applied sciences

MDPI

Article

A Novel DFA on AES: Based on Two–Byte Fault Model with Discontiguous Rows

Xusen Wan †, Jinbao Zhang †, Shi Cheng, Weixiang Wu and Jiehua Wang *

School of Information Science and Technology, Nantong University, Nantong 226019, China
* Correspondence: wang.jh@ntu.edu.cn; Tel.: +86-13962955885
† These authors contributed equally to this work.

Abstract: Differential fault attack (DFA) is a distinctive methodology for acquiring the key to block ciphers, which comprises two distinct strategies: DFA on the state and DFA on the key schedule. Given the widespread adoption of the Advanced Encryption Standard (AES), it has emerged as a prominent target for DFA. This paper presents an efficient DFA on the AES, utilizing a two−byte fault model that induces faults at the state with discontiguous rows. The experiment demonstrates that, based on the proposed fault model, the key for AES–128, AES–192, and AES–256 can be successfully recovered by exploiting two, two, and four faults, respectively, without the need for exhaustive research. Notably, in the case of AES–256, when considering exhaustive research, two (or three) faults are needed with 2^{32} (or 2^{16}) exhaustive searches. In comparison to the currently available DFA on the AES state, the proposed attack method shows a higher efficiency due to the reduced induced faults.

Keywords: differential fault attack; advanced encryption standard; two−byte fault model; information security

Citation: Wan, X.; Zhang, J.; Cheng, S.; Wu, W.; Wang, J. A Novel DFA on AES: Based on Two–Byte Fault Model with Discontiguous Rows. *Appl. Sci.* **2023**, *13*, 8282. https://doi.org/10.3390/app13148282

Academic Editor: Yutaka Ishibashi

Received: 14 June 2023
Revised: 15 July 2023
Accepted: 15 July 2023
Published: 18 July 2023

1. Introduction

In 2001, The National Institute of Standards and Technology (NIST) adopted the advanced encryption standard, also known as AES, as a symmetric block cryptographic standard in 2001 [1]. The widespread adoption of AES is largely due to its reputation for being secure and tamper–resistant. As a result, it has been used as a cryptographic protocol for subscriber identity module (SIM) cards, wireless fidelity (WIFI) routing, and the encrypted delivery of sensitive data [2–4]. In wireless communication, data transmission is fault–tolerant, so an energy–efficient and cooperative fault–tolerant communication approach was proposed to improve fault–tolerance [5]. On the contrary, data encryption is essential to ensure data reliability. A vehicular network with an AES encoder circuit was designed to keep the vehicular data from attacking when the vehicle is running [6]. Researchers implemented AES using different tools, Ramya et al. proposed an efficient AES using VLSI [7]. In wireless sensor networks (WSN), Luminiţa et al. designed an improved C Language implementation of AES [8]. However, as technology has developed, numerous academics have examined AES's security. Potential dangers are present in real–world encryption systems and are vulnerable to side–channel and fault attacks.

One FA technique proposed to crack block ciphers like AES [9–27], KLEIN [28,29], SIMON [30,31], and SIMECK [32,33] is differential fault attack (DFA). It operates by taking advantage of differences in information between correct and fault ciphertexts, which poses a serious threat to the encryption executive equipment. However, the hardware devices need to be obtained and have the ability to induce a fault. Thus, it is assumed that these two prerequisites are fulfilled when DFA is implemented. The DFA was successfully used to crack DES after being first introduced by Biham and Shamir [9]. The key schedule [10–16] and the state [17–26] are two categories into which DFA on AES can be applied, depending on where the fault is introduced. In addition, a hybrid DFA model that considers faults caused at both the key schedule and state exists [27].

DFA is often classified into single-byte fault model and multi–byte fault model categories when reviewing fault models. In this article, discontiguous rows are referred to as Dcor, while PCFCs express pairs of correct and faulty ciphertexts. Previous DFA research in this area has been divided into several distinct aspects, including minimizing fault caused, utilizing various fault models, generating faults at early rounds, and extending AES–192 and AES–256. In 2011, Tunstall et al. [23] described an analysis based on a single–byte fault induced at the state to crack AES–192 and AES–256, which require 16 and 16 PCFCs, respectively. In 2012, Kim et al. [14] introduced a single–byte fault model, which could deduce the AES–129 key with 16 PCFCs. Meanwhile, AES–128 key retrieval was shown by Chong et al. [21] using one PCFC and thorough searches of 2^8 candidates. In 2012, Kim et al. [10] proposed a single–byte fault model to crack AES–128, AES–192, and AES–256 with two, four, and four PCFCs without exhaustive search, respectively, to decrease the number of PCFCs and the exhaustive search of key candidates. AES–192 was successfully broken in 2020 by Han et al. [18], who hypothesized a single–byte flaw in the key schedule.

Moreover, multi–byte fault models have been applied. Two hypotheses were put up by the authors in 2006 [17]: one is that all four bytes in a column are faulty, while the other is that at most three bytes of a column are faulty. While the second type only requires six PCFCs, the first model requires roughly 1500 PCFCs. Four fault models, M0, M1, M2, and M3, were provided by the authors in 2009 (see Section 3.2), and M0, M1, and M2 are utilized to recover the AES–128 key with one, two, and four PCFCs, respectively [21]. In 2012, Kim [22] showed AES can be cracked with the same multi–byte fault proposed by Kim (M1 and M2), and successfully recovered AES–128, AES–192, and AES–256 keys. In 2017, Liao et al. [26] carried out a fault model with unknown and random multi–byte fault to crack AES–128, using on average only 2.17 PCFCs to recover the last round–key. In 2019, Zhang et al. [13] presented a fault model by exploiting a two–byte fault model with discontiguous rows to crack AES–128, which requires two PCFCs without exhaustive search.

Zhang et al. first proposed the two–byte fault with Dcor, which fault was induced at the key schedule. We were inspired by the two–byte fault with Dcor at the key schedule; in this article, a two–byte fault model with Dcor at the state is proposed to crack AES and successfully crack AES–128, AES–192, and AES–256. The method has the following advantages compared to previous works:

- A Dcor fault model that has been developed calls for fewer PCFCs than the existing model. Using the M1 model, for instance, [22] shows that although breaking AES–192 requires two PCFCs and 2^8 exhaustive searches, cracking AES–256 requires three PCFCs and 2^{32} exhaustive searches. The proposed Dcor fault model, on the other hand, requires fewer PCFCs and thorough searches to recover AES–192. More specifically, recovering AES–192 only requires two PCFCs. The quantity of PCFCs and thorough searches needed for AES–256 relies on the use of different PCFCs. AES–256 can be cracked with 2^{32} (or 2^{16}) exhaustive searches and two (or three) PCFCs, respectively.
- The fault model with faults induced at Dcor of the state. The multi–byte fault is condensed into the Dcor fault, which is present in the first column of the state, as opposed to the earlier one [21,22]. The faults are dispersed throughout each state's column in M0, M1, M2, and M3, which are more intricate. AES–128 is cracked in [13], and the authors presented a two–byte model with Dcor in the key schedule. However, the Dcor model proposed in this article deals with all variants of AES.
- The location of the fault induced is invariant. The location of the fault injection remains the same throughout the study, the conditions of fault induction (e.g., voltage, frequency) don't need to be altered.

The rest of this article is organized as follows: In Section 2, AES is introduced. In Section 3, the Dcor fault model is proposed and attacks AES. Experimental results and discussion in Section 4. Finally, Section 5 concludes this paper.

2. Description of AES

AES is divided into three forms, encrypting 128-bit plaintext with 128–, 192–, and 256–bit keys, and the result of each intermediate round is called state. The ith round state, round key, and ciphertext of AES can be divided into a 4*4 matrix, and each grid represents 8 bytes, denoted as S^i, K^i, and C as shown in Figure 1. The ith round–key can also be denoted as $\{w(4i), w(4i+1), w(4i+2), w(4i+3)\}$.

