Improved Differential Fault Analysis on AES Key Schedule

Chong Hee Kim

***Abstract*—Differential fault analysis (DFA) finds the key of a block cipher using differential information between correct and faulty ciphertexts obtained by inducing faults during the computation of ciphertexts. Among many ciphers, advanced encryption standard (AES) has been the main target of DFA due to its popularity. The naive implementation of AES is known to be vulnerable to DFA, which can be split into two categories depending on the fault location: the DFA on the State and the DFA on the Key Schedule. For the first category, much research has been done and very efficient methods were devised. However, there is still a lack of research in the second category. The advantage of DFA on the Key Schedule is that it can even defeat some fault-protected AES implementations. Research on DFA has been diversified into several directions: reducing the number of required faults, changing fault models (from one-byte fault to multibyte fault and *vise versa*), extending to AES-192 and AES-256, and exploiting faults induced at an earlier round. This paper deals with all these directions together in DFA on AES Key Schedule. We introduce new attacks that find the AES-128 key with two faults in a one-byte fault model without exhaustive search and the AES-192 and the AES-256 keys with six and four faults, respectively.**

***Index Terms*—Advanced encryption standard (AES), block ciphers, cryptanalysis, differential fault analysis (DFA).**

I. Introduction

WE CAN easily find cryptographic hardware devices such as smart cards everywhere in our daily lives from banking cards to subscriber identity module (SIM) cards for global system for mobile (GSM) communications. One of the main reasons why these cryptographic hardware devices are widely used is that they are believed to be tamper-resistant. This is why they host the implementation of many cryptographic protocols. However, recent development of physical attacks shows that a naive implementation of cryptographic protocols does not provide security anymore. From a classical point of view, cryptanalysis is an abstract mathematical notion. However, in practice algorithms have to be implemented on real physical devices that are exposed to side-channel attacks like timing attacks [8], [11], power attacks [20], or electromagnetic attacks [28] as well as fault attacks [3].

Manuscript received December 10, 2010; revised June 09, 2011; accepted June 27, 2011. Date of publication July 05, 2011; date of current version January 13, 2012. This work was supported in part by the Walloon Region Marshall plan through the SPW DG06 Project TRASILUX. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Kouichi Itoh.

The author is with the Information Security Group, ICTEAM Institute, Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgium (e-mail: chhkim7@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2011.2161289

Differential fault analysis (DFA) is one of the fault attacks used to break block ciphers using differential information between correct and faulty ciphertexts. An attacker gets faulty ciphertexts by giving external impact on a device with voltage variation, glitch, laser, etc. [3]. The first DFA presented by Biham and Shamir in 1997 [6] targeted DES [1]. Afterward many people tried to break several cryptosystems such as Triple-DES [16], RC4 [5], [17], CLEFIA [10], [32], advanced encryption standard (AES) [4], [7], [9], [12], [15], [18], [19], [23], [24], [27], [31], [33], [34], SMS4, and MacGuffin [21].

DFA on AES can be split into two categories depending on the fault location: the *DFA on the State* and the *DFA on the Key Schedule*. Furthermore, research on DFA has been diversified into several directions: reducing the number of required faults, changing fault models (from one-byte fault model to multibyte fault model and *vise versa*), extending to AES-192 and AES-256, and exploiting faults induced at an earlier round. The first direction has been most actively researched because the attacker can perform the attack more easily as the required number of faults decreases. Piret and Quisquater showed that two faults induced on the AES State are sufficient to find the AES-128 key in a one-byte fault model [27]. In 2009, it was shown that the AES-128 key could be deduced with one fault and exhaustive search of 2^{32} candidates by Fukunaga and Takahashi [14] and Li *et al.* [22], respectively. In 2010, Tunstall and Mukhopadhyay demonstrated that the exhaustive search could be further reduced to 2^8 [35].

The extension to AES with 192 and 256-bit key (AES-192 and AES-256, respectively) becomes important in recent years because AES-192 and AES-256 have been deployed more and more. In 2009, Li *et al.* [22] proposed two DFA on AES-192 and AES-256 based on the Moradi *et al.*'s DFA on AES-128 [23]. The first method retrieved the key with 16 pairs of correct and faulty ciphertexts and the second method needs 3000 pairs. Barenghi *et al.* [4] showed that the AES-192 key (and the AES-256 key) could be retrieved with 16 pairs of correct and faulty ciphertexts. Takahashi and Fukunaga showed that they could retrieve the AES-192 key using three pairs of correct and faulty ciphertexts and the AES-256 key using two pairs of correct and faulty ciphertexts and two pairs of correct and faulty plaintexts [33]. In [18], Kim showed that the AES-192 key could be found with two pairs of correct and faulty ciphertexts and AES-256 key could be found with three pairs.

Moradi *et al.* proposed DFA in multibyte fault models where several bytes are assumed to be corrupted together in 2006 [23]. In 2009, Saha *et al.* categorized multibyte fault models in detail and presented more efficient DFA [30], where two or four pairs of correct and faulty ciphertexts are required. The last research direction is the utilization of the faults induced at an earlier round. The general countermeasure of DFA is to protect the last few

rounds. As redundancy is costly, one should ascertain exactly which rounds need to be protected. Phan and Yen first showed that the faults induced one or two rounds earlier than other DFA could be used to find the key [26]. In DFA of DES, Rivain showed that the faults in the middle rounds could be used [29].

A different form of DFA, targeting the AES Key Schedule, was introduced by Giraud in [15] and improved by Chen and Yen [9]. However, these attacks target both the key schedule and the intermediate states and require too many faults. Later, Peacham and Thomas proposed DFA on AES Key Schedule [25] by assuming random faults only on the AES Key Schedule, that reduced the number of correct and faulty ciphertexts to 12 pairs. Takahashi *et al.* generalized Peacham and Thomas's attack and succeeded in recovering the key with seven pairs [34]. Kim and Quisquater showed that they could find the AES-128 key with two pairs with exhaustive search of 2^{32} or four pairs without it [19]. There is only one result on DFA on the Key Schedule of AES-192 and AES-256 [13] that requires 16 pairs.

The general countermeasure against DFA on the AES State is to recompute the last few rounds of AES and compare it with the original output [26], [29]. However, if the key schedule is not redone for recomputation, it cannot prevent DFA on the AES Key Schedule. Therefore, the advantage of DFA on the Key Schedule is that it can defeat some fault-protected AES implementations where the round keys are not rescheduled prior to the check.

