A New Differential Fault Attack on SPN Structure, with Application to AES Cipher

Wei Li[1,2], Xiaoling Xia[1], Dawu Gu[2], Zhiqiang Liu[2], Juanru Li[2], Ya Liu[2]
[1]School of Computer Science and Technology, Donghua University
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University
Email: liwei.cs.cn@gmail.com

***Abstract*—The Substitution-Permutation Network (SPN) is a main type of structure in block ciphers. This paper proposes a new and practical differential fault attack technique on SPN structure. As an instance of SPN cipher, AES-256 can be recovered by 4 faulty ciphertexts. Compared with the previous techniques, our work can recover all subkeys of an SPN cipher with all key sizes. Therefore, our attacking method on AES not only improves the efficiency of fault injection, but also decreases the number of faulty ciphertexts. It provides a new approach for fault analysis on block ciphers.**

***Index Terms*—Cryptanalysis, Side channel attacks, Differential fault analysis, SPN, AES**

I. INTRODUCTION

During the last ten years a new class of attacks against cryptographic devices has become public [21]. These attacks exploit easily accessible information like power consumption, running time, input-output behavior under malfunctions, and can be mounted by anyone using low-cost equipment. These side-channel attacks amplify and evaluate leaked information with the help of statistical methods, and are often much more powerful than classical cryptanalysis. Examples show that a very small amount of side-channel information is enough to completely break a cryptosystem [6, 14, 21]. While many previously-known cryptanalytic attacks can be analyzed by studying algorithms, the vulnerabilities of side-channel attacks result from electrical behavior of transistors and circuits of an implementation. This ultimately compromises cryptography, and shifts the top priority in cryptography from the further improvement of algorithms to the prevention of such attacks by reducing variations in timing, power and radiation from the hardware, and reduction of observability of system behavior after fault injection [2, 6, 17, 25]. Therefore, it extends theoretically the current mathematical models of cryptography to the physical setting which takes into consideration side-channel attacks [4, 18, 20, 23].

Corresponding author: Wei Li.
Tel.: +86 1379 5438 126.

In this paper we focus on one type of side-channel attacks, known as a Differential Fault Analysis (DFA) [6]. The DFA attack was first proposed in 1997 as an attack on DES [6]. Later the similar attacks have been applied to AES [1, 3, 7-9, 11, 12], Triple-DES [13], RC4 [5, 19] 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.

A Substitution-Permutation Network (SPN) is a basic structure, which was first proposed by Feistel [10]. Its basic elements include a substitution transformation and a permutation transformation, which form the foundation of many modern block ciphers, such as AES etc.

As a representative SPN structure, the security of AES against differential fault analysis has been investigated by many researchers [1, 7-9, 11, 12]. There are two kinds of fault models concerning the security in general. One is the bit-oriented model. Several researchers have reported that a single bit fault can be induced on the temporary result within AES [7, 12]. However, this kind of model has its limitation because the operands of most computers are 'byte' or 'word', instead of 'bit'. In practice, it is not easy to induce a one-bit fault. So more important results has been drawn in the byte-oriented model since this fault model requires fewer faulty ciphertexts to mount the attack and makes more reasonable assumption. In the byte-oriented model, P. Dusart et al. took advantage of a single-byte error occurring after the ShiftRows layer of the 9th round in AES-128. They used the particular form of the MixColumns and the SubBytes transformations to solve a list of equations [9]. The method requires about 40 faulty ciphertexts to recover the 128-bit secret key. Under the same fault model, P. Gilles et al. induced errors to the MixColumns transformations between the last but one and two rounds [11]. This method only requires 2 faulty ciphertexts to recover 128-bit secret key. Later M. Amir et al. proposed a generalized method of DFA against AES Their fault model covers all locations before the MixColumns transformation in the 9th round [1]. They could recover 128-bit key of AES with 6 and 1500 faulty ciphertexts in two fault models. This generalized

doi:10.4304/jcp.6.2.216-223

method uses a common and general assumption for error locations and values.

Although many researchers have investigated the representative SPN cipher as above, the security of SPN structure against differential fault analysis was only published in [11]. In the byte-oriented fault model, a single-byte fault can be induced to the diffusion layer in the last but one round. By comparing the correct and faulty ciphertexts, the attacker can retrieve part bytes or all bytes of the subkey in the last round. However, their practical attacking method still has its limitation because the method can only recover one subkey, that is, the subkey in the last round. For some ciphers of SPN structure, the last subkey is not enough to recover all bits in the secret key. For example, recovering the secret key of AES-256 cipher needs at least 2 subkeys. So the practical method in [11] is limited to recover only the last subkey of AES-256, not the whole secret key.

