intersection is 5.57. In the same way, all values of RMSE could be derived in **Table 6**. Eventually, the values of RMSE in 12th intersection for the 80-bit version and in the 24th intersection for the 128-bit version are both zero, so we could derive the secret keys in the corresponding intersections. That is, at most 13 and 25 faulty ciphertexts are required to recover secret keys for the two versions, respectively. Furthermore, the accuracy in every group for the same interaction is similar or equal.

Reliability is the ratio of successful experiments out of all experiments made. If the attackers could derive only one secret key, we consider that the experiment is successful. Referring to **Table 8** and **Table 9**, one can observe the ratios of successful experiments in every intersection for the 80-bit and the 128-bit versions. The experimental results show that 12 intersections are enough to recover the secret key for the 80-bits version and 24 intersections are enough for the 128-bits version. That is, the reliability is 100% if the attackers induce at most 13 and 25 random faults to break a secret key for the two versions, respectively. Furthermore, the reliability in every group for the same interaction is similar or equal.

Latency is the time to the recovery the secret key in our software simulation. It is measured in seconds for the 80-bit version and in 0.01 seconds for the 128-bit version. **Fig. 4** and **Fig. 5** show the time of 1000 experiments. The DFA could break the TWINE by recovering 64 bits of the 80-bit secret key and 120 bits of the 128-bit secret key, respectively. The brute-force search could be applied in deriving the remaining 16 bits of the 80-bit secret key and the remaining 8 bits of the 128-bit secret key, respectively. According to the experiment results, the whole attacking procedure requires 4.4s and 0.046s on average to recover the secret keys. The time in the brute-force search of the remaining 16 bits is more than that of the remaining 8 bits. Thus, the whole time of breaking the TWINE-80 is more than that of breaking the TWINE-128.

Table 6. Accuracy by RMSE for the the TWINE–80

Intersection	G_1	G_2	G_3	G_4	G_5
1st	7.30	7.33	7.38	7.31	7.30
2nd	6.49	6.50	6.54	6.52	6.43
3rd	5.57	5.57	5.64	5.59	5.56
4th	4.63	4.58	4.75	4.68	4.65
5th	3.69	3.63	3.80	3.73	3.72
6th	2.78	2.82	2.90	2.79	2.77
7th	1.94	1.96	2.04	1.98	1.98
8th	1.26	1.26	1.31	1.26	1.35
9th	0.79	0.81	0.86	0.76	0.81
10th	0.51	0.49	0.56	0.49	0.57
11th	0.14	0	0.28	0.14	0.40
12th	0	0	0	0	0

Table 7. Accuracy by RMSE for the the TWINE–128

Intersection	G_1	G_2	G_3	G_4	G_5
1st	10.52	10.52	10.49	10.48	10.47
2nd	9.95	9.92	9.92	9.92	9.89
3rd	9.32	9.35	9.41	9.34	9.36
4th	8.84	8.78	8.88	8.82	8.89
5th	8.40	8.37	8.43	8.38	8.42
6th	8.05	8.01	8.06	8.01	8.05
7th	7.86	7.82	7.83	7.81	7.82
8th	7.68	7.64	7.74	7.70	7.75
9th	7.39	7.42	7.47	7.41	7.43
10th	6.90	6.97	6.94	6.98	6.88
11th	6.23	6.30	6.29	6.34	6.19
12th	5.44	5.45	5.51	5.63	5.48
13th	4.60	4.60	4.64	4.71	4.63
14th	3.75	3.74	3.80	3.92	3.78
15th	2.97	2.83	2.95	3.06	2.95
16th	2.29	2.15	2.21	2.34	2.31
17th	1.54	1.48	1.57	1.66	1.56
18th	1.03	1.01	1.07	1.20	1.10
19th	0.73	0.68	0.66	0.76	0.63
20th	0.45	0.45	0.40	0.58	0.47
21st	0.20	0.35	0.28	0.40	0.32
22nd	0.14	0.28	0.14	0.31	0.28
23rd	0	0.20	0	0.14	0.14
24th	0	0	0	0	0

Table 8. Reliability for the the TWINE–80

Intersection	G_1	G_2	G_3	G_4	G_5
1st	0	0	0	0	0
2nd	0	0	0	0	0
3rd	0	0	0	0	0
4th	0	0	0	0	0
5th	3.5%	3.0%	1.5%	3.5%	3.0%
6th	24.0%	22.0%	16.0%	22.0%	21.5%
7th	49.5%	51.0%	44.0%	50.5%	49.5%
8th	75.5%	77.5%	73.5%	77.5%	74.0%
9th	90.0%	88.0%	89.5%	91.0%	89.5%
10th	94.5%	95.0%	95.0%	96.0%	94.5%
11th	99.5%	100.0%	97.5%	99.5%	97.0%
12th	100.0%	100.0%	100.0%	100.0%	100.0%