The slow progress of DFA on AES Key Schedule comes from the complex propagation of an error that affects both states and round keys. In this paper, we try to reduce the gap between two categories (the DFA on the State and the DFA on the Key Schedule) by proposing new efficient methods in DFA on AES Key Schedule. Our methods have the following advantages compared to the previous works:

1) The smallest faults are required: two faults for AES-128, four to six faults for AES-192, four faults for AES-256. Our results require the smallest faults in DFA on AES Key Schedule. As our attack corrupts the "key scheduling process," the faulty value affects both the round keys and the states. Hence, the attack on key schedule is more difficult to analyze and the faults are more widely propagated. Therefore, our analyzing method is different from that of [35]. For example, if the eighth round state is corrupted (as in [35]), the faulty value propagates into the eighth, ninth, and tenth states. However, if the eighth round key schedule is corrupted (our case), the eighth, ninth, and tenth round keys as well as eighth, ninth, and tenth states are affected.
2) One-byte fault model is used: while the previous DFA on AES Key Schedule [19], [25], [31] assume that three or four bytes are corrupted at once, which is impractical in 8-bit or 16-bit architecture.
3) No exhaustive search is required. Therefore, we do not need to know the plaintexts.
4) A fault induced one round earlier is used. Hence, we can break the countermeasures that protect the last two rounds. This is an important study since redundancy is a costly countermeasure against DFA, thus one should ascertain exactly which rounds need to be protected.
5) New DFA on AES-192 and AES-256 Key Schedule are proposed. Our attacks need less faults and exploit the faults induced one round earlier compared to the previous result in [13]. Our attacks utilize our new DFA on AES-128 Key Schedule and the structure of AES Key Schedule simultaneously while methods in [13] use only DFA on AES-128 Key Schedule of [19].
6) As shown in Tables I and II, the paper deals with all different research directions together.

TABLE I
RESEARCH DIRECTIONS OF DFA ON AES STATE

Ref.	DFA on state			
	Reduce faults	Multi-byte	Extension	Earlier rounds
[14], [22], [27], [35]	✓			
[23], [30]		✓		
[22]		✓	✓	
[4], [18], [33]	✓		✓	
[26]				✓

TABLE II
RESEARCH DIRECTIONS OF DFA ON AES KEY SCHEDULE

Ref.	DFA on key schedule			
	Reduce faults	Multi-byte	Extension	Earlier rounds
[9], [15], [19], [25], [34]	✓			
[13]			✓	
This article	✓	✓	✓	✓

The rest of this paper is organized as follows. We briefly describe AES in Section II. We explain our DFA on AES-128 in Section III. Sections II–VII describe DFA on AES-192 and AES-256. Finally, Section VI compares our attacks with existing ones and Section VII concludes the paper.

II. AES

AES [2] can encrypt and decrypt 128-bit blocks with 128-, 192-, or 256-bit keys. The intermediate computation result of AES, called *state*, is usually represented by a 4×4 matrix, where each cell represents a byte. We denote the input of the ith round by S^i, where $i \in \{1, \ldots, r\}$ and r is the number of rounds. S^1 is the plaintext and S^r is the input to the final round. As shown in Fig. 1, S^i_j denotes the $(j+1)$th byte of the ith state, where $j \in \{0, \ldots, 15\}$. AES-128, AES-192, and AES-256 have 10, 12, and 14 rounds, respectively. Each round function is composed of four transformations except the last round: *SubBytes*, *ShiftRows*, *MixColumns*, and *AddRoundKey*. The last round is lacking *MixColumns*.

1) *SubBytes:* It is made up of the application of 16 identical 8×8 Sboxes. This is a nonlinear byte substitution. We denote the function of *SubBytes* by $\mathbf{SB}$. That is, $\mathbf{SB}(S^i) = SubBytes(S^i)$. We denote *Inverse SubBytes* by $\mathbf{SB^{-1}}$.
2) *ShiftRows:* Each row of the *state* is cyclically shifted over different offsets. Row 0 is not shifted, row 1 is shifted by 1 byte, row 2 is shifted by 2 bytes, and row 3 by 3 bytes. We denote *ShiftRows* and its inverse, *InverseShiftRows*, by $\mathbf{SR}$ and $\mathbf{SR^{-1}}$, respectively.
3) *MixColumns:* This is a linear transformation to each column of the *state*. Each column is considered as polynomial over $\mathbb{F}_{2^8}$ and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x) = 03 * x^3 + 01 * x^2 + 01 * x + 02$. We denote the function of *MixColumns* by $\mathbf{MC}$ and its inverse by $\mathbf{MC^{-1}}$.
4) *AddRoundKey:* It is a bitwise XOR with a round key.

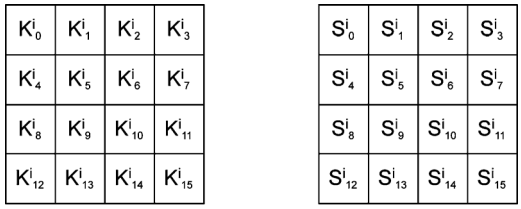


Fig. 1. Each AES State and round key is represented by a 4×4 matrix, where each cell represents a byte.

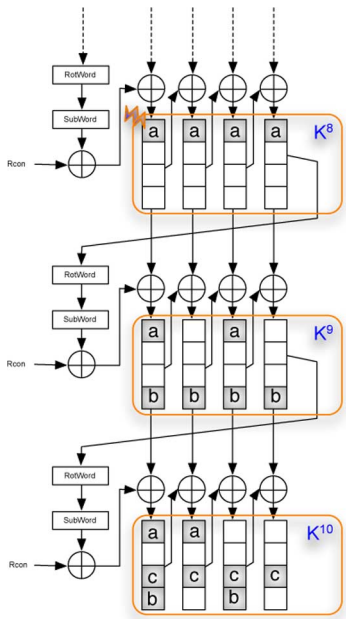


Fig. 2. Propagation of an error in the key schedule of AES-128.