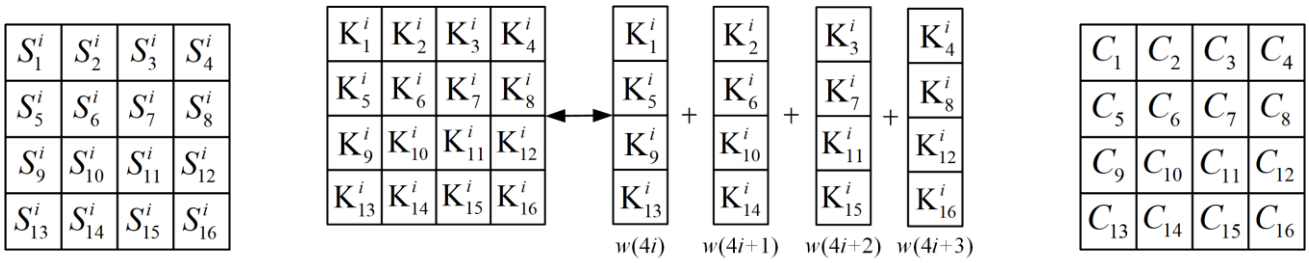

Figure 1. Use of a 4*4 matrix to represent the state, the key, and the ciphertext. (**a**) The state; (**b**) the key; (**c**) the ciphertext.

The number of rounds is dependent on the key length: for keys of 128 bits, 192 bits, and 256 bits, the equivalent encryption rounds are 10, 12, and 14, respectively. Except for the first round, which adds an extra round–key, and the last round, which omits the MixColumn, each round is made up of identical stages. The amount of AES encryption rounds and key schedule rounds is displayed in Table 1.

Table 1. Comparison of the three types of AES encryption.

Type	Encryption Rounds (R)	Key Schedule Rounds
AES–128	10	10
AES–192	12	8
AES–256	14	7

2.1. Notations

The notations used in this article are listed below, and two notes are defined when they are mentioned.

R: the number of rounds of AES encryption.
C: the correct ciphertext.
C_j^{*i}: the jth byte corresponding to the ith fault ciphertext.
S^i: the state of the ith round in the round transformation.
S_j^i: the jth byte of the ith round state.
K^i: the round–key of the ith round in the key schedule.
K_j^i: the jth byte of the ith round–key in the key schedule.
SB: SubByte.
SB^{-1}: InvSubByte.
SR: ShiftRow.
SR^{-1}: InvShiftRow.
MC: MixColumn.
MC^{-1}: InvMixColumn.
ARK: AddRoundKey.

2.2. Encryption Process

The round operations and key scheduling make up the AES encryption process. The following three stages of operation are performed on the round operations:

(a) SubByte (*SB*) Layer.
(b) Diffusion Layer: it consists of two sublayers of ShiftRow (*SR*) and MixColumn (*MC*).
(c) AddRoundKey (*ARK*) Layer.

The last round of AES–128, AES–192, and AES–256 encryption does not contain *MC*. The encryption process of AES is represented in Figure 2. The 128 bits of input data are calculated through the R round to get the 128 bits of output data, and each round will have a corresponding round–key involved.

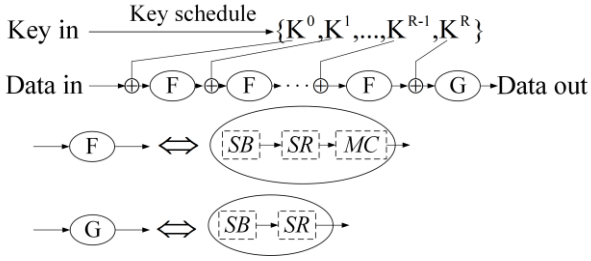

Figure 2. The encryption process of AES.

The round–key is acquired through the key schedule. The same function 'T', consisting of RotWord, SubByte, and XOR with Rcon[*i*] will be applied to each round–key in the last column of the key scheduling.

(a) RotWord: A cyclic rotation operation, such as a four–byte input (X1, X2, X3, X4) through the RotWord function will produce a four-byte output (X2, X3, X4, X1).

(b) SubByte: A byte substitution operation in which each byte entered is mapped to another byte by a nonlinear substitution S box, similar to the S–byte of transformations in the round transform.

(c) Rcon[*i*]: A 32–bit round constant word set defined as Rcon[*i*] = (*RC*[*i*], {00}, {00}, {00}), where *RC*[*i*] is represented as the number on $GF(2^8)$. The specific values are shown in Table 2.

Table 2. The value of each round of constants.

i	1	2	3	4	5	6	7	8	9	10
RC[*i*]	01	02	04	08	10	20	40	80	1B	36

The final two rounds of the AES–128 key schedule are depicted in Figure 3. It is necessary to know the round–key of any round to recover the AES–128 key, usually to recover the last round of keys K^{10} {*w*(40), *w*(41), *w*(42), *w*(43)} (see the green part of Figure 3). Figure 3 only shows the process of the last two round–keys of AES–128, for all this, it contains all information of the initial key, therefore, the initial key can be recovered according to the key schedule.

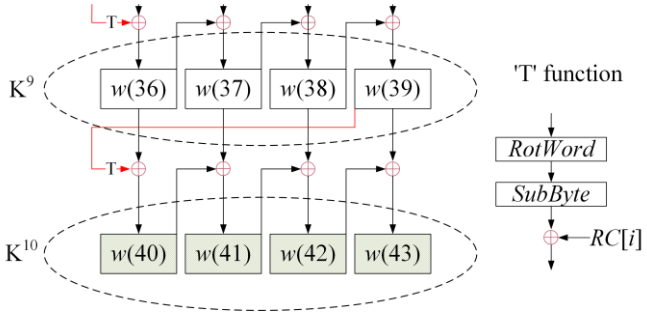

Figure 3. AES–128 key schedule process.

3. DFA on AES

3.1. Fault Model Analysis

In the previous works, the fault model is unfixed, and most of them are according to the fault location and the number of faults induced. We analyze the fault induced at the state, and the effect of the number of fault induction and the fault rounds are jointly considered. If the induced faults are in bytes, the number of induced faults ranges from 1 to 16, and the fault round ranges from 1 to R.

In the beginning, the fault round is analyzed, due to the number of fault rounds determining whether the final output can be used to obtain the key information, if the number of induction rounds is not appropriate, the output information is not available. It is assumed that the attacker can induce τ–byte ($1 \leq \tau \leq R$) fault at round ψ ($1 \leq \psi \leq R$), and obtain the number of differential fault equations, represented by χ. The three values (τ, ψ, χ) are considered to measure a fault model. A one–byte fault is easy to analyze, the fault will affect four bytes of the output (S^{ψ}), and the value of χ is 4. After the second round, a four–byte fault will affect 16–byte of the output ($S^{\psi+1}$), up to now, all normal values are infected, see Figure 4. In Figure 4, F is a fault induced by the attacker, {a, b, c, d, $a1$, $b1$, $c1$, $d1$, $a2$, $b2$, $c2$, $d2$, $a3$, $b3$, $c3$, $d3$, $a4$, $b4$, $c4$, $d4$} belongs to {1, 2, 3}, and the specific value depends on the location of the fault between *SR* and *MC*.

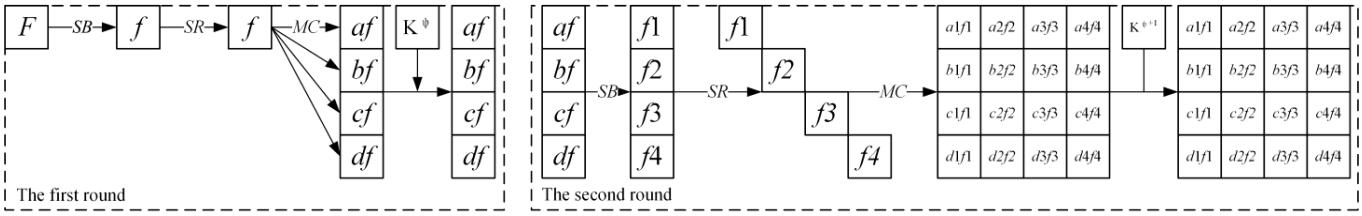

Figure 4. The diagram of one–byte fault diffusion through two rounds.

Therefore, the value of χ is 16. In the encryption process of AES, the last round is missing the *MC* operation, and the *MC* operation plays a diffusion role in the operation, the fault diffusion is also through the *MC* operation. If the next round contains the *MC*, the second round faults will be transformed, and the result will be not available. Thus, the next round can be the last round of encryption, therefore, $\psi = R-2$ is suitable.