As for the security of AES cipher, P. Gilles et al. reported that their practical attack can be applied to AES cipher with fewer faulty ciphertexts with less complexity [11]. However, we observe that this kind of practical attacking technique only breaks the last subkey of AES-128 since only the last round has no MixColumns transformation. As for AES-192 and AES-256, the last subkey is not enough to derive all bits in the secret key. Furthermore, M. Amir et al. pointed out that their method is not applied to AES with 192-bit and 256-bit keys [1]. So the research in [1] and [11] is limited to recover AES-128. In addition, P. Dusart et al. reported that their method can be applied to AES with all key sizes and 10 ciphertexts to get four bytes of a subkey in the byte-oriented model [9]. In this case, to recover the secret key of AES-256 requires about 80 faulty ciphertexts. Unfortunately, this method decreases the efficiency of fault injection and requires more faulty ciphertexts.

We thus propose a new method to recover the secret key of SPN ciphers, with application to AES. It adopts the byte-oriented model, so the attacker can induce a single-byte error in the encryption. Compared with techniques available, our method can recover all subkeys of SPN ciphers and be applied to SPN ciphers with more key sizes. As for AES-256, we only need 4 faulty ciphertexts to recover the whole secret key. Our attacking method on AES not only improves the efficiency of fault injection, but also decreases the number of faulty ciphertexts. The idea of this attack is also suitable for AES-128, AES-192 and other SPN ciphers.

This paper is organized as follows. Section 2 briefly introduces the SPN structure. The next section describes the assumption of fault model, and proposes our method of DFA on SPN structure. Then section 4 and 5 show our analysis on AES-256 and the experimental results. Finally section 5 concludes the paper.

II. Description of SPN structure

A Substitution-Permutation Network (SPN) processes a *N*-bit plaintext through a series of *R* rounds, and each round consists of a substitution transformation and a permutation transformation. In the substitution layer, every current block is viewed as *n* *m*×*n* subblocks, each of which is a bijective function mapping $\{0,1\}^m \to \{0,1\}^m$. The permutation layer is originally a bit-wise permutation, but more generally an invertible linear transformation [15, 16]. The permutation layer is usually omitted in the last round. An example of a SPN structure is shown in Fig. 1.

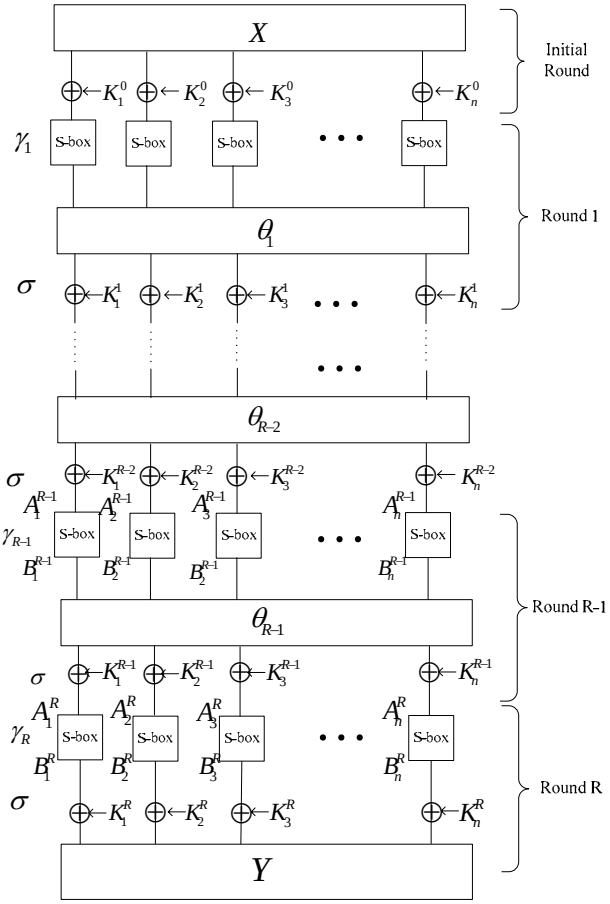


Fig. 1. The Substitution-Permutation Network structure

In an SPN structure, the key schedule generates a total of *R+1* **N**-bit subkeys $(K^0, K^1, \cdots, K^R)$ by the original key K.

The plaintexts are XORed with the initial subkey K^0, and then the subkey K^r are XORed with the output of the permutation layer in the *r*-th round with $1 \le r \le R-1$. The last subkey K^R is XORed with the output of the substitution layer to form the ciphertext.

For the purpose of this paper, we adopt the following structure in the ***r***-th round ($1 \le r \le R-1$).