III. DFA on AES-128 Key Schedule

A. Fault Model

We assume that a random byte fault is induced at the first column of the eighth round of AES Key Schedule and the corrupted value is random and unknown to the attacker. She may know which byte of the column is corrupted by precise control of fault injection like in [14]. Although she does not know it, she can conduct four independent and equivalent analyses.

B. Attack

Without loss of generality, we assume that the first byte of the eighth round key is corrupted as shown in Fig. 2. We denote by a the random faulty value. Then all bytes of the first row of the eighth round key, K_0^8, K_1^8, K_2^8, and K_3^8, are impacted by the same faulty value a. The Rotword and nonlinear *SubBytes* transformation are applied to the last column of the eighth round key to compute the ninth round key. This transforms the faulty value a to a new value b that is unpredictable. The b can be expressed from a as follows:

$$b = \mathbf{SB}(K_3^8) \oplus \mathbf{SB}(K_3^8 \oplus a). \tag{1}$$

The b is again transformed to a new value c to compute tenth round key. The c can be expressed as follows:

$$c = \mathbf{SB}(K_{15}^9) \oplus \mathbf{SB}(K_{15}^9 \oplus b). \tag{2}$$

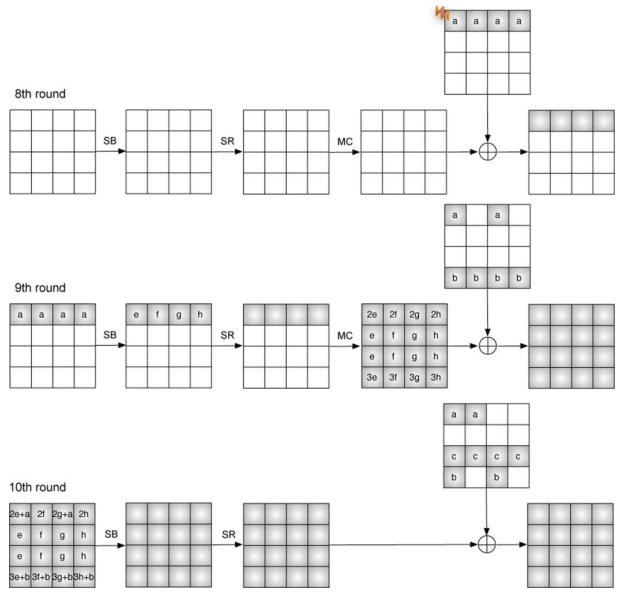


Fig. 3. Propagation of an error in the last three rounds of AES-128.

The corrupted round keys affect the AES States as shown in Fig. 3 where four bytes of S^9 are corrupted by the faulty value a. The *SubBytes* of the ninth round transforms the faulty value a to new values e, f, g, and h. At the end of round 9, all bytes of the state are affected by faulty values. We focus on each column of the input to the final round, S^{10}. We can construct differential equations at the first column as follows:

$$\begin{cases} \Delta S_0^{10} \oplus a = 2 \cdot \Delta S_4^{10} = 2 \cdot \Delta S_8^{10} \\ \Delta S_{12}^{10} \oplus b = 3 \cdot \Delta S_4^{10} = 3 \cdot \Delta S_8^{10}. \end{cases} \tag{3}$$

We assume that the attacker can obtain two pairs of correct and faulty ciphertexts, $(\mathbf{C^1}, \mathbf{C^{1*}})$ and $(\mathbf{C^2}, \mathbf{C^{2*}})$. We denote by a_i the random faulty value induced at the eighth round key and by C^{i*} the corresponding faulty ciphertext, where $i = 1, 2$. Furthermore, we denote by b_i, c_i, e_i, f_i, g_i, and h_i faulty values occurred during computation of $\mathbf{C^{i*}}$.

In (3), we have

$$\begin{aligned} \Delta S_0^{10} &= \mathbf{SB}^{-1}\left(\mathbf{C_0^i} \oplus K_0^{10}\right) \oplus \mathbf{SB}^{-1}\left(\mathbf{C_0^{i}}^{*} \oplus K_0^{10}\right) \\ \Delta S_4^{10} &= \mathbf{SB}^{-1}\left(\mathbf{C_7^i} \oplus K_7^{10}\right) \oplus \mathbf{SB}^{-1}\left(\mathbf{C_7^{i}}^{*} \oplus K_7^{10}\right) \\ \Delta S_8^{10} &= \mathbf{SB}^{-1}\left(\mathbf{C_{10}^i} \oplus K_{10}^{10}\right) \oplus \mathbf{SB}^{-1}(\mathbf{C_{10}^{i}}^{*} \oplus K_{10}^{10}) \\ \Delta S_{12}^{10} &= \mathbf{SB}^{-1}(\mathbf{C_{13}^i} \oplus K_{13}^{10}) \oplus \mathbf{SB}^{-1}(\mathbf{C_{13}^{i}}^{*} \oplus K_{13}^{10}). \end{aligned}$$

Hence, the complexity to solve (3) is 2^{56} as there are seven unknown variables, K_0^{10}, K_7^{10}, K_{10}^{10}, K_{13}^{10}, a, b, and c, which is impractical. However, we can reduce the complexity to at most 2^{24} by dealing with two bytes each time. We explain out attack step by step as follows:

1) Step 1. Find K_3^{10}, K_6^{10}, h_1 and h_2: We start with S_3^{10} and S_7^{10} as these two bytes are affected by only one faulty value h. We construct the differential equations with $(\mathbf{C^1}, \mathbf{C^{1*}})$ as follows:

$$\begin{aligned} &\mathbf{SB}^{-1}\left(\mathbf{C_3^1} \oplus K_3^{10}\right) \oplus \mathbf{SB}^{-1}\left(\mathbf{C_3^{1*}} \oplus K_3^{10}\right) = \Delta S_3^{10} \\ &\quad = 2 \cdot h_1, \\ &\mathbf{SB}^{-1}\left(\mathbf{C_6^1} \oplus K_6^{10}\right) \oplus \mathbf{SB}^{-1}\left(\mathbf{C_6^{1*}} \oplus K_6^{10}\right) \\ &\quad = \Delta S_7^{10} = h_1. \end{aligned}$$

Hence, we obtain:

$$\mathbf{SB^{-1}}\left(\mathbf{C_3^1}\oplus K_3^{10}\right)\oplus\mathbf{SB^{-1}}(\mathbf{C_3^{1*}}\oplus K_3^{10}) = 2\cdot\left\{\mathbf{SB^{-1}}\left(\mathbf{C_6^1}\oplus K_6^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_6^{1*}}\oplus K_6^{10}\right)\right\}$$

where only K_3^{10} and K_6^{10} are unknown. Among 2^{16} candidates for (K_3^{10}, K_6^{10}), about $2^{16}\cdot 255/255^2$ candidates satisfy the equation. Because the probability that a candidate satisfies the equation is $255/255^2$. We construct another equation with $(\mathbf{C^2}, \mathbf{C^{2*}})$ and reduce the number of candidates to one.[1] Hence, two pairs are sufficient to find K_3^{10} and K_6^{10} from that we can also compute h_1 and h_2.