Next, the values of τ and χ are considered, according to the above analysis, we list different τ and to obtain the value of χ in Table 3. *Y* means we can recover the key with the fault induction that does not exceed four times, and *N* means not. For example, when $\tau = 5$, after the first round, all values will be affected and there is a column with two faults. Thus, the output will contain five unknown fault values. To eliminate the excess unknowns, it is necessary to induce the fault again to obtain more differential equations. As a result, the number of fault induction times is more than four.

Table 3. The three round values of χ with different τ.

τ	the First Round χ	the Second Round χ	the Third Round χ	Is the Result Available?
1	4	16	16	*Y*
2	8	16	16	*Y*
3	12	16	16	*Y*
4	16	16	16	*Y*
5	16	16	16	*N*
...	...	...	...	...
15	16	16	16	*N*
16	16	16	16	*N*

Here, when $1 \leq \tau \leq 4$, for the first round, all fault values do not affect each other, that is, the fault value does not appear in the same column after SR operation. When $\tau > 4$, the fault value must be affected in the first round, and there will be superfluous unknowns in the result. As a result, the number of fault induction times will increase to offset the excess unknowns. For the AES–128, only the last round–key is recovered, but for the AES–192 and 256, we must obtain the last two round–keys and it is enough to recover the initial key. From Figure and Table, the second round is about the K^{R-1}, therefore, the differential fault equation must be obtained from the SR. When $\tau = 2$, the second round $\chi = 16$, then four-byte information of K^{11} can be obtained for AES–192 and 16 equations are obtained. According to the size of K, 128, 192, and 256 bits, $\tau = 2$ is suitable. AES–128 is flawed since one byte is enough to recover the key when the fault induction times are the same. However, the strengths can be seen in the crack process of AES–192 and 256.

3.2. Fault Model

In the above analysis, the fault model of DFA is typed due to two parameters: the number of faults induced and the fault location (including the difference of fault round at the state or the key schedule).

The multi–byte fault (M0, M1, M2, and M3 in [17]) is assessed before constructing the Dcor fault model. The term 'diagonal' (D_i) refers to a group of four state bytes. D_0, D_1, D_2, and D_3 can therefore be stated as follows:

$$D_0 = \{S_1, S_6, S_{11}, S_{16}\},$$

$$D_1 = \{S_2, S_7, S_{12}, S_{13}\},$$

$$D_2 = \{S_3, S_8, S_9, S_{14}\},$$

$$D_3 = \{S_4, S_5, S_{10}, S_{15}\}.$$

The fault models M0, M1, M2, and M3 are described in [17]. Examples of M0, M1, M2, and M3 are shown in Figure 5.

Figure 5. The example of fault models. (**M0**) The random fault is induced in one of the diagonals; (**M1**) a random non–zero fault occurs at most across two diagonals; (**M2**) a random non–zero fault occurs at most across three diagonals; (**M3**) a random non–zero fault occurs at most across four diagonals.

The Dcor fault model is now introduced. It is assumed that a two–byte fault is corrupted by fault induced at the first column of the R−2 round of AES state, in which the value is random and unknown to the attacker. However, the position of the fault is controlled by the attacker and the fault value can be deduced by a set of equations. This Dcor fault model can be described as inducing two–byte faults at discontiguous rows of state S^{R-2}. Namely, the first two–byte fault is induced at S_1^R and S_9^R. Second, it is induced at S_5^R and S_{13}^R.

Two–byte fault (Dcor fault, see Figure 6) model is used to break AES. There are two phases to breaking AES. First, the fault is induced at S_1^{R-2} and S_9^{R-2}; Then, another fault is induced at S_5^{R-2} and S_{13}^{R-2}. Finally, the source of the fault will infect all 128 bytes of the state S^R. We can obtain the faulty ciphertext C^{1*} and C^{2*}, and the propagation of the Dcor fault is shown in Figures 7 and 8. Without loss of generality, the two–byte Dcor faults

are corrupted as shown in Figure 7. We denote '1′ and '2′ as the random faulty value, homoplastically, '3′ and '4′ is a couple of faults in Figure 8.

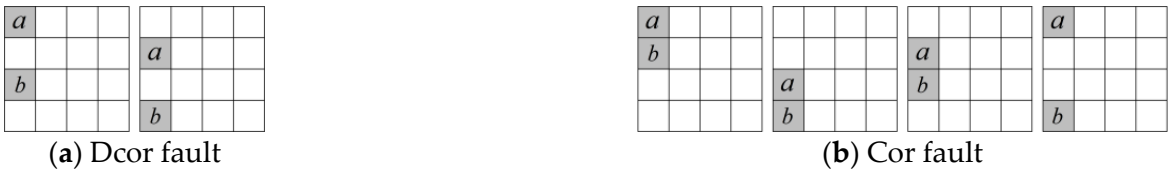

Figure 6. The Dcor and Cor fault model. (**a**) Two–byte faults induced at the first column with discontiguous rows; (**b**) two–byte faults induced at the first column with contiguous rows.

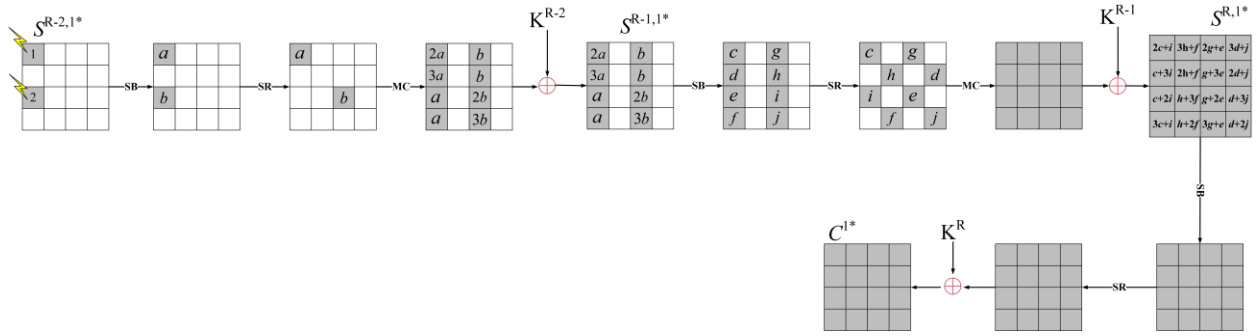

Figure 7. The propagation of the Dcor fault induced at S_1^{R-2} and S_9^{R-2}.

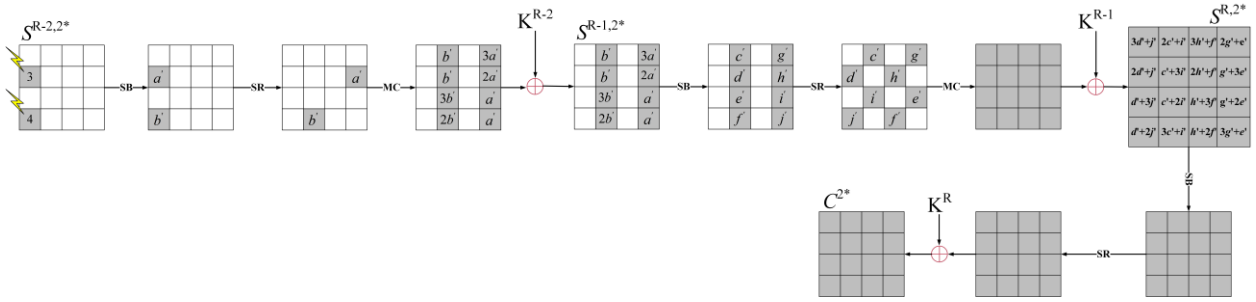

Figure 8. The propagation of the Dcor fault induced at S_5^{R-2} and S_{13}^{R-2}.

{1, 2, 3, 4} is the fault source with unknown values and {a~j, a'~j'} indicates the diffusion value with inequable fault source.

For example:

$$2a = S_1^{R-1,1} \oplus S_1^{R-1,1*}$$
$$3b' = S_{10}^{R-1,1} \oplus S_{10}^{R-1,1*}.$$

The fault {1, 2, 3, 4} are transformed to new values {a, b, a', b'}. Finally, it is transformed {c~j, c'~j'}.