— The substitution layer γ_r *n* parallel *m*×*n* S-boxes:

$$\gamma_r(A_j^r) = B_j^r,$$

where $1 \le j \le n$.

—The permutation layer θ_r is an invertible linear transformation:

$$\theta_r(B^r) = C^r,$$

$\sigma[K^r]$ denotes the key addition layer:

$$\sigma[K^r] = C^r \oplus K^r.$$

The initial round is proceeded by a key addition layer. The last round has the special form $\sigma[K^R]\circ\gamma_R$, since the θ_r layer at this stage has no cryptographic significance. Thus the whole cipher can be described as

$$\sigma[K^R]\circ\gamma_R\circ\left(\sum_{1}^{r+R-1}\sigma[K^r]\circ\theta_r\circ\gamma_r\right)\circ\sigma[K^0].$$

III. Differential Fault Analysis on SPN Structure

A. Basic idea

The main procedure of this attack is as follows: a ciphertext can be obtained when a plaintext is encrypted with a secret key. When inducing a random single-byte error in some round of the encryption, we can obtain a faulty ciphertext. By differential fault analysis, part bytes or all bytes of the subkey in the last round can be recovered. Repeat the above procedure until we can get the subkeys in all rounds. So the secret key is obtained by the key schedule.

B. The Proposed Attack on SPN structure

In this subsection, we describe a differential fault analysis on the SPN structure. The attack is based on the above basic idea.

Step 1. Obtain a ciphertext *Y* when encrypting an arbitrary plaintext *X* with a secret key *K*.

Step 2. Recover the subkey K^R and perform the following:

(a) In the process of encryption, a random error is induced before diffusion layer in the *R-1*-th round. Then let Y^* and $\triangle Y$ be the faulty ciphertext and ciphertext difference in the encryption, respectively. So $\triangle Y = Y \oplus Y^*$. The last round has no diffusion layer, so the output difference of the S-boxes in the *R*-th round $\triangle B^R$ can be calculated as follows:

$$\triangle B^R = \triangle K^R \oplus \triangle Y = \triangle Y.$$

The ***j***-th output difference of S-boxes in the *R*-th round can be represented by

$$\triangle B_j^R = (\triangle Y)_j,$$

where $1 \le j \le n$.

(b) The input difference and input of the S-boxes in the ***R***-th round $\triangle A_j^R$ and A_j^R satisfy:

$$\gamma_R(A_j^R) \oplus \gamma_R(A_j^R \oplus \triangle A_j^R) = \triangle B_j^R,$$

where $1 \le j \le n$. Brute-force search for the value of $\triangle A^R$ to deduce A^R (See the description of subsection 3.4).

(c) Derive the output of the S-box in the *R*-th round B^R as below:

$$B_j^R = \gamma_R(A_j^R),$$

where $1 \le j \le n$. So all bytes of K^R could be deduced as follows:

$$K^R = Y \oplus B^R$$

Step 3. Recover the subkey K^{R-1}, and perform the following:

(a) Decrypt the last round to get the output A^R when K^R is derived. In the process of encryption, a random error is induced before diffusion layer in the *R-2*-th round in the encryption. The faulty input and output of S-boxes in the last round, denoted by A^{R^*} and B^{R^*} can be derived as follows:

$$B^{R^*} = Y^* \oplus K^R,$$

$$A^{R^*} = \gamma_R^{-1}(B^{R^*}) = \gamma_R^{-1}(Y^* \oplus K^R).$$

So the output difference of S-boxes in the *R-1*-th round can be written as

$$\triangle B^{R-1} = \theta_R^{-1}(\triangle A^R).$$

The *j*-th output difference of S-boxes in the *R*-1-th round can be represented by

$$\triangle B_j^{R-1} = (\theta_R^{-1}(\triangle A^R))_j,$$

where $1 \le j \le n$.

(b) The input difference of the S-box in the *R-1*-th round A^{R-1} satisfies:

$$\gamma_{R-1}(A_j^{R-1}) \oplus \gamma_{R-1}(A_j^{R-1} \oplus \triangle A_j^{R-1}) = \triangle B_j^{R-1},$$

where $1 \le j \le n$.

Brute-force search for the value of $\triangle A^{R-1}$ to deduce A^{R-1} Similar to Step 2(c), we can deduce all bytes of A^{R-1}.

(c) Derive the output B^{R-1} of the S-box in the *R-1*-th round as below:

$$B^{R-1} = \gamma_{R-1}(A^{R-1}).$$

So all bytes of K^{R-1} could be deduced as below:

$$K^{R-1} = A^R \oplus \theta_{R-1}(B^{R-1}).$$

Step 4. Similar to the procedure of step 3, we continue recovering the subkeys until the secret key could be calculated by the above subkeys.