2) Step 2. Find K_9^{10}, c_1 and c_2: We construct the differential equations at S_{11}^{10} as follows:

$$h_i=\mathbf{SB^{-1}}\left(\mathbf{C_9^i}\oplus K_9^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_9^{i\,*}}\oplus K_9^{10}\oplus c_i\right).$$

As we know h_1 and h_2 from the previous analysis, only K_9^{10}, c_1, and c_2 are unknown. Hence, among 2^{24} candidates for them, $2^{24}\cdot(1/255)^2\simeq 256$ candidates satisfy the above equations simultaneously.

3) Step 3. Find K_5^{10}, K_8^{10}, g_1 and g_2: We construct the differential equations at S_6^{10} and S_{10}^{10} as follows:

$$\begin{aligned}g_i&=\mathbf{SB^{-1}}\left(\mathbf{C_5^i}\oplus K_5^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_5^{i\,*}}\oplus K_5^{10}\right)\\&=\mathbf{SB^{-1}}\left(\mathbf{C_8^i}\oplus K_8^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_8^{i\,*}}\oplus K_8^{10}\oplus c_i\right)\end{aligned}$$

where K_5^{10}, K_8^{10}, c_1, and c_2 are unknown. However, we have already found 256 candidates for (c_1, c_2) in the previous analysis. Hence, the space for search is $2^8\cdot 2^{16}=2^{24}$ and the number of candidates satisfying the above equation is $2^{24}\cdot(255/255^2)^2\simeq 2^8$.

4) Step 4. Find K_4^{10}, K_{11}^{10}, f_1, f_2, K_7^{10}, K_{10}^{10}, e_1, and e_2: Similar to the previous analysis, we construct the differential equations at S_5^{10} and S_9^{10} as follows:

$$\begin{aligned}f_i&=\mathbf{SB^{-1}}\left(\mathbf{C_4^i}\oplus K_4^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_4^{i\,*}}\oplus K_4^{10}\right)\\&=\mathbf{SB^{-1}}\left(\mathbf{C_{11}^i}\oplus K_{11}^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_{11}^{i\,*}}\oplus K_{11}^{10}\oplus c_i\right).\end{aligned}$$

The number of candidates for K_4^{10}, K_{11}^{10}, c_1, and c_2 satisfying the above equation is about 2^8. We perform similar analysis to find K_7^{10} and K_{10}^{10} by constructing and solving equations at S_4^{10} and S_8^{10}. Hence, up to now we have found h_1, h_2, K_3^{10}, K_6^{10} and have about 256 candidates for c_1, c_2, e_1, e_2, f_1, f_2, g_1, g_2, and nine key bytes.

5) Step 5. Find K_{12}^{10}, K_{13}^{10}, K_{14}^{10}, K_{15}^{10}, b_1, and b_2: We consider the fourth row of S^{10}. At the first and the third bytes, we construct differential equations as follows:

$$\begin{aligned}b_i&=\mathbf{SB^{-1}}\left(\mathbf{C_{13}^i}\oplus K_{13}^{10}\right)\\&\quad\oplus\mathbf{SB^{-1}}\left(\mathbf{C_{13}^{i\,*}}\oplus K_{13}^{10}\right)\oplus(3\cdot e_i)\\&=\mathbf{SB^{-1}}\left(\mathbf{C_{15}^i}\oplus K_{15}^{10}\right)\\&\quad\oplus\mathbf{SB^{-1}}\left(\mathbf{C_{15}^{i\,*}}\oplus K_{15}^{10}\right)\oplus(3\cdot g_i).\end{aligned}$$

[1]The number of candidates satisfying equations is $2^{16}\cdot(255/255^2)^2=1.0079$. Hence, sometimes two candidates remain. However, we can remove the wrong candidate in the subsequent analysis.

Similarly, we construct differential equations at the second and the fourth bytes as follows:

$$\begin{aligned}b_i&=\mathbf{SB^{-1}}\left(\mathbf{C_{14}^i}\oplus K_{14}^{10}\right)\\&\quad\oplus\mathbf{SB^{-1}}\left(\mathbf{C_{14}^{i}}^{*}\oplus K_{14}^{10}\oplus b_i\right)\oplus(3\cdot f_i)\\&=\mathbf{SB^{-1}}\left(\mathbf{C_{12}^i}\oplus K_{12}^{10}\right)\\&\quad\oplus\mathbf{SB^{-1}}\left(\mathbf{C_{12}^{i}}^{*}K_{12}^{10}\oplus b_i\right)\oplus(3\cdot h_i).\end{aligned}$$

We can think that the last row of S^{10} is affected by only b as we know e, f, g, h. Hence, we can remove about $(255/255^4)^2=2^{-48}$ wrong candidates with two pairs of correct and faulty ciphertexts. The total space for search is $2^{32}\cdot 2^8=2^{40}$ in the equations as four key bytes are unknown and the number of candidates for $e_1, e_2, f_1, f_2, g_1, g_2$ is 2^8. Therefore, we can find one candidate satisfying these equations. In practice, we first find candidates satisfying one equation, then extend search with the other equation to make the time complexity to be 2^{24} instead of 2^{40}.