Table 9. Reliability for the the TWINE–128

Intersection	G_1	G_2	G_3	G_4	G_5
1st	0	0	0	0	0
2nd	0	0	0	0	0
3rd	0	0	0	0	0
4th	0	0	0	0	0
5th	0	0	0	0	0
6th	0	0	0	0	0
7th	0	0	0	0	0
8th	0	0	0	0	0
9th	0	0	0	0	0
10th	0	0	0	0	0
11th	0.5%	0	0	0	0
12th	1.5%	1.0%	1.5%	0.5%	1.0%
13th	7.0%	6.0%	5.0%	4.5%	4.5%
14th	14.5%	15.5%	16.5%	15.5%	12.0%
15th	32.0%	34.0%	37.5%	30.5%	30.0%
16th	52.0%	51.0%	53.0%	49.0%	53.5%
17th	74.5%	72.5%	72.0%	70.5%	71.5%
18th	86.0%	87.0%	86.0%	83.5%	84.0%
19th	90.5%	94.5%	93.0%	93.0%	94.0%
20th	96.0%	97.0%	97.5%	96.0%	97.5%
21st	99.0%	98.0%	98.5%	97.5%	99.5%
22nd	99.5%	99.0%	99.5%	99.0%	99.5%
23rd	100.0%	99.5%	100.0%	99.5%	99.5%
24th	100.0%	100.0%	100.0%	100.0%	100.0%

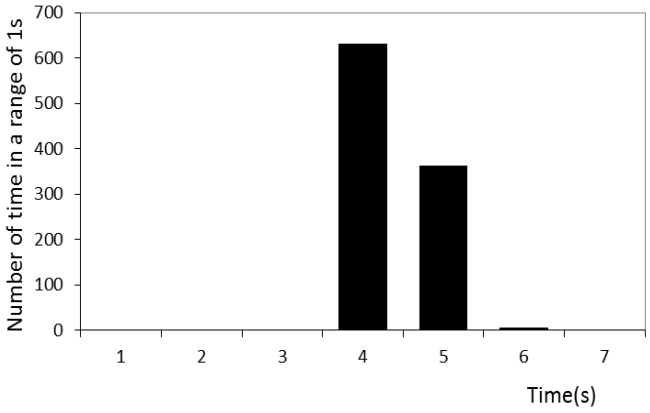

Fig. 4. Latency in attacking the TWINE–80

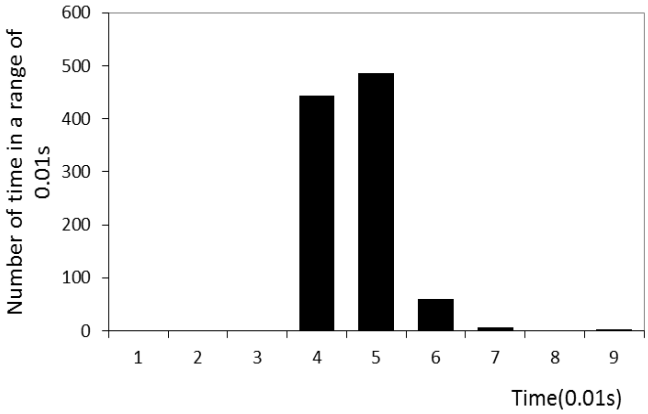

Fig. 5. Latency in attacking the TWINE–128

Referring to the experimental results, breaking the TWINE-80 requires at least 6 faulty ciphertexts and at most 13 faulty ciphertexts. The average number of the faulty ciphertexts of the TWINE-80 is 8. And breaking the TWINE-128 requires at least 12 ciphertexts and at most 25 faulty ciphertexts. The average number of the faulty ciphertexts of the TWINE-128 is 18.

On the basis of the number of faulty ciphertexts in our simulated experiments, the overall time complexities are

$$2^8 \cdot \left\lceil \frac{64}{4} \right\rceil \cdot 13 + 2^{16} \approx 2^{16.86}$$

and

$$2^8 \cdot \left\lceil \frac{64}{4} \right\rceil \cdot 25 + 2^8 \approx 2^{16.65}$$

to break the TWINE-80 and TWINE-128, respectively.

The data complexities in practice are

$$13+1 \approx 2^{3.81}$$

and

$$25+1 \approx 2^{4.71}$$

chosen plaintext-cipertext pairs to break the TWINE-80 and TWINE-128 by the DFA, respectively.