C. Calculation of the S-boxes Input

Suppose that the diffusion layer propagates one single-byte fault to ***t*** nonzero subblocks difference in the input of the substitution layer where $1 \le t \le n$. When we do brute force search for the input of S-boxes, the complexity to recover one subkey is up to 2^{mt} where ***t*** is the number of nonzero output difference and ***m*** is the

subblock size. If $t \geq 4$ and $m = 8$, the brute-force search is not really practical. However, by modifying the attack considerably, this complexity can be reduced. So an effective approach is proposed to select the S-boxes input $A^r (1 \leq r \leq R)$. For simplicity, we take the derivation of A^R as an example with $m = 8$.

Step 1. Induce an error in the encryption and observe the faulty ciphertexts. One single byte error before the diffusion layer in the *R-1*-th round can cause ***t***-byte ($1 \leq t \leq n$) difference after the computation of the diffusion layer.

Step 2. Compute all possible non-zero differences $\triangle A^R$ at the input of S-boxes. Store them in a list

$$\mathcal{Q} = (\mathcal{Q}_1, \mathcal{Q}_2, \cdots, \mathcal{Q}_t).$$

(When a fault is induced before the diffusion layer in the *R-1*-th round, the values of the faulty byte has the possibility of $2^8 - 1$. Moreover, the diffusion layer has ***t***-byte output difference. So $\triangle A^R$ has $(2^8 - 1) * t$ different values.)

Step 3. Brute-force search ***t*** nonzero bytes in $\triangle A^R$ for deriving A^R. For simplicity, suppose that two left-most bytes of $\triangle Y$ are nonzero in ***t*** bytes. It shows that $\triangle A^R$ has two nonzero left-most bytes, denoted by $(\triangle A_1^R, \triangle A_2^R)$.

(a) For any $(\triangle A_1^R, \triangle A_2^R) \in (\mathcal{Q}_1, \mathcal{Q}_2)$, compute the candidates of (A_1^R, A_2^R):

$$(\gamma_R(A_1^R) \oplus \gamma_R(A_1^R \oplus \triangle A_1^R), \gamma_R(A_2^R) \oplus \gamma_R(A_2^R \oplus \triangle A_2^R)) = (\triangle B_1^R, \triangle B_2^R).$$

(b) Since the computation of different S-boxes is independent, we continue searching the 3rd and 4th non-zero bytes for candidates of (A_3^R, A_4^R) where all candidates of $(\triangle A_3^R, \triangle A_4^R)$ are in $(\mathcal{Q}_3, \mathcal{Q}_4)$. Store the candidates of (A_3^R, A_4^R) in the corresponding two bytes $(\mathcal{U}_3, \mathcal{U}_4)$ of a list

$$\mathcal{U} = (\mathcal{U}_1, \mathcal{U}_2, \cdots, \mathcal{U}_t).$$

Repeat this procedure until ***t*** non-zero bytes candidates in A^R are computed. If ***t*** is odd, then we search the ***t-1***-th and ***t***-th non-zero bytes for candidates of (A_{t-1}^R, A_t^R), where all candidates of $(\triangle A_{t-1}^R, \triangle A_t^R)$ are in $(\mathcal{Q}_{t-1}, \mathcal{Q}_t)$ in the last search procedure. Store the results of A^R in the list $\mathcal{U}$.

(c) For any $A^R \in \mathcal{U}$ and $\triangle A^R \in \mathcal{Q}$, we continue shrinking the candidates of A^R as follows:

$$\gamma_R(A_1^R) \oplus \gamma_R(A_1^R \oplus \triangle A_1^R), \gamma_R(A_2^R) \oplus \gamma_R(A_2^R \oplus \triangle A_2^R), \cdots,$$
$$\gamma_R(A_t^R) \oplus \gamma_R(A_t^R \oplus \triangle A_t^R)) = (\triangle B_1^R, \triangle B_2^R, \cdots, \triangle B_t^R).$$

Store the candidates of A^R in the list $\mathcal{U}$.

(d) We gather some faulty ciphertexts caused by the same plaintext and different faults. Then we decrease the number of A^R candidates by repeating the proposed method and collected faulty ciphertexts until the $\mathcal{U}$ has only one element.

Step 4. Repeat the above procedure until the n-byte candidates set of A^R has only one element.