In Figure 7, the fault source '1′ and '2′ is converted to a and b by SB operation, and it affects the eight–byte fault of S^{R-1}. In the next round, a and b will exert eight fault values {c~j}. At the end of round R−1, all bytes of the state are affected by faulty values. We can construct differential equations at the first column of $S^{R,1}$ as follows:

$$\begin{pmatrix} 2 & 1 \\ 1 & 3 \\ 1 & 2 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} c \\ i \end{pmatrix} = \begin{pmatrix} 2c \oplus i \\ c \oplus 3i \\ c \oplus 2i \\ 3c \oplus i \end{pmatrix}.$$

The second, third column and the fourth column is similar. Then, these formulas include all difference values of $S^{R,1}$ after inducing faults.

Note 1: $\Delta S_j^{\mathrm{R},i} \overset{\mathrm{R};i;j}{\rightarrow} \mathrm{K}^{\mathrm{R}}_{j<<a}$.

A differential equation describing the important information can be obtained by analyzing the fault propagation. The equation can be used to gather important relevant data. The expression (1) is defined for the differential information from the previous round. There, $j << a$ express j cycle shifts left a. Such as: $(j = 5) << 2$ $(5 \leq j \leq 8) = 7$.

$$\Delta S_j^{\mathrm{R},i} \overset{\mathrm{R};i;j}{\rightarrow} \mathrm{K}^{\mathrm{R}}_{j<<a}(a = 0,1,2,3) \Leftrightarrow \Delta S_j^{\mathrm{R},i} = \begin{cases} SB^{-1}(C_j^i \oplus \mathrm{K}_j^{\mathrm{R}}) \oplus SB^{-1}(C_j^{i*} \oplus \mathrm{K}_j^{\mathrm{R}}); 1 \leq j \leq 4, a = 0 \\ SB^{-1}(C^i_{j<<1} \oplus \mathrm{K}^{\mathrm{R}}_{j<<1}) \oplus SB^{-1}(C^{i*}_{j<<1} \oplus \mathrm{K}^{\mathrm{R}}_{j<<1}); 5 \leq j \leq 8, a = 1 \\ SB^{-1}(C^i_{j<<2} \oplus \mathrm{K}^{\mathrm{R}}_{j<<2}) \oplus SB^{-1}(C^{i*}_{j<<2} \oplus \mathrm{K}^{\mathrm{R}}_{j<<2}); 9 \leq j \leq 12, a = 2 \\ SB^{-1}(C^i_{j<<3} \oplus \mathrm{K}^{\mathrm{R}}_{j<<3}) \oplus SB^{-1}(C^{i*}_{j<<3} \oplus \mathrm{K}^{\mathrm{R}}_{j<<3}); 13 \leq j \leq 16, a = 3 \end{cases} . \tag{1}$$

Then, by exploiting the above analysis, these equations between differential $S^{\mathrm{R},1}$ and K^{R} can be obtained:

$$\begin{array}{c} 2c \oplus i = \Delta S_1^{\mathrm{R},1} \overset{\mathrm{R};1;1}{\rightarrow} \mathrm{K}_1^{\mathrm{R}}; 3h \oplus f = \Delta S_2^{\mathrm{R},1} \overset{\mathrm{R};1;2}{\rightarrow} \mathrm{K}_2^{\mathrm{R}} \\ c \oplus 3i = \Delta S_5^{\mathrm{R},1} \overset{\mathrm{R};1;5}{\rightarrow} \mathrm{K}_8^{R}; 2h \oplus f = \Delta S_6^{\mathrm{R},1} \overset{\mathrm{R};1;6}{\rightarrow} \mathrm{K}_5^{\mathrm{R}} \\ c \oplus 2i = \Delta S_9^{\mathrm{R},1} \overset{\mathrm{R};1;9}{\rightarrow} \mathrm{K}_{11}^{\mathrm{R}}; h \oplus 3f = \Delta S_{10}^{\mathrm{R},1} \overset{\mathrm{R};1;10}{\rightarrow} \mathrm{K}_{12}^{\mathrm{R}} \\ 3c \oplus i = \Delta S_{13}^{\mathrm{R},1} \overset{\mathrm{R};1;13}{\rightarrow} \mathrm{K}_{14}^{\mathrm{R}};\ h \oplus 2f = \Delta S_{14}^{\mathrm{R},1} \overset{\mathrm{R};1;14}{\rightarrow} \mathrm{K}_{15}^{\mathrm{R}} \end{array}, \tag{2}$$

$$\begin{array}{c} 2g \oplus e = \Delta S_3^{\mathrm{R},1} \overset{\mathrm{R};1;3}{\rightarrow} \mathrm{K}_3^{\mathrm{R}}; 3d \oplus j = \Delta S_4^{\mathrm{R},1} \overset{\mathrm{R};1;4}{\rightarrow} \mathrm{K}_4^{\mathrm{R}} \\ g \oplus 3e = \Delta S_7^{R,1} \overset{\mathrm{R};1;7}{\rightarrow} \mathrm{K}_6^{\mathrm{R}}; 2d \oplus j = \Delta S_8^{\mathrm{R},1} \overset{\mathrm{R};1;8}{\rightarrow} \mathrm{K}_7^{\mathrm{R}} \\ g \oplus 2e = \Delta S_{11}^{\mathrm{R},1} \overset{\mathrm{R};1;11}{\rightarrow} \mathrm{K}_9^{\mathrm{R}}; d \oplus 3j = \Delta S_{12}^{\mathrm{R},1} \overset{\mathrm{R};1;12}{\rightarrow} \mathrm{K}_{10}^{\mathrm{R}} \\ 3g \oplus e = \Delta S_{15}^{\mathrm{R},1} \overset{\mathrm{R};1;15}{\rightarrow} \mathrm{K}_{16}^{\mathrm{R}};\ d \oplus 2j = \Delta S_{16}^{\mathrm{R},1} \overset{\mathrm{R};1;16}{\rightarrow} \mathrm{K}_{13}^{\mathrm{R}} \end{array} . \tag{3}$$

Consequently, the complexity to solve Equation (3) is 2^{96} because there are twelve unknown variables, g, e, d, j, $\mathrm{K}_3^{\mathrm{R}}$, $\mathrm{K}_4^{\mathrm{R}}$, $\mathrm{K}_6^{\mathrm{R}}$, $\mathrm{K}_7^{\mathrm{R}}$, $\mathrm{K}_9^{\mathrm{R}}$, $\mathrm{K}_{10}^{\mathrm{R}}$, $\mathrm{K}_{13}^{\mathrm{R}}$, and $\mathrm{K}_{16}^{\mathrm{R}}$, which is impractical. However, the complexity can be reduced by constructing another equation when faults '3′ and '4′ are induced.

In Figure 8, similarly, these fault values {c'~j'} will entirely diffuse to the state $S^{\mathrm{R},2}$, and the difference value of $S^{\mathrm{R},2}$ ($\Delta S^{\mathrm{R},2}$) can be written.

$$\begin{pmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 3 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} d' \\ j' \end{pmatrix} = \begin{pmatrix} 3d' \oplus j' \\ 2d' \oplus j' \\ d' \oplus 3j' \\ d' \oplus 2j' \end{pmatrix}; \begin{pmatrix} 2 & 1 \\ 1 & 3 \\ 1 & 2 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} c' \\ i' \end{pmatrix} = \begin{pmatrix} 2c' \oplus i' \\ c' \oplus 3i' \\ c' \oplus 2i' \\ 3c' \oplus i' \end{pmatrix}, \tag{4}$$

$$\begin{pmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 3 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} h' \\ f' \end{pmatrix} = \begin{pmatrix} 3h' \oplus f' \\ 2h' \oplus f' \\ h' \oplus 3f' \\ h' \oplus 2f' \end{pmatrix}; \begin{pmatrix} 2 & 1 \\ 1 & 3 \\ 1 & 2 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} g' \\ e' \end{pmatrix} = \begin{pmatrix} 2g' \oplus e' \\ g' \oplus 3e' \\ g' \oplus 2e' \\ 3g' \oplus e' \end{pmatrix}. \tag{5}$$