6) Step 6. Find K_0^{10}, K_1^{10}, K_2^{10}, a_1, and a_2: We construct the differential equations at S_2^{10} as follows:

$$a_i=\mathbf{SB^{-1}}\left(\mathbf{C_2^i}\oplus K_2^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_2^{i}}^{*}\oplus K_2^{10}\right)\oplus(2\cdot g_i).$$

As we know g_1 and g_2, we can find 2^8 candidates for (a_1, a_2, K_2^{10}) satisfying the above equations. Then we construct the differential equations at S_1^{10} as follows:

$$2\cdot f_i=\mathbf{SB^{-1}}\left(\mathbf{C_1^i}\oplus K_1^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_1^{i}}^{*}\oplus K_1^{10}\oplus a_i\right).$$

As we know f_1 and f_2, we can find K_1^{10} and hence a_1, a_2, K_2^{10}. Finally, we construct the differential equations at S_0^{10} as follows and find K_0^{10}:

$$2\cdot e_i\oplus a_i=\mathbf{SB^{-1}}\left(\mathbf{C_0^i}\oplus K_0^{10}\right)\oplus\mathbf{SB^{-1}}\left(\mathbf{C_0^{i}}^{*}\oplus K_0^{10}\oplus a_i\right).$$

As a result, we can find the AES-128 key by AES Key Schedule from the tenth round key we have found.

IV. DFA on AES-256 Key Schedule

As the AES Key Schedule algorithm differs for each variant, we need both K^{14} and K^{13} to find the AES-256 key. We first find the last round key K^{14} by DFA and then reduce the AES algorithm by reversing a cipher back up to the output of the 13th round with the knowledge of the last round key. Hence, DFA can be applied on a shorter AES algorithm to find K^{13}, the penultimate round becoming the last one. As mentioned in [13], we have two problems in this approach. First, the fault diffusion does not follow the same path of AES-128 due to the differences in Key Schedule algorithm. Therefore, we have to find a new path. The second problem comes from operating the inverse transformation on the cipher until the end of the 13th round. The effect of diffused faults on the last round key has to be canceled during the inverse transformation, that is, the faulty round key K^{14^*} should be obtained. By modifying the method used in Section III, we show how to overcome these problems. We use two pairs of correct and faulty ciphertexts to find the last round key and two additional pairs to find the penultimate round key. This is a big improvement compared to the method in [13] that requires 16 pairs.

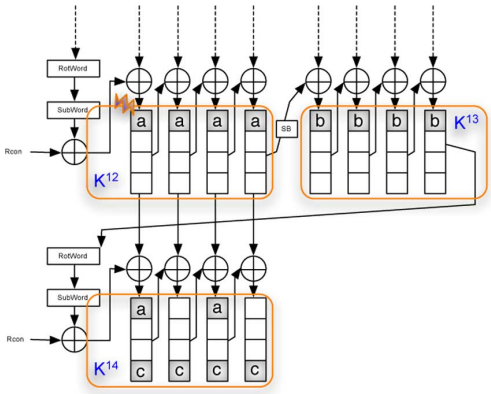


Fig. 4. Propagation of an error induced at K^{12} in the key schedule of AES-256.

A. Fault Model

We assume that the attacker obtains two faulty ciphertexts by inducing a random byte fault at the first column of the 12th round of AES Key Schedule and two faulty ciphertexts by inducing a fault one round earlier as shown in Figs. 4 and 6.

B. Attack

We start with finding the last round key. Without loss of generality we assume that the first byte of the 12th round key is corrupted as shown in Fig. 4. We denote by a the random faulty value. The Rotword and nonlinear *SubBytes* transform the faulty value a to a new value b that is unpredictable. The b can be expressed from a as follows:

$$b = \mathbf{SB}\left(K_3^{12}\right) \oplus \mathbf{SB}\left(K_3^{12} \oplus a\right). \tag{4}$$

The b is again transformed to a new value c to compute the 14th round key. The c can be expressed as follows:

$$c = \mathbf{SB}\left(K_3^{13}\right) \oplus \mathbf{SB}\left(K_3^{13} \oplus b\right). \tag{5}$$

The corrupted round keys affect the AES States as shown in Fig. 5. The *SubBytes* of the 13th round transforms the faulty value a to new values e, f, g, and h. We focus on the input to the final round, S^{14}. We assume that the attacker can obtain two pairs of correct and faulty ciphertexts, $(\mathbf{C^1}, \mathbf{C^{1*}})$ and $(\mathbf{C^2}, \mathbf{C^{2*}})$ and explain out attack step by step as follows:

1) Step A1. Find K_6^{14}, K_9^{14}, h_1, and h_2: We start with S_7^{14} and S_{11}^{14} as these two bytes are affected by only one faulty value h. We construct the differential equations with $(\mathbf{C^1}, \mathbf{C^{1*}})$ as follows:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(\mathbf{C_6^1} \oplus K_6^{14}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(\mathbf{C_6^{1*}} \oplus K_6^{14}\right) = \Delta S_7^{14} = h_1, \\
&\mathbf{SB^{-1}}\left(\mathbf{C_9^1} \oplus K_9^{14}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(\mathbf{C_9^{1*}} \oplus K_9^{14}\right) = \Delta S_{11}^{14} = h_1.
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(\mathbf{C_6^1} \oplus K_6^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_6^{1*}} \oplus K_6^{14}\right) \\
&\qquad = \mathbf{SB^{-1}}\left(\mathbf{C_9^1} \oplus K_9^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_9^{1*}} \oplus K_9^{14}\right)
\end{aligned}$$

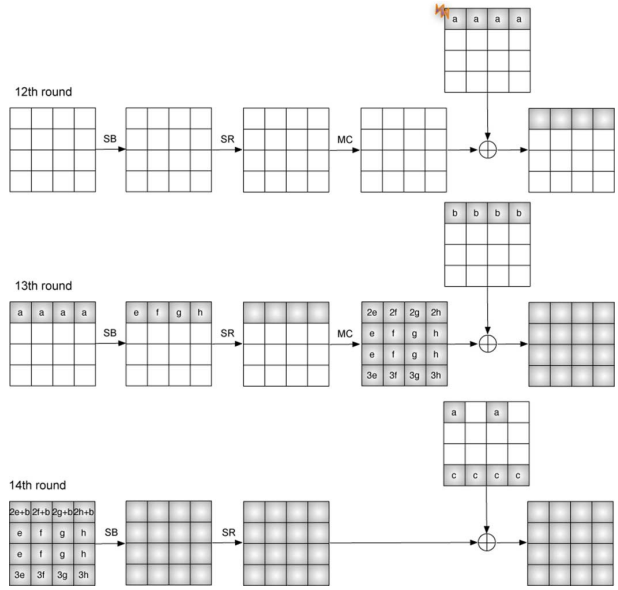


Fig. 5. Propagation of an error induced at K^{12} in the last three rounds of AES-256.

where only K_6^{14} and K_9^{14} are unknown. We construct another equation with $(\mathbf{C}^2, \mathbf{C^{2*}})$ and find K_6^{14} and K_9^{14}. Once we found two key bytes, we can compute h_1 and h_2.