The memory complexities in practice are

$$64 \cdot 2+64+2^{16} \approx 2^{16.01}$$

and

$$64 \cdot 2+120+2^8 \approx 2^{8.98}$$

to break the TWINE-80 and TWINE-128 by the DFA, respectively.

6. Conclusion

As for the fault analysis on lightweight block ciphers, current studies have been published regarding mathematical analysis on cryptographic algorithms, fault injection on cryptographic algorithm in software implementation, fault injection on cryptographic algorithm in hardware implementation. This paper examines fault injection on the TWINE in software implementation. It shows that the TWINE is vulnerable to the differential fault analysis. In the 4-bit fault model, only 8 and 18 ciphertexts on average is required to obtain the 80-bit and 128-bit secret keys of the TWINE, respectively. Our work provides a new reference to fault analysis on other lightweight cryptosystems.

In consequence, we are working on fault injection and detection on the TWINE in hardware implementation. Furthermore, future analysis should be able to support more fault locations of the TWINE, such as the key schedule.

References

[1] P. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *Proc. of 16th Annual Int. Cryptology Conf.,* pp. 104-113, August 18-22, 1996. Article (CrossRef Link).

[2] P. Kocher, J. Jaffe and B. Jun, "Differential power analysis," in *Proc. of 19th Annual Int. Cryptology Conf.,* pp. 388-397, August 15-19, 1999. Article (CrossRef Link).

[3] H. C. Kim and J.-J. Quisquater, "Faults, injection methods, and fault attacks," *IEEE Des. Test Comput.,* vol. 24, no. 6, pp. 544-545, November-December, 2007. Article (CrossRef Link).

[4] M. Joye, J. J. Quisquater, S. M. Yen and M. Yung, "Observability analysis-detecting when improved cryptosystems fail," in *Proc. of Cryptographer's Track RSA Conf.*, pp. 17-29, February 18-22, 2002. Article (CrossRef Link).

[5] I. C. Lin and C. C. Chang, "Security enhancement for digital signature schemes with fault tolerance in RSA," *Inform. Sciences*, vol. 177, no. 19, pp. 4031-4039, February, 2007. Article (CrossRef Link).

[6] J. Kelsey, B. Schneier, D. Wagner and C. Hall, "Side channel cryptanalysis of product ciphers," in *Proc. of 5th European Symp. Research Comp. Security*, pp. 97-110, September 16-18, 1998. Article (CrossRef Link).

[7] W. Erich and G. Johann, "An 8-bit AVR-based elliptic curve cryptographic RISC processor for the internet of things," in *Proc. of 45th Annual Int. Symposium on Microarchitecture*, pp. 39-46, December 1-5, 2012. Article (CrossRef Link).

[8] K. Zhang, L. Ding and J. Li, "Real time related key attack on Hummingbird-2," *KSII T. Internet Inf.*, vol. 6, no. 8, pp. 1946-1963, August 25, 2012. Article (CrossRef Link).

[9] T. Cui and C. Jin, "Finding impossible differentials for Rijndael-like and 3D-like Structures," *KSII T. Internet Inf.*, vol. 7, no. 3, pp. 509-521, March 31, 2013. Article (CrossRef Link).

[10] T. Suzaki, K. Minematsu, S. Morioka and E. Kobayashi, "TWINE: a lightweight block cipher for multiple platforms," in *Proc. of 19th Int. Conf. Selected Areas in Cryptography*, pp. 339-354, August 15-16, 2012. Article (CrossRef Link).

[11] F. Karako, H. Demirci and A. E. Harmanc, "Biclique cryptanalysis of LBlock and TWINE," *Infor. Processing Letters,* vol. 113, no. 12, pp. 423-429, June 30, 2013. Article (CrossRef Link).

[12] M. Coban, F. Karako and O. Boztas, "Biclique cryptanalysis of TWINE," in *Proc. of 11th Int. Conf. Cryptology Network Security*, pp. 43-45, December 12-14, 2012. Article (CrossRef Link).

[13] M. Coban, F. Karako and O. Boztas, "Multidimensional meet-in-the-middle attacks on reduced-round TWINE-128," in *Proc. of 2nd Int. Workshop* on *Lightweight Cryptography for Security and Privacy*, pp. 55-67, May 6-7, 2013. Article (CrossRef Link).