Exploiting Equations (4) and (5), another key association with these relationships is as follows:

$$\begin{array}{c} 3d' \oplus j' = \Delta S_1^{\mathrm{R},2} \overset{\mathrm{R};2;1}{\rightarrow} \mathrm{K}_1^{\mathrm{R}};\ 2c' \oplus i' = \Delta S_2^{\mathrm{R},2} \overset{\mathrm{R};2;2}{\rightarrow} \mathrm{K}_2^{\mathrm{R}} \\ 2d' \oplus j' = \Delta S_5^{\mathrm{R},2} \overset{\mathrm{R};2;5}{\rightarrow} \mathrm{K}_8^{\mathrm{R}};\ c' \oplus 3i' = \Delta S_6^{\mathrm{R},2} \overset{\mathrm{R};2;6}{\rightarrow} \mathrm{K}_5^{\mathrm{R}} \\ d' \oplus 3j' = \Delta S_9^{\mathrm{R},2} \overset{\mathrm{R};2;9}{\rightarrow} \mathrm{K}_{11}^{\mathrm{R}};\ c' \oplus 2i' = \Delta S_{10}^{\mathrm{R},2} \overset{\mathrm{R};2;10}{\rightarrow} \mathrm{K}_{12}^{\mathrm{R}} \\ d' \oplus 2j' = \Delta S_{13}^{\mathrm{R},2} \overset{\mathrm{R};2;13}{\rightarrow} \mathrm{K}_{14}^{\mathrm{R}};\ 3c' \oplus i' = \Delta S_{14}^{\mathrm{R},2} \overset{\mathrm{R};2;14}{\rightarrow} \mathrm{K}_{15}^{\mathrm{R}} \end{array}, \tag{6}$$

$$\begin{array}{c} 3h' \oplus f' = \Delta S_3^{\mathrm{R},2} \overset{\mathrm{R};2;3}{\rightarrow} \mathrm{K}_3^{\mathrm{R}};\ 2g' \oplus e' = \Delta S_4^{\mathrm{R},2} \overset{\mathrm{R};2;4}{\rightarrow} \mathrm{K}_4^{\mathrm{R}} \\ 2h' \oplus f' = \Delta S_7^{\mathrm{R},2} \overset{\mathrm{R};2;7}{\rightarrow} \mathrm{K}_6^{\mathrm{R}};\ g' \oplus 3e' = \Delta S_8^{\mathrm{R},2} \overset{\mathrm{R};2;8}{\rightarrow} \mathrm{K}_7^{\mathrm{R}} \\ h' \oplus 3f' = \Delta S_{11}^{\mathrm{R},2} \overset{\mathrm{R};2;11}{\rightarrow} \mathrm{K}_9^{\mathrm{R}};\ g' \oplus 2e' = \Delta S_{12}^{\mathrm{R},2} \overset{\mathrm{R};2;12}{\rightarrow} \mathrm{K}_{10}^{\mathrm{R}} \\ h' \oplus 2f' = \Delta S_{15}^{\mathrm{R},2} \overset{\mathrm{R};2;15}{\rightarrow} \mathrm{K}_{16}^{\mathrm{R}};\ 3g' \oplus e' = \Delta S_{16}^{\mathrm{R},2} \overset{\mathrm{R};2;16}{\rightarrow} \mathrm{K}_{13}^{\mathrm{R}} \end{array} . \tag{7}$$

From (2) and (6), for the corresponding byte of K^R, such as $\{K_1^R, K_8^R, K_{11}^R, K_{14}^R\}$ and $\{c, i, d', j'\}$, there are eight equations, thus, $\{K_1^R, K_8^R, K_{11}^R, K_{14}^R\}$ can be retrieved, and other information about the key can be found.

Algorithm 1 Recovering the last round–key of AES.

Input: The PCFCs (C^1, C^{1*}) and (C^2, C^{2*}).
Output: K^R.

Construct $\Delta S_j^{R,1} \overset{R;1;j}{\rightarrow} K_{j\ll a}^R$ and $\Delta S_j^{R,2} \overset{R;2;j}{\rightarrow} K_{j\ll a}^R$ (j = 1, 5, 9, 13) and find $\{K_1^R, K_8^R, K_{11}^R, K_{14}^R\}$.
For the other three columns of ΔS^R, construct similar $\Delta S_j^{R,i}$ and solve them.

Return K^R.

For AES–128, the initial key can be found by algorithm 1, it is not enough to break AES–192 and AES–256. Since the length of the key is inconsistent, breaking AES–192 and 256 is needed to retrieve the right values of K^{11} and K^{13}, respectively.

When K^R is known, and two equations can be exploited to break AES–192 and AES–256. For AES–192:

$$K_j^{R-1} = K_{j+1}^R \oplus K_{j+2}^R \ (j = 1,\ 2,\ 5,\ 6,\ 9,\ 10,\ 13,\ 14). \tag{8}$$

For AES–192 and AES–256, the pairs of correct and fault states (PCFSs) $S^{R,i}$ and $S^{R,i*}$ can be obtained.

$$\begin{array}{c} C^i = SB(SR(S^{R,i})) \oplus K^R \\ C^{i*} = SB(SR(S^{R,i*})) \oplus K^R \end{array}. \tag{9}$$

Note 2: $\Delta S_j^{R-1,i} \overset{R-1;i;j}{\rightarrow} K_{j<<a}^{R-1}$.

For a detailed explanation, see Formula (10). If the round–key for the final round of AES encryption has been obtained, it is possible to decrypt the penultimate round via reverse analysis.

$$\Delta S_j^{R-1,i} \overset{R-1;i;j}{\rightarrow} K_{j<<a}^{R-1}(a = 0,1,2,3) \Leftrightarrow \Delta S_j^{R-1,i} = \begin{cases} MC^{-1}[SB^{-1}(S_j^{R,i} \oplus K_j^{R-1})] \oplus MC^{-1}[SB^{-1}(S_j^{R,i*} \oplus K_j^{R-1})], 1 \le j \le 4 \\ MC^{-1}[SB^{-1}(S_{j<<1}^{R,i} \oplus K_{j<<1}^{R-1})] \oplus MC^{-1}[SB^{-1}(S_{j<<1}^{R,i*} \oplus K_{j<<1}^{R-1})], 5 \le j \le 8 \\ MC^{-1}[SB^{-1}(S_{j<<2}^{R,i} \oplus K_{j<<2}^{R-1})] \oplus MC^{-1}[SB^{-1}(S_{j<<2}^{R,i*} \oplus K_{j<<2}^{R-1})], 9 \le j \le 12 \\ MC^{-1}[SB^{-1}(S_{j<<3}^{R,i} \oplus K_{j<<3}^{R-1})] \oplus MC^{-1}[SB^{-1}(S_{j<<3}^{R,i*} \oplus K_{j<<3}^{R-1})], 13 \le j \le 16 \end{cases}. \tag{10}$$

In the next section, we would make use of the Dcor fault model to outright break AES–192 and AES–256.

3.3. Proposed Attack on AES–192

For AES–192, the Dcor fault model mentioned (see Section 3.2) is utilized to crack AES–192, K^{12} and the right half of K^{11} should be recovered. Firstly, we induced the fault at S_1^{10} and S_9^{10}. Secondly, we induced the fault at S_5^{10} and S_{13}^{10}. Different from AES–128, K^{11} should be recovered, that says, the unknown values (a, b, a' and b' in $S^{R-1,i*}$) could be found to retrieve the right part of K^{11} $\{w(46), w(47), w(48), w(49), w(50), w(51)\}$ (the green part of Figure 9). Figure 9 only shows the process of the last two rounds' key schedules of AES–192, the key is 192 bits, and it contains all information of the initial key, therefore, the initial key can be recovered according to the last two round–key.

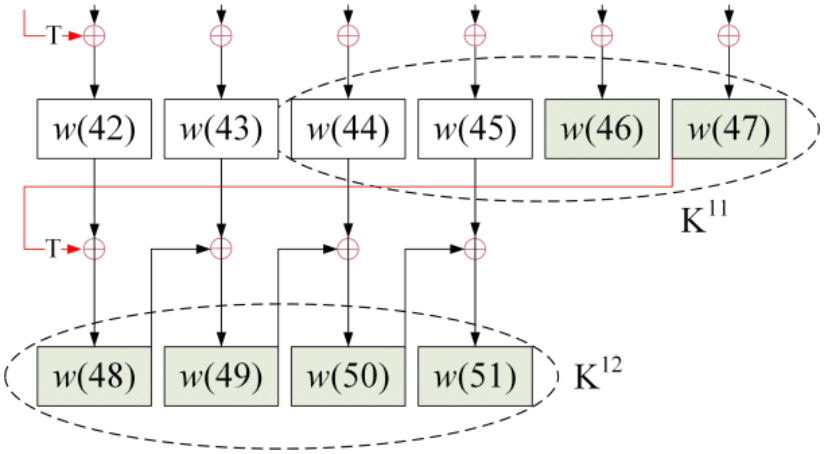

Figure 9. AES–192 key schedule process.