2) Step A2. Find the Second and the Third Rows of K^{14}: Similar to the previous analysis, we construct the differential equations at S_6^{14} and S_{10}^{14} with two pairs of correct and faulty ciphertexts, and find K_5^{14}, K_8^{14}, g_1, and g_2. The similar analysis is also performed for (S_5^{14}, S_9^{14}) and (S_4^{14}, S_8^{14}). Therefore, we can find the second and the third rows of K^{14} and e_1, e_2, f_1, f_2, g_1, g_2, h_1, and h_2.

3) Step A3. Find K_1^{14}, K_3^{14}, b_1, and b_2: We construct differential equations at the second and the fourth bytes of the first row of S^{14} as follows:

$$\begin{aligned}
b_i &= \mathbf{SB^{-1}}\left(\mathbf{C_1^i} \oplus K_1^{14}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(\mathbf{C_1^{i}}^{*} \oplus K_1^{14}\right) \oplus (2 \cdot f_i) \\
&= \mathbf{SB^{-1}}\left(\mathbf{C_3^i} \oplus K_3^{14}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(\mathbf{C_3^{i}}^{*} \oplus K_3^{14}\right) \oplus (2 \cdot h_i).
\end{aligned}$$

As only K_1^{14} and K_3^{14} are unknown, we can find one candidate for them satisfying the above equations.

4) Step A4. Find k_{12}^{14}, c_1, and c_2: We construct the differential equations at S_{15}^{14} as follows:

$$3 \cdot h_i = \mathbf{SB^{-1}}\left(\mathbf{C_{12}^i} \oplus K_{12}^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_{12}^i}^{*} \oplus K_{12}^{14} \oplus c_i\right).$$

As we know h_1 and h_2 from the previous analysis, only K_{12}^{14}, c_1, and c_2 are unknown. Hence, among 2^{24} candidates for them, $2^{24} \cdot (1/255)^2 \simeq 256$ candidates satisfy the above equations simultaneously.

5) Step A5. Find K_{14}^{14}: We construct the differential equations at S_{13}^{14} as follows:

$$3 \cdot f_i = \mathbf{SB^{-1}}\left(\mathbf{C_{14}^i} \oplus K_{14}^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_{14}^i}^{*} \oplus K_{14}^{14} \oplus c_i\right).$$

We know (f_1, f_2) and have 256 candidates for (c_1, c_2) from the previous analysis. As K_{14}^{14} is unknown, we have $2^8 \cdot 2^8 = 2^{16}$

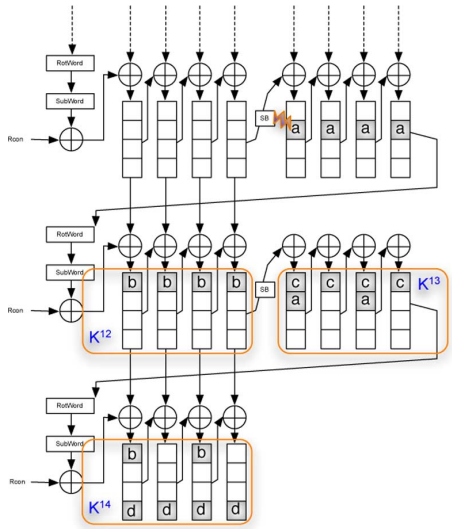


Fig. 6. Propagation of an error induced at K^{11} in the key schedule of AES-256.

candidates for (K_{14}^{14}, c_1, c_2). Hence, we can find one candidate satisfying the above equations simultaneously.

6) Step A6. Find K_{15}^{14} and K_{13}^{14}: Similar to the previous step, we construct the differential equations at S_{14}^{14} and find K_{15}^{14}. The K_{13}^{14} can also be computed from the differential equations constructed at S_{12}^{14}.

7) Step A7. Find K_0^{14}, a_1 and a_2: We construct the differential equations at S_0^{14} as follows:

$$2 \cdot e_i \oplus b_i = \mathbf{SB^{-1}}\left(\mathbf{C_0^i} \oplus K_0^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_0^{i}}^{*} \oplus K_0^{14} \oplus a_i\right).$$

As only K_0^{14}, a_1, and a_2 are unknown, we have about 2^{24} candidates. Hence, $2^{24} \cdot (1/255)^2 \simeq 256$ candidates satisfy the above equations simultaneously.

8) Step A8. Find K_2^{14}: We construct the differential equations at S_2^{14} as follows:

$$2 \cdot g_i \oplus b_i = \mathbf{SB^{-1}}\left(\mathbf{C_2^i} \oplus K_2^{14}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_2^{i}}^{*} \oplus K_2^{14} \oplus a_i\right).$$

As we have 256 candidates for (a_1, a_2) from the previous analysis and K_2^{14} is unknown, we can find one candidate satisfying the above equations simultaneously.

We found K^{14} with two pairs of correct and faulty ciphertexts. With the knowledge of the last round key we can reduce the AES algorithm by neutralizing its last round. In order for the penultimate round to be equivalent to the last round, the *MixColumns* transformation should be missing. Due to the linear property of *MixColumns* and *AddRoundKey*, we can switch them as follows:

$$\mathbf{MC}(S) \oplus K^{13} = \mathbf{MC}(S \oplus \mathbf{MC}^{-1}(K^{13})).$$

We denote by IK^{13} the inverse *MixColumns* transformation applied to K^{13}, that is, $IK^{13} = \mathbf{MC^{-1}}(K^{13})$. We can construct differential equations at the first row of S^{13} where all bytes are affected by the faulty value a. Hence, we can find the first row of IK^{13}. By inducing a fault at a different byte, we can find a different row of IK^{13}. Therefore, we can find the last round and the penultimate round keys with eight pairs of correct and faulty ciphertexts. However, we show how to find the penultimate round key with just two pairs.