D. Attacking Complexity

In the procedure of recovering the subkey, l denotes the number of faulty ciphertexts caused by the same plaintext, where $l \geq 1$. We continue deriving intersection of subkey candidates sets until the intersection has only one element. This approach selects the subkey candidates with high efficiency. The time complexity to compute ***t***-byte input of the S-boxes is

$$(2^{16} - 1) * (2^8 - 1) * t * \left\lceil \frac{t}{2} \right\rceil * l,$$

where $1 \leq t \leq n$ and $l \geq 1$. So the complexity to compute the whole S-boxes input is at most

$$(2^{16} - 1) * (2^8 - 1) * t * \left\lceil \frac{t}{2} \right\rceil * l * \left\lceil \frac{n}{t} \right\rceil \approx 2^{23} * t * l * n,$$

where $1 \leq t \leq n$ and $l \geq 1$.

IV. APPLICATION TO AES-256

A. Description of AES

AES is a SPN block cipher with 128-bit, 192-bit or 256-bit keys, where the number of rounds is 10, 12 and 14, respectively. The round function is composed of 4 transformations: SubBytes, ShiftRows, MixColumns and AddRoundkey. Let B^r, C^r, D^r, A^{r+1} be the output of every transformation in the ***r***-th round with $1 \leq r \leq R-1$. The final round of AES is composed of the same function, but excludes the MixColumns transformation.

SubBytes: this transformation is a non-linear substitution and operates on each input byte independently. So the substitution table(S-box) is applied on each byte of the input to obtain the output.

ShiftRows: the rows of the temporary result are cyclically shifted over different offsets. Row 0 is not shifted, Row 1 is shifted over 1 byte, Row 2 is shifted over 2 bytes and Row 3 is shifted over 3 bytes.

MixColumns: the columns of the temporary result are considered as polynomials over F_{2^8} and multiplied modulo $x^4 + 1$ with a fixed polynomial.

AddRoundkey: the sukbeys are Xored with the output of the MixColumns. In the last round, the subkey is Xored with the output of the ShiftRows.

B. Our results

As Fig. 2 shows, one byte difference at the input of the MixColumns layer of AES results in a 4-byte difference at its output. Concretely, it means that a fault on one byte before the MixColumns layer will give information on only 4 bytes of the last round key.

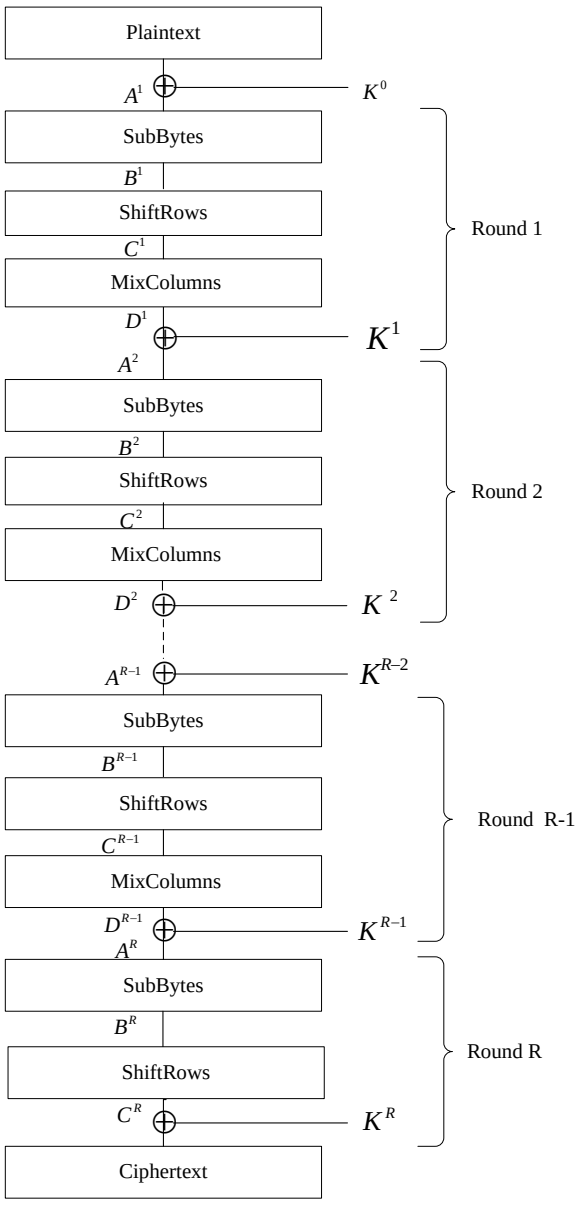


Fig. 2. The structure of AES

When a fault is deduced in anywhere between θ_{R-3} and θ_{R-2}, there are four-byte difference between θ_{R-2} and θ_{R-1} as Fig. 3 and Table. 1 show. Let C_j^r denote the *j*-th byte in C^r where $1 \le j \le 16$.