As well, the PCFSs (pairs of correct and fault output of the penultimate round) are obtained for the attacker by using Equation (9). Therefore, the value of a, b, a' and b' in Figures 6 and 7 is expressed as:

$$\begin{array}{l} 2a = \Delta S_1^{11,1} \overset{11;1;1}{\rightarrow} \mathrm{K}_1^{11};\ b = \Delta S_3^{11,1} \overset{11;1;3}{\rightarrow} \mathrm{K}_3^{11} \\ 3a = \Delta S_5^{11,1} \overset{11;1;5}{\rightarrow} \mathrm{K}_8^{11};\ b = \Delta S_7^{11,1} \overset{11;1;7}{\rightarrow} \mathrm{K}_6^{11} \\ a = \Delta S_9^{11,1} \overset{11;1;9}{\rightarrow} \mathrm{K}_{11}^{11};\ 2b = \Delta S_{11}^{11,1} \overset{11;1;11}{\rightarrow} \mathrm{K}_9^{11} \\ a = \Delta S_{13}^{11,1} \overset{11;1;13}{\rightarrow} \mathrm{K}_{14}^{11};\ 3b = \Delta S_{15}^{11,1} \overset{11;1;15}{\rightarrow} \mathrm{K}_{16}^{11} \end{array}, \tag{11}$$

$$\begin{array}{l} b' = \Delta S_2^{11,2} \overset{11;2;2}{\rightarrow} \mathrm{K}_2^{11};\ 3a' = \Delta S_4^{11,2} \overset{11;2;4}{\rightarrow} \mathrm{K}_4^{11} \\ b' = \Delta S_6^{11,2} \overset{11;2;6}{\rightarrow} \mathrm{K}_5^{11};\ 2a' = \Delta S_8^{11,2} \overset{11;2;8}{\rightarrow} \mathrm{K}_7^{11} \\ 3b' = \Delta S_{10}^{11,2} \overset{11;2;10}{\rightarrow} \mathrm{K}_{12}^{11};\ a' = \Delta S_{12}^{11,2} \overset{11;2;12}{\rightarrow} \mathrm{K}_{10}^{11} \\ 2b' = \Delta S_{14}^{11,2} \overset{11;2;14}{\rightarrow} \mathrm{K}_{15}^{11};\ a' = \Delta S_{16}^{11,2} \overset{11;2;16}{\rightarrow} \mathrm{K}_{13}^{11} \end{array}. \tag{12}$$

Algorithm 2 outlines the process for cracking AES–192. The values of a, b, a', b' can be obtained using Equations (11) and (12). The remaining key information for K^{11} can then be determined. For AES–192, all keys can be solved. In comparison to cracking AES–128, the penultimate round is used to find the right portion of K^{11}, and two PCFCs are required for cracking AES–128 and AES–192.

Algorithm 2 DFA on AES–192.

Input: The PCFCs $\left(C^{\mathrm{I}},\ C^{i*}\right)$.
Output: K^{12} and the right part of K^{11}.

1. Construct $\Delta S_j^{12,1} \overset{12;1;j}{\rightarrow} \mathrm{K}_{j \ll a}^{12}$ and $\Delta S_j^{12,2} \overset{12;2;j}{\rightarrow} \mathrm{K}_{j \ll \mathrm{a}}^{12}$ (j = 1, 5, 9, 13) and find $\left\{\mathrm{K}_1^{12}, \mathrm{K}_8^{12}, \mathrm{K}_{11}^{12}, \mathrm{K}_{14}^{12}\right\}$.
2. For the other three columns of ΔS^{R}, construct similar $\Delta S_j^{\mathrm{R},\mathrm{i}}$ and solve them. Finally, K^{12} is obtained.
3. Find the left part of K^{11} with Equation (8).
4. Find the PCFSs $\left(S^{11,i},\ S^{11,i*}\right)$ with Equation (9).
5. With equations (11) and (12), $\{a, b, a', b'\}$ can be solved, and the right part of K^{11} is known to the attacker.

Return K^{12} and the right part of K^{11}, namely, $\{w(46), w(47), w(48), w(49), w(50), w(51)\}$.

3.4. Proposed Attack on AES–256

The process of cracking AES–192 through the fault model is described in Section 3.2 and Section 3.3. The difference between AES–192 and AES–256 is that the key length

is distinct. However, if the attacker wants to recover the key of AES–256, the last two round–keys must be found. For AES–256, K^{14} and K^{13} contains all information of the initial key, and the initial key could be recovered (the green part of Figure 10).

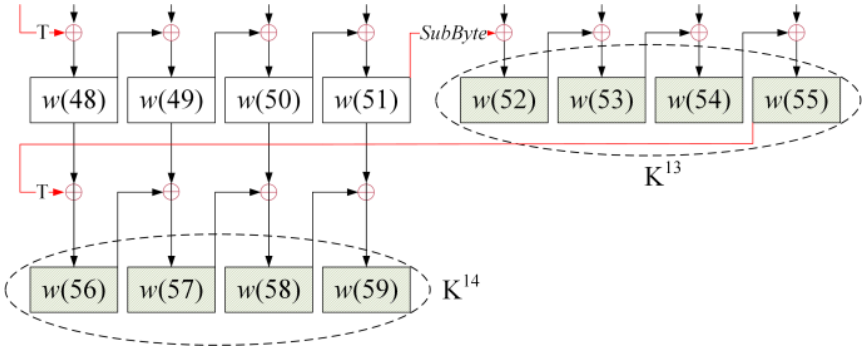

Figure 10. AES–256 key schedule process.

The first stage is to crack the last round–key and then use the information from the penultimate round state to find the penultimate round–key. Algorithm 1 outlines the method of cracking the last round–key, making it easy to find K^{14}. Figure 10 only shows the process of the last round–key schedule of AES–256, the key is 256 bits, and it contains all information of the initial key, therefore, the initial key can be recovered according to the last two round–key.

The penultimate round $S^{R,i*}$ contains the unknown value a, b, a', and b', but the equations can be listed as follows:

$$\begin{array}{l} 2a = \Delta S_1^{13,1} \overset{13;1;1}{\rightarrow} K_1^{13};\ b = \Delta S_3^{13,1} \overset{13;1;3}{\rightarrow} K_3^{13} \\ 3a = \Delta S_5^{13,1} \overset{13;1;5}{\rightarrow} K_8^{13};\ b = \Delta S_7^{13,1} \overset{13;1;7}{\rightarrow} K_6^{13} \\ a = \Delta S_9^{13,1} \overset{13;1;9}{\rightarrow} K_{11}^{13};\ 2b = \Delta S_{11}^{13,1} \overset{13;1;11}{\rightarrow} K_9^{13} \\ a = \Delta S_{13}^{13,1} \overset{13;1;13}{\rightarrow} K_{14}^{13};\ 3b = \Delta S_{15}^{13,1} \overset{13;1;15}{\rightarrow} K_{16}^{13} \end{array}, \tag{13}$$

$$\begin{array}{l} b' = \Delta S_2^{13,2} \overset{13;2;2}{\rightarrow} K_2^{13};\ 3a' = \Delta S_4^{13,2} \overset{13;2;4}{\rightarrow} K_4^{13} \\ b' = \Delta S_6^{13,2} \overset{13;2;6}{\rightarrow} K_5^{13};\ 2a' = \Delta S_8^{13,2} \overset{13;2;8}{\rightarrow} K_7^{13} \\ 3b' = \Delta S_{10}^{13,2} \overset{13;2;10}{\rightarrow} K_{12}^{13};\ a' = \Delta S_{12}^{13,2} \overset{13;2;12}{\rightarrow} K_{10}^{13} \\ 2b' = \Delta S_{14}^{13,2} \overset{13;2;14}{\rightarrow} K_{15}^{13};\ a' = \Delta S_{16}^{13,2} \overset{13;2;16}{\rightarrow} K_{13}^{13} \end{array}. \tag{14}$$

In comparison to AES–192, the values of a, b, a', and b' cannot be directly calculated as they are variables with a value range of 0 to 255. Therefore, an exhaustive search of 2^{32} attempts is necessary to recover the key K^{13} after inducing faults twice at S^{R-2}. If faults ('5' and '6') are induced at S_1^{12} and S_9^{12} (as shown in Figure 11), equations related to $\{K_1^{13}, K_8^{13}, K_{11}^{13}, K_{14}^{13}\}$ and $\{K_3^{13}, K_6^{13}, K_9^{13}, K_{16}^{13}\}$ can be obtained, thereby reducing the exhaustive search to 2^{16} attempts once $\{a, a''\}$ and $\{b, b''\}$ is solved.