We assume that the second byte[2] of the first column of the 11th round key is corrupted, as shown in Fig. 6. The Rotword and nonlinear *SubBytes* transform the faulty value a to a new value b that is unpredictable. The b can be expressed from a as follows:

$$b = \mathbf{SB}\left(K_7^{11}\right) \oplus \mathbf{SB}\left(K_7^{11} \oplus a\right). \tag{6}$$

The b is again transformed to a new value c to compute 13th round key. The c can be expressed as follows:

$$c = \mathbf{SB}\left(K_3^{12}\right) \oplus \mathbf{SB}\left(K_3^{12} \oplus b\right). \tag{7}$$

The c is transformed to a new value d to compute the last round key. The d can be expressed as follows:

$$d = \mathbf{SB}(K_3^{13}) \oplus \mathbf{SB}(K_3^{13} \oplus c). \tag{8}$$

The corrupted round keys affect the AES States as shown in Fig. 7. We focus on the input to the penultimate round S^{13}, where each column is affected by the same error. We assume that the attacker can obtain two pairs of correct and faulty ciphertexts, $(\mathbf{C^3}, \mathbf{C^{3*}})$ and $(\mathbf{C^4}, \mathbf{C^{4*}})$. We denote by a_i the random faulty value induced at the 11th round key and by $\mathbf{C^{i*}}$ the corresponding faulty ciphertext, where $i = 3, 4$. Furthermore, we denote by b_i, c_i, d_i, e_i, f_i, g_i, and h_i faulty values occurred during computation of $\mathbf{C^{i*}}$.

9) Step B1. Find IK_9^{13}, b_3, c_3, d_3, b_4, c_4, and d_4: We first guess b_i, $i = 3, 4$. Then c_i is computed from (7), where K_3^{12} is computed from the 14th round key as follows:

$$K_3^{12} = K_3^{14} \oplus K_2^{14}. \tag{9}$$

The d_i's are computed from (8), where K_3^{13} is computed from (5).[3] As b_i's and d_i's are known, we can get the output of 13th round S^{14} by decrypting the last round with K^{14} and K^{14^*}. We denote by T (and T^*) the inverse *MixColumns* transformation applied to S^{14} (and S^{14^*}), that is, $T = \mathbf{MC}^{-1}(S^{14})$ and $T^* = \mathbf{MC}^{-1}(S^{14^*})$.

We construct the differential equations at S_3^{13} and S_{11}^{13} with $(\mathbf{C^3}, \mathbf{C^{3*}})$ as follows:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_3 \oplus IK_3^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(T_3{}^* \oplus IK_3^{13} \oplus 14 \cdot c_3\right) \\
&\quad = \Delta S_3^{13} = 3 \cdot e_3 + b_3 \\
&\mathbf{SB^{-1}}\left(T_9 \oplus IK_9^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left(T_9{}^* \oplus IK_9^{13} \oplus 13 \cdot c_3\right) \\
&\quad = \Delta S_{11}^{13} = e_3.
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
&\mathbf{SB^{-1}}(T_3 \oplus IK_3^{13}) \oplus \mathbf{SB^{-1}}(T_3{}^* \oplus IK_3^{13} \oplus 14 \cdot c_3) \\
&\quad \oplus b_3 = 3 \cdot (\mathbf{SB^{-1}}(T_9 \oplus IK_9^{13}) \\
&\quad \oplus \mathbf{SB^{-1}}(T_9{}^* \oplus IK_9^{13} \oplus 13 \cdot c_3))
\end{aligned}$$

[2]We assume that the locations of the faults induced at K^{12} and K^{11} are consecutive. In that case, the time complexity to find the penultimate round key is 2^{24}. If it is not, the time complexity becomes 2^{32}, which is still practical to implement.

[3]If the locations of the faults induced at K^{12} and K^{11} are not consecutive, we cannot compute K_3^{13}. In that case, we guess the value of it.

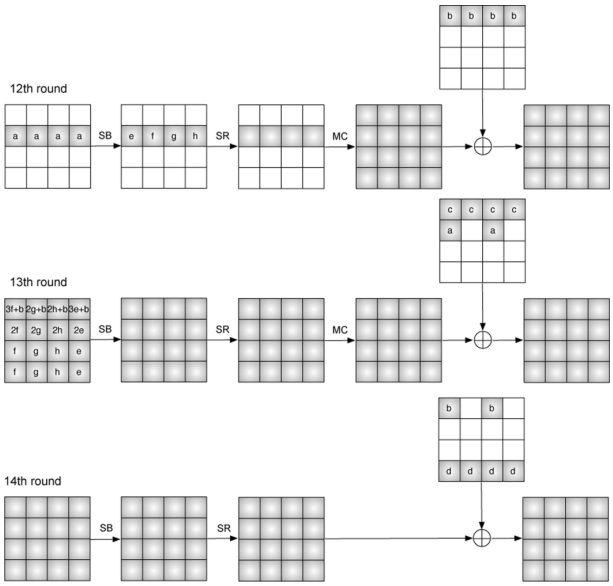


Fig. 7. Propagation of an error induced at K^{11} in the last three rounds of AES-256.

where only IK_9^{13} and b_3 are unknown. We construct another equation with $(\mathbf{C^4}, \mathbf{C^{4*}})$, where IK_9^{13} and b_4 are unknown. Therefore, from three unknown values and two equations, we can find $2^{24} \cdot (255/255^2)^2 \simeq 2^8$ candidates for (IK_9^{13}, b_3, b_4) satisfying the equations.

10) Step B2. Find IK_{11}^{13}, b_3, c_3, d_3, b_4, c_4, and d_4: We construct the differential equations at S_1^{13} and S_9^{13} as follows:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_1 \oplus IK_1^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left({T_1}^* \oplus IK_1^{13} \oplus 14 \cdot c_i\right) \\
&\quad = \Delta S_1^{13} = 3 \cdot g_i + b_i \\
&\mathbf{SB^{-1}}\left(T_{11} \oplus IK_{11}^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left({T_{11}}^* \oplus IK_{11}^{13} \oplus 13 \cdot c_i\right) \\
&\quad = \Delta S_9^{13} = g_i
\end{aligned}$$

where $i = 3, 4$. As IK_{11}^{13} is unknown and there are 2^8 candidates for (b_3, b_4), we can find $2^8 \cdot 2^8 \cdot (255/255^2)^2 \simeq 1$ candidate satisfying the equations.