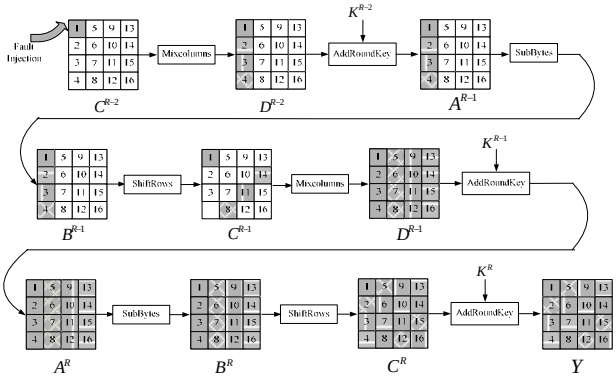


Fig. 3. Differential fault attack on AES-256

(a) A fault on byte C_1^{R-2}, C_2^{R-2}, C_3^{R-2} or C_4^{R-2} will release information about the byte C_1^{R-1}, C_8^{R-1}, C_{11}^{R-1} and C_{14}^{R-1}.

(b) A fault on byte C_5^{R-2}, C_6^{R-2}, C_7^{R-2} or C_8^{R-2} will release information about the byte C_2^{R-1}, C_5^{R-1}, C_{12}^{R-1} and C_{15}^{R-1}.

(c) A fault on byte C_9^{R-2}, C_{10}^{R-2}, C_{11}^{R-2} or C_{12}^{R-2} will release information about the byte C_3^{R-1}, C_6^{R-1}, C_9^{R-1} and C_{16}^{R-1}.

(d) A fault on byte C_{13}^{R-2}, C_{14}^{R-2}, C_{15}^{R-2} or C_{16}^{R-2} will release information about the byte C_4^{R-1}, C_7^{R-1}, C_{10}^{R-1} and C_{13}^{R-1}.

Furthermore, the four-byte difference are in the four different columns. This important property makes all bytes change in the ciphertexts (See Table. 1).

TABLE I.
THE EFFECT OF ONE SINGLE BYTE ON THE CIPHERTEXT

Column	Byte location on		
	C^{R-2}	C^{R-1}	Y
1	1, 2, 3 or 4	1, 8, 11 and 14	1-16
2	5, 6, 7 or 8	2, 5, 12 and 15	1-16
3	9, 10, 11 or 12	3, 6, 9 and 16	1-16
4	13, 14, 15 or 16	4, 7, 10 and 13	1-16

(a) Difference on byte C_1^{R-1}, C_2^{R-1}, C_3^{R-1} or C_4^{R-1} will release information about ciphertexts Y_1, Y_8, Y_{11} and Y_{14}.

(b) Difference on byte C_5^{R-1}, C_6^{R-1}, C_7^{R-1} or C_8^{R-1} will release information about ciphertexts Y_2, Y_5, Y_{12} and Y_{15}.

(c) Difference on byte C_9^{R-1}, C_{10}^{R-1}, C_{11}^{R-1} or C_{12}^{R-1} will release information about ciphertexts Y_3, Y_6, Y_9 and Y_{16}.

(d) Difference on byte C_{13}^{R-1}, C_{14}^{R-1}, C_{15}^{R-1} or C_{16}^{R-1} will release information about ciphertexts Y_4, Y_7, Y_{10} and Y_{13}.

V. Experiments and Results

A. Experiments

In the experiment, R=14, l=2, t=2, m=8, and n=16. Only 2 faults are required to retrieve one subkey. Since the last two subkeys are enough to deduce the secret key, breaking AES-256 requires 4 faulty ciphertexts.

The error locations are in the 12th round and the 11th round, respectively. Our attacking method only requires the complexity about 2^{25}.

We implemented our attack on a PC using Visual C++ 8.0 Compiler on a 2.53 GHz celeron with 512MB memory. The fault induction was simulated by computer software. In this situation, we ran the attack algorithm to 100 encryption unit with different random generated keys. Fig. 4 shows some results in experiments. It describes that the complexity of brute-force search for 4-byte S-boxes input can be decreased from 2^{32} to about 2^{10}. So two errors could recover four bytes of a subkey. While the errors are induced in the last but two and three rounds, recovering a subkey requires 2 faulty ciphertexts. To recover 256-bit secret key, 4 faulty ciphertexts in average are required. The time to complete the attack is a few seconds.

B. Results

Compared with all previous techniques, our method on SPN structure has the following properties. The assumption of our proposed fault model is the same as that in [11]. That is, some register in one round can be induced into a single-byte error (See Table 2). For example, to recover the subkey of the last round, the first fault location is in anywhere between θ_{R-2} and θ_{R-1}.