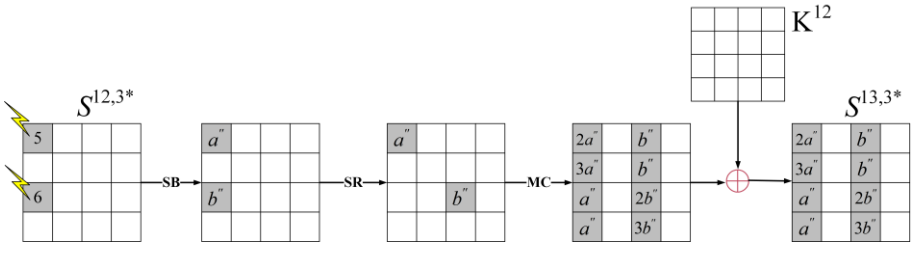

Figure 11. The propagation of faults '7' and '8' induced at S_1^{12} and S_9^{12}.

However, to crack AES–256 without exhaustive search, faults '7′ and '8′ must be induced at S_5^{12} and S_{13}^{12} (as shown in Figure 12). This allows us to obtain equations related to $\{K_2^{13}, K_5^{13}, K_{12}^{13}, K_{15}^{13}\}$ and $\{K_4^{13}, K_7^{13}, K_{10}^{13}, K_{13}^{13}\}$. The method to crack AES–256 is shown in Algorithm 3.

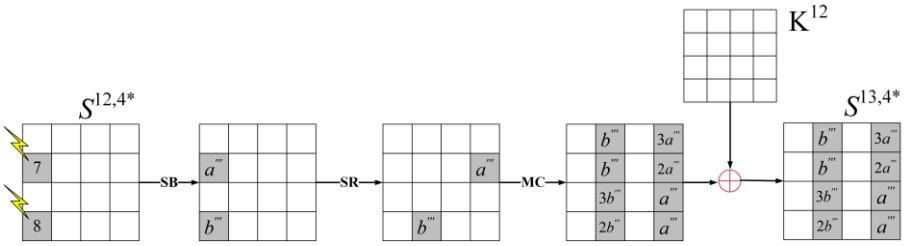

Figure 12. The propagation of faults '7′ and '8′ induced at S_5^{12} and S_{13}^{12}.

Algorithm 3 DFA on AES–256.

Input: The PCFCs $\left(C^i,\ C^{i*}\right)$ i = 2, 3, and 4.
Output: K^{14} and K^{13}.

1. Find the last round–key K^{14} according to algorithm 1.
2. Find the PCFSs $\left(S^{R,i}, S^{R,i*}\right)$ with Equation (9).
3. With equations (13) and (14), $\{a, b, a', b'\}$ cannot be solved. There are 2^{32} exhaustive research to crack AES–256.
4. The range of exhaustive research could be reduced from 2^{32} to 2^{16} by injecting faults '5′ and '6′ at S^{12}.
5. Based on the above conditions, the attacker injects faults '7′ and '8′ at S^{12}. The unknow values $\{a, a', a'', a''', b, b', b'', b'''\}$ could be computed without exhaustive research.

Return K^{14} and K^{13}.

4. Simulation Result and Discussion

4.1. Experimental Result

The DFA on AES based on the Dcor fault model was discussed in this work, and it was implemented using Dev–C++ on a PC with a 3.20 GHz Intel processor and 16 GB of RAM. The experimental results showed that the Dcor fault model could improve the cracking efficiency of AES.

As shown in Table 4, we summarized DFA on AES and contrasted the multi–byte fault models in terms of the number of PCFCs, the depth of fault injection, and the application of the models. Although alternative fault models can be used to break AES [22], they are not fixed, with M1 having a limit of eight–byte faults and M2 having a limit of twelve–byte faults. In [22], M1 is used to break AES, while the Dcor fault model requires fewer PCFCs and thorough searches for AES–192 and 256. While the multi–byte fault models used in [13,21,26] have not been expanded to include more AES variations, the Dcor fault model proposed can be utilized to recover three different AES versions.

Additionally, the depth of the fault injection is another factor in cracking AES. In Table 4, such as [19], two variants were cracked of AES by injecting faults between R–2 and R−3. Another paper [13] induced faults at K^9 for AES–128 and at K^{R-1} and K^{R-2} for AES–192 and AES–256. The location of the fault induced at S^{R-2} with the Dcor model proposed in this paper does not require changing the fault induction conditions, such as voltage and frequency. Considering the required PCFCs and model scalability, we propose that the Dcor model is more efficient and applicable to all forms of AES.

Table 4. Comparison with existing DFA on AES.

Type	Ref	Fault Model	Fault Round	No. of Faults	Exhaustive Search
AES–128	[13]	Dcor	9	2	1
	[21]	M0	Between 7 and 8	4	1
		M1	Between 7 and 8	2	1
		M2	Between 7 and 8	1	1
	[22]	M1	Between 7 and 8	2	1
		M2	Between 7 and 8	3	1
	[26]	Multi–fault	9	≈2.17	1
	Our work	Dcor	8	2	1
AES–192	[19]	Method1	10 and 11	12	1
		Method2	10 and 11	≈3000	1
	[22]	M1	Between 9 and 10	2	2^8
		M2	Between 9 and 10	3	2^{32}
	Our work	Dcor	10	2	1
AES–256	[19]	Method1	12 and 13	12	1
		Method2	12 and 13	≈3000	1
	[22]	M1	Between 11 and 12	3	2^{32}
		M2	Between 11 and 12	4	1
	Our work	Dcor	12	2	2^{32}
				3	2^{16}
				4	1

4.2. Result Discussion

Accounting for the fault model, a two–byte fault model (Dcor fault model) is proposed to break AES. Especially, for AES–192, two faulty ciphertexts are required, the penultimate round data (S^{12}) is unpredictable when the last round–key (K^{12}) is indistinct. However, after obtaining the K^{12}, the decryption operation is performed to recover S^{12}. Additionally, observe the fault in S^{11}, two columns (eight–byte) are corrupted by a and b (or a' and b'), eight related equations are listed and there are six unknown values (including four values of round–key and two fault values), thus the only solution is determinable. For AES–256, since the final two round–keys in AES–256 are independent of one another, two additional fault ciphertexts are necessary. Thus, the key schedule cannot be used to acquire more equations when cracking AES–256.

5. Conclusions

This paper presents an efficient DFA for AES using a two–byte fault induced in the state S^{R-2} with discontiguous rows (Dcor fault model). The proposed method can retrieve the AES–128, 192, and 256 with two, two, and four pairs of correct and fault ciphertexts (PCFCs), respectively. Furthermore, when considering the exhaustive searches for AES–256, it is possible to extract AES–256 using two (or three) PCFCs and 2^{32} (or 2^{16}) exhaustive searches. Compared with these existing multi–byte fault models, the Dcor fault model proposed needs fewer PCFCs and reduces the exhaustive searches of keys when using identical PCFCs.

The Dcor fault model could be extended to additional SPN ciphers like RC6 and SM4, which will be the subject of our future study, even if it only pertains to AES in this paper.