[14] D. Boneh, R. A. DeMillo and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proc. of Int. Conf. Theory Application Cryptographic Techniques*, pp. 37-51, May 11-15, 1997. Article (CrossRef Link).
[15] D. Boneh, R. A. DeMillo and R. J. Lipton, "On the importance of eliminating errors in cryptographic computations," *J. CRYPTOL.*, vol. 14, no. 2, pp. 101-119, March, 2001. Article (CrossRef Link).
[16] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proc. of 17th Annual Int. Cryptology Conf.*, pp. 513-525, August 15-19, 1997. Article (CrossRef Link).
[17] G. Bertoni, L. Breveglieri, I. Koren, P. Maistri and V. Piuri, "Error analysis and detection procedures for a hardware implementation of the Advanced Encryption Standard," *IEEE T. Comput.,* vol. 52, no. 4, pp. 492-505, April 2, 2003. Article (CrossRef Link).
[18] P. Dusart, G. Letourneux and O. Vivolo, "Differential fault analysis on A.E.S," in *Proc. of 1st Int. Conf. Applied Cryptography and Network Security*, pp. 293-306, October 16-19, 2003. Article (CrossRef Link).
[19] C. Giraud, "DFA on AES," in *Proc. of 4th Int. Conf. Advanced Encryption Standard*, pp. 27-41, May 10-12, 2004. Article (CrossRef Link).
[20] A. Moradi, M. T. M. Shalmani and M. Salmasizadeh, "A generalized method of differential fault attack against AES cryptosystem," in *Proc. of 8th Int. Workshop on Cryptographic Hardware and Embedded Systems*, pp. 91-100, October 10-13, 2006. Article (CrossRef Link).
[21] P. Gilles and J. J. Quisquater, "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD," in *Proc. of 5th Int. Workshop on Cryptographic Hardware and Embedded Systems*, pp. 77-88, September 8-10, 2003. Article (CrossRef Link).
[22] C. Christophe, G. Benedikt and V. Ingrid, "Fault analysis study of IDEA," in *Proc. of Cryptographers' Track at the RSA Conf.*, pp. 247-287, April 8-11, 2008. Article (CrossRef Link).
[23] W. Li, D. Gu and J. Li, "Differential fault analysis on the ARIA algorithm," *Inform. Sciences,* vol. 178, no. 19, pp. 3727-3737, October 1, 2008. Article (CrossRef Link).

Wei Li is currently an associate professor in School of Computer Science and Technology, Donghua University. She was awarded as B.S. degree in engineering from Anhui University in 2002, and her M.S. degree and Ph.D. degree in engineering in 2006 and 2009, both from Shanghai Jiao Tong University. She serves as the member for CACR (China Association of Cryptologic Research), CCF (China Computer Federation) and ACM. Her research interests include the design and analysis of symmetric ciphers.

Wenwen Zhang is currently a Master candidate in School of Computer Science and Technology, Donghua University. Her research interests include security analysis of symmetric ciphers**.**

Dawu Gu is a professor at Shanghai Jiao Tong University in Computer Science and Engineering Department. He was awarded a B.S. degree in applied mathematics in 1992, and a Ph.D. degree in cryptograpgy in 1998, both from Xidian University of China. He serves as technical committee members for CACR (China Association of Cryptologic Research) and CCF (China Computer Federation), also as the members of ACM, IACR, IEICE. He was the winner of New Century Excellent Talent Program made by Ministry of Education of China in 2005. He has been invited as Chairs and TPC members for many international conferences like E-Forensics, ISPEC, ICIS, ACSA, CNCC, etc. His research interests cover cryptology and computer security. He has got over 100 scientific papers in academic journals and conferences.

Zhi Tao is currently a Master candidate in School of Computer Science and Technology, Donghua University. His research interests include security analysis of lightweight ciphers.

Zhihong Zhou is a lecturer in Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, Shanghai Jiao tong Univeristy. He was awarded his Ph.D. degree from Zhejiang Univeristy in 2005. His research interests include security analysis of the Internet of Things.

Ya Liu is currently a lecturer in Department of Computer Science and Engineering, University of Shanghai for Science and Technology. She was awarded her Ph.D. degree from Shanghai Jiao Tong University in 2013. Her research interests include the design and analysis of symmetric ciphers and computational number theory.

Zhiqiang Liu is now a Post-doc in the department of Computer Science and Engineering, Shanghai Jiao Tong University. He received his B.S. degree and M.S. degree in Mathematics, and Ph.D. degree in Cryptography from Shanghai Jiao Tong University in 1998, 2001 and 2012 respectively. From 2001 to 2008, he worked in ZTE, Alcatel and VLI in the realm of NextGeneration Network (NGN)/IP Multimedia Subsystem (IMS). Currently, his research interests include cryptanalysis and design of block ciphers and hash functions.