11) Step B3. Find $(IK_5^{13}, IK_{15}^{13})$ and $(IK_7^{13}, IK_{13}^{13})$: We construct the differential equations at S_5^{13} and S_{13}^{13} and hence find IK_5^{13} and IK_{15}^{13}:

$$\begin{aligned}
&\mathbf{SB^{-1}}(T_5 \oplus IK_5^{13}) \oplus \mathbf{SB^{-1}}({T_5}^* \oplus IK_5^{13} \oplus 9 \cdot c_i) \\
&\quad = \Delta S_5^{13} = 2 \cdot g_i \\
&\mathbf{SB^{-1}}(T_{15} \oplus IK_{15}^{13}) \oplus \mathbf{SB^{-1}}({T_{15}}^* \oplus IK_{15}^{13} \oplus 11 \cdot c_i) \\
&\quad = \Delta S_{13}^{13} = g_i.
\end{aligned}$$

Similarly we find $(IK_7^{13}, IK_{13}^{13})$ from the equations at S_7^{13} and S_{15}^{13}:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_1 \oplus IK_7^{13}\right) \oplus \mathbf{SB^{-1}}\left({T_7}^* \oplus IK_7^{13} \oplus 9 \cdot c_i\right) \\
&\quad = \Delta S_7^{13} = 2 \cdot e_i \\
&\mathbf{SB^{-1}}\left(T_{11} \oplus IK_{13}^{13}\right) \oplus \mathbf{SB^{-1}}\left({T_{13}}^* \oplus IK_{13}^{13} \oplus 11 \cdot c_i\right) \\
&\quad = \Delta S_{15}^{13} = e_i.
\end{aligned}$$

12) Step B4. Find a_3 and a_4: We find a_3 and a_4 by constructing differential equations at S_0^{13} and S_4^{13}

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_0 \oplus IK_0^{13}\right) \oplus \\
&\quad \mathbf{SB^{-1}}\left({T_0}^* \oplus IK_0^{13} \oplus 14 \cdot c_i \oplus 11 \cdot a_i\right) \\
&= \Delta S_0^{13} = 3 \cdot f_i \oplus b_i, \mathbf{SB^{-1}}(T_7 \oplus IK_7^{13}) \\
&\quad \oplus \mathbf{SB^{-1}}({T_7}^* \oplus IK_7^{13} \oplus 9 \cdot c_i) \\
&= \Delta S_4^{13} = 2 \cdot f_i,
\end{aligned}$$

where only a_i's are unknown.

13) Step B5. Find IK_{10}^{13} and IK_8^{13}: We find IK_{10}^{13} by constructing equations at S_0^{13} and S_8^{13}

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_0 \oplus IK_0^{13}\right) \oplus \\
&\quad \mathbf{SB^{-1}}\left({T_0}^* \oplus IK_0^{13} \oplus 14 \cdot c_i \oplus 11 \cdot a_i\right) \\
&= \Delta S_0^{13} = 3 \cdot f_i \oplus b_i, \mathbf{SB^{-1}}\left(T_{10} \oplus IK_{10}^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left({T_{10}}^* \oplus IK_{10}^{13} \oplus 13 \cdot c_i \oplus 9 \cdot c_i\right) \\
&= \Delta S_8^{13} = f_i.
\end{aligned}$$

Similarly, we find IK_8^{13} by constructing equations at S_2^{13} and S_{10}^{13}

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_2 \oplus IK_2^{13}\right) \oplus \\
&\quad \mathbf{SB^{-1}}\left({T_2}^* \oplus IK_2^{13} \oplus 14 \cdot c_i \oplus 11 \cdot a_i\right) \\
&= \Delta S_2^{13} = 3 \cdot h_i \oplus b_i, \mathbf{SB^{-1}}\left(T_8 \oplus IK_8^{13}\right) \\
&\quad \oplus \mathbf{SB^{-1}}\left({T_8}^* \oplus IK_8^{13} \oplus 13 \cdot c_i \oplus 9 \cdot c_i\right) \\
&= \Delta S_{10}^{13} = h_i.
\end{aligned}$$

14) Step B6. Find $(IK_4^{13}, IK_{14}^{13})$ and $(IK_6^{13}, IK_{12}^{13})$: Finally, we find the remaining bytes by constructing and solving the differential equations at (S_5^{13}, S_{13}^{13}) and (S_7^{13}, S_{15}^{13}) as follows:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(T_4 \oplus IK_4^{13}\right) \oplus \\
&\quad \mathbf{SB^{-1}}\left({T_4}^* \oplus IK_4^{13} \oplus 9 \cdot c_i \oplus 14 \cdot a_i\right) \\
&= \Delta S_5^{13} = 2 \cdot g_i \\
&\mathbf{SB^{-1}}(T_{14} \oplus IK_{14}^{13}) \oplus \\
&\quad \mathbf{SB^{-1}}\left({T_{14}}^* \oplus IK_{14}^{13} \oplus 11 \cdot c_i \oplus 13 \cdot c_i\right) \\
&= \Delta S_{13}^{13} = g_i \\
&\mathbf{SB^{-1}}(T_6 \oplus IK_6^{13}) \oplus \\
&\quad \mathbf{SB^{-1}}({T_6}^* \oplus IK_6^{13} \oplus 9 \cdot c_i \oplus 14 \cdot a_i) \\
&= \Delta S_7^{13} = 2 \cdot e_i \\
&\mathbf{SB^{-1}}(T_{12} \oplus IK_{12}^{13}) \oplus \\
&\quad \mathbf{SB^{-1}}({T_{12}}^* \oplus IK_{12}^{13} \oplus 11 \cdot c_i \oplus 13 \cdot c_i) \\
&= \Delta S_{15}^{13} = e_i.
\end{aligned}$$

As a result, we found all bytes of IK^{13}. Hence, the AES-256 key can be computed from K^{14} and $K^{13} = \mathbf{MC}(IK^{13})$.

V. DFA on AES-192 Key Schedule

The knowledge of K^{12} and the right half of K^{11} enables finding the AES-192 key. We first find the last round key K^{12} by DFA and then reduce the AES algorithm by reversing a