Author Contributions: Conceptualization, X.W. and J.Z.; methodology, X.W. and J.Z.; software, X.W.; validation, X.W., J.Z., and S.C.; formal analysis, X.W., J.Z., and W.W.; resources, J.W. and J.Z.; data curation, X.W.; writing—original draft preparation, X.W.; writing—review and editing, X.W. and J.Z.; supervision, S.C. and J.Z.; project administration, J.W. and J.Z.; funding acquisition, J.W., J.Z. and W.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Natural Science Research in Colleges of Jiangsu Province under Grant 21KJB520040 and 22KJD520006; the Basic Science Research Project of Nantong under Grant JC2020143.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The author would like to thank the anonymous reviewers for their valuable suggestions and comments that improved the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jain, R. *Advanced Encryption Standard (AES). Federal Information Processing Standard*; Springer: Berlin/Heidelberg, Germany, 2001. [CrossRef]
2. Toughi, S.; Fathi, M.H.; Sekhavat, Y.A. An image encryption scheme based on elliptic curve pseudo random and Advanced Encryption System. *Signal Process.* **2017**, *141*, 217–227. [CrossRef]
3. Shi, L. Design and Implementation of WiFi Security Intelligent Check-in System Encrypted by AES. *J. Anhui Univ. Sci. Technol. Nat. Sci.* **2019**, *39*, 56–59.
4. Hashim, A.T.; Jabbar, A.K.; Hassan, Q.F. Medical Image Encryption Based on Hybrid AES with Chaotic Map. *J. Physics Conf. Ser.* **2021**, *1973*, 12–37. [CrossRef]
5. Mehmood, G.; Khan, M.Z.; Abbas, S.; Faisal, M.; Rahman, H.U. An Energy-Efficient and Cooperative Fault- Tolerant Communication Approach for Wireless Body Area Network. *IEEE Access* **2020**, *8*, 69134–69147. [CrossRef]
6. Scripcariu, L.; Burdia, D.; Diaconu, F. FPGA Synthesis of an AES Encoder Circuit for Vehicular Communication Networks. In Proceedings of the 2021 International Symposium on Signals, Circuits and Systems (ISSCS), Iasi, Romania, 15–16 July 2021; pp. 1–4. [CrossRef]
7. Ramya, T.; Ramya, G.; Raju, K.; Ravi, J.; Verma, D. An Efficient AES Algorithm for Cryptography Using VLSI. *ECS Trans.* **2022**, *107*, 5605–5612. [CrossRef]
8. Luminiţa, S.; Andreea-Elena, B.; Petre-Daniel, M. Improved C-Language Implementation of AES Algorithm for WSN. In Proceedings of the 2021 International Symposium on Signals, Circuits and Systems (ISSCS), Iasi, Romania, 15–16 July 2021; pp. 1–4. [CrossRef]
9. Biham, E.; Shamir, A. Differential Fault Analysis of Secret Key Cryptosystems. In Proceedings of the International Cryptology Conference, Santa Barbara, CA, USA, August 15–19 1999; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1924, pp. 513–525. [CrossRef]
10. 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 & Privacy, Wollongong, Australia, 9–11 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2727, pp. 118–129. [CrossRef]
11. Kim, C.H.; Quisquater, J.-J. New Differential Fault Analysis on AES Key Schedule: Two faults are enough. In Proceedings of the Proceedings 8th IFIP WG 8.8/11.2 International Conference, London, UK, 8–11 September 2008; (CARDIS2008); Volume 5189, pp. 48–60. [CrossRef]
12. Ali, S.S.; Mukhopadhyay, D. Differential Fault Analysis of AES–128 Key Schedule Using a Single Multi-byte Fault. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Leuven, Belgium, 14–16 September 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 50–64. [CrossRef]
13. Zhang, J.; Wu, N.; Li, J.; Zhou, F. A novel differential fault analysis using two-byte fault model on AES Key schedule. *IET Circuits Devices Syst.* **2019**, *13*, 661–666. [CrossRef]
14. Kim, C.H. Improved Differential Fault Analysis on AES Key Schedule. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 41–50. [CrossRef]
15. Floissac, N.; L'Hyver, Y. From AES–128 to AES–192 and AES–256, How to Adapt Differential Fault Analysis Attacks on Key Expansion. In Proceedings of the 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, Nara, Japan, 28 September 2011; pp. 43–53. [CrossRef]
16. Kiranmayee, T.S.; Maniraj, S.P.; Thakur, A.; Bhagyashree, M.; Gupta, R. Analyzing DFA Attack on AES–192. In *Lecture Notes on Data Engineering and Communications Technologies*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 31, pp. 211–218. [CrossRef]
17. Moradi, A.; Shalmani, M.T.M.; Salmasizadeh, M. A Generalized Method of Differential Fault Attack Against AES Cryptosystem. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Yokohama, Japan, 10–13 October 2006; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4249, pp. 91–100. [CrossRef]
18. Han, L.; Wu, N.; Ge, F.; Zhou, F.; Wen, J.; Qing, P. Differential Fault Attack for the Iterative Operation of AES–192 Key Expansion. In Proceedings of the 2020 IEEE 20th International Conference on Communication Technology (ICCT), Nanning, China, 28–31 October 2020; IEEE: Piscataway, NJ, USA; pp. 1156–1160. [CrossRef]

19. Li, W.; Gu, D.; Wang, Y.; Li, J.; Liu, Z. An Extension of Differential Fault Analysis on AES. In Proceedings of the International Conference on Network and System Security, Los Alamitos, CA, USA, 19 October 2009; IEEE Computer Society: Washington, DC, USA, 2009; pp. 443–446. [CrossRef]
20. Piret, G.; Quisquater, J.-J. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Cologne, Germany, 8–10 September 2003; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2779, pp. 77–88. [CrossRef]
21. Saha, D.; Mukhopadhyay, D.; Chowdhury, D.R. A Diagonal Fault Attack on the Advanced Encryption Standard. *Cryptol. Eprint Arch.* **2009**. Report2009/581.
22. Kim, C.H. Differential fault analysis of AES: Toward reducing number of faults. *Inf. Sci.* **2012**, *199*, 43–57. [CrossRef]
23. Prior, S.; Maciver, D.; Forsyth, K.; Walsh, M.; Meiklejohn, A.; Irvine, L. Differential Fault Analysis of the Advanced Encryption Standard Using a Single Fault. *Community Ment. Health J.* **2013**, *49*, 658–667. [CrossRef] [PubMed]
24. Barenghi, A.; Bertoni, G.M.; Breveglieri, L.; Pellicioli, M.; Pelosi, G. Low voltage fault attacks to AES. In Proceedings of the 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), Anaheim, CA, USA, 14 June 2010; pp. 7–12. [CrossRef]
25. Kim, C.H. Differential Fault Analysis against AES–192 and AES–256 with Minimal Faults. In Proceedings of the 2010 Workshop on Fault Diagnosis and Tolerance in Cryptography, Santa Barbara, CA, USA, 21 August 2010; IEEE Computer Society: Washington, DC, USA; pp. 3–9. [CrossRef]
26. Liao, N.; Cui, X.; Liao, K.; Wang, T.; Yu, D.; Cui, X. Improving DFA attacks on AES with unknown and random faults. *Sci. China Inf. Sci.* **2017**, *60*, 166–179. [CrossRef]
27. Liu, Y.; Cui, X.; Cao, J.; Zhang, X. A hybrid fault model for differential fault attack on AES. In Proceedings of the 2017 IEEE 12th International Conference on ASIC (ASICON), Guiyang, China, 25–28 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 784–787. [CrossRef]
28. Long, M.; Kong, M.; Long, S.; Zhang, X. An Improved Differential Fault Analysis on Block Cipher KLEIN-64. *Comput. Mater. Contin.* **2020**, *65*, 1425–1436. [CrossRef]
29. Xiao, H.; Wang, L. The differential fault analysis on block cipher KLEIN-96. *J. Inf. Secur. Appl.* **2022**, *67*, 103205. [CrossRef]
30. Anand, R.; Siddhanti, A.; Maitra, S. Differential Fault Attack on SIMON with Very Few Faults; Progress in Cryptology-INDOCRYPT. In Proceedings of the 19th International Conference on Cryptology in India, New Delhi, India, 9–12 December 2018; pp. 107–119. [CrossRef]
31. Zhang, J.; Wang, J.; Bin, G.; Li, J. An efficient differential fault attack against SIMON key schedule. *J. Inf. Secur. Appl.* **2022**, *66*, 103155. [CrossRef]
32. Nalla, V.; Sahu, R.A.; Saraswat, V. Differential Fault Attack on SIMECK. In Proceedings of the Third Workshop on Cryptography and Security in Computing Systems, Prague, Czech Republic, 20 January 2016; pp. 45–48. [CrossRef]
33. Le, D.-P.; Lu, R.; Ghorbani, A.A. Improved fault analysis on SIMECK ciphers. *J. Cryptogr. Eng.* **2022**, *12*, 169–180. [CrossRef]