TABLE II.
Type Sizes for Camera-Ready Papers

Ref.	Appearance		
	Fault model	First fault location	Subkeys scope
[11]	Disturb a byte	Between θ_{R-2} and θ_{R-1}	The last subkey
This paper	Disturb a byte	Between θ_{R-2} and θ_{R-1}	All subkeys

In the same assumption as above, the subkey attacking scope has been extended any round of SPN ciphers(See Table 2), so our proposed method can be applied to SPN ciphers with all key sizes, such as 128 bits, 192 bits and 256 bits.

As for the security of AES-256 against DFA, our approach has the following properties. Firstly, in the byte-oriented model, the efficiency of our proposed attacking method is higher than that in [9] (See Table 3). Our presented method shows that two errors can recover one subkeys, while 10 errors can recover four bytes of one subkey in [9]. Secondly, the number of faulty ciphertexts is less than that in [9] (See Table 3). Our method needs only 4 faulty ciphertexts to recover 256-bit secret key, while 80 faulty ciphertexts are required in [9]. Lastly, our work can recover AES with all key sizes, while the results of [1] and [11] are both limited to recover AES-128.

TABLE III.
Comparison of Differential Fault Attacks on AES

Ref.	Fault model	First fault location	Key size	Fault Ciphertexts
[7]	Disturb a byte	Chosen		128, 256
[12]	Disturb a byte	Any bit of chosen bytes		50
[11]	Disturb a byte	Between θ_{R-2} and θ_{R-1}	128	2
[1]	Disturb a byte	Between θ_{R-1}	128	6 and 1500
[9]	Disturb a byte	Between θ_{R-2} and θ_{R-1}	All	40, 80
This paper	Disturb a byte	Between θ_{R-3} and θ_{R-2}	All	2, 4

VI. Conclusion

In this paper we examine the security of SPN ciphers against the differential fault analysis. In the byte--oriented fault model, only 4 ciphertexts are required to obtain AES-256 secret key. Compared with all techniques of AES against DFA, our method can expand the fault models and recover all subkeys of AES. The idea of this attack is also suitable for AES-128 and AES-192. It provides a new practical approach for fault analysis on block ciphers.

Acknowledgment

This work is supported by the national natural science foundation of china under grant no. 61003278, and Chinese universities scientific fund.

References

[1] Bertoni G., Breveglieri L., Koren I., et al. Error analysis and detection procedures for a hardware implementation of the Advanced Encryption Standard, IEEE Transactions on Computers, 52(4) (2003), pp. 492-505.

[2] Boneh D., DeMillo R., Lipton R.. On the importance of checking cryptographic Protocols for faults. Journal of Cryptology, 14(2) (2001), pp. 101-119.

[3] Biham E., Dunkelman O., Keller N.. The rectangle attack--rectangling the Serpent. In: EUROCRYPT 2001, LNCS, vol. 2045, 2001, pp. 340-357.

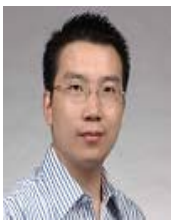
[4] Biham E., Dunkelman O., Keller N.. Linear Cryptanalysis of reduced round Serpent. In: Fast Software Encryption--FSE 2001, LNCS, vol. 2355, 2001, pp. 16-27.

[5] Biham E., Dunkelman O., Keller N.. Differential-linear cryptanalysis of Serpent. In: Fast Software Encryption--FSE 2003, LNCS, vol. 2887, 2003, pp. 9-21.
[6] Biham E., Dunkelman O., Keller N.. New Results on boomerang and rectangle attacks. In: Fast Software Encryption-FSE 2002, LNCS, vol. 2501, 2002, pp. 254-266.
[7] Biham E., Shamir A.. Differential fault analysis of secret key cryptosystems. In: Advances in Cryptology--CRYPTO'97, LNCS, vol. 1294, 1997, pp. 513--525.
[8] Blomer J., Seifert J. P.. Fault based cryptanalysis of the advanced encryption standard (AES). In: Financial Cryptography-FC 2003, LNCS, vol. 2742, 2003, pp. 162--181.
[9] Christophe C., Benedikt G., Ingrid V.. Fault analysis study of IDEA. In: Topics in Cryptography-CT-RSA 2008, LNCS, vol. 4964, 2008, pp. 247-287.
[10] Collard B., Standaert F.- X., Quisquater J.- J.. Improved and multiple linear cryptanalysis of reduced round Serpent. Inscrypt 2007, Lecture Notes in Computer Science 4990 (Springer, Heidelberg, 2008), pp. 51-65.
[11] Chen H., Wu W., Feng, D.. Differential fault analysis on CLEFIA. In: International Conference on Information and Communication Security-ICICS 2007, LNCS, vol. 4861, 2007, pp. 284-295.
[12] Chen C. N., Yen S. M.. Differential fault analysis on AES key schedule and some countermeasures. In: Proceedings of the Australasian Conference on Information Security and Privacy-ACISP 2003, LNCS, vol. 2727, 2003, pp. 118--129.
[13] Dunkelman O., Indesteege S., Keller N.. A Differential--linear attack on 12--Round Serpent, In: INDOCRYPT 2008, LNCS 5365, 2008, pp. 308-321.
[14] Duo L., Li C., Feng K.. New observation on Camellia. In: Selected Areas in Cryptography--SAC 2005, LNCS, vol. 3897, 2005, pp. 51-64.
[15] Dusart P., Letourneux G., Vivolo O.. Differential fault analysis on AES. In: Applied Cryptography and Network Security-ACNS 2003, LNCS, vol. 2846, 2003, pp. 293--306.
[16] Giraud C.. DFA on AES. In: Advanced Encryption Standard--AES, LNCS, vol. 3373, 2005, pp. 27-41.
[17] Hemme L.. A differential fault analysis against early rounds of (Triple-) DES. In: Cryptographic Hardware and Embedded Systems-CHES 2004, LNCS, vol. 3156, 2004, pp. 254-267.
[18] Kim C. H., Quisquater J. J.. Faults, injection methods, and fault attacks. IEEE Design&Test of Computers, 24(6) (2007), pp. 544-545.
[19] Kelsey J., Schneier B., Wagner D., et al. Side channel cryptanalysis of product ciphers. In: ESORICS '98 Proceedings, LNCS, vol. 1485, 1998, pp. 97-110.
[20] Li W., Gu D., Li J.. Differential fault analysis on the ARIA algorithm. Information Sciences, 10(178)(2008), pp. 3727--3737.
[21] Li W., Gu D., Wang Y.. Differential fault analysis on the contracting UFN structure, with application to SMS4 and MacGuffin. Journal of Systems and Software, 82(2009), pp. 346-354.
[22] Kelsey J., Kohno T., Schneier B.. Amplified boomerang attacks against reduced-round MARS and Serpent. In: Fast Software Encryption-FSE 2001, LNCS, vol. 1978, 2001, pp. 75-93.
[23] Kohno T., Kelsey J., Schneier B.. Preliminary Cryptanalysis of Reduced--Round Serpent. In: AES Candidate Conference, 2000, pp. 195-211.
[24] Moradi A., Shalmani, M. T. M., Salmasizadeh M.. A generalized method of differential fault attack against AES cryptosystem. In: Cryptographic Hardware and Embedded Systems-CHES 2006, LNCS, vol. 4249, 2006, pp. 91-100.
[25] Piret G., Quisquater J. J.. A differential fault attack technique against SPN structures, with application to the AES and KHAZAD. In: Cryptographic Hardware and Embedded Systems-CHES 2003, LNCS, vol. 2779, 2003, pp. 77-88.

Wei Li, born in 1980, is currently a lecturer in School of Computer Science and Technology, Donghua University. She was awarded a 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 members for CACR (China Association of Cryptologic Research), CCF (China Computer Federation), and IEEE. Her research interests include the design and analysis of symmetric ciphers.

Xiaoling Xia is a vice professor in School of Computer Science and Technology, Donghua University. She was awarded a B.S. degree and a Ph.D. degree in engineering in 1988 and 1994, both from Shanghai Jiao Tong University. Her research interests include software engineering.

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 cryptography in 1998, both from Xidian niversity 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.



Zhiqiang Liu, born in 1977, is now a Ph.D. candidate in the department of Computer Science and Engineering, Shanghai Jiao Tong University. He received his B.S. degree and M.S. degree in mathematics from Shanghai Jiao Tong University in 1998 and 2001, respectively. From 2001 to 2008, he worked in ZTE, Alcatel and VLI in the realm of Next Generation Network (NGN)/IP Multimedia Subsystem (IMS). Currently, his research interests include cryptanalysis and design of block ciphers and Hash functions.

Juanru Li is currently a Ph.D. candidate in Computer Science and EngineeringDepartment, Shanghai Jiao Tong University. He was awarded his B.S. degree in engineering from Shanghai Jiao Tong University in 2007. His research interests include lightweight block cipher, software security and reverse code engineering.

Ya Liu is currently a Ph.D. candidate in Computer Science and Engineering Department, Shanghai Jiao Tong University. She was awarded her B.S. degree and M.S. degree in applied mathematics from Anhui Normal University in 2004 and 2007, respectively. Her research interests include the design and analysis of symmetric ciphers and computational number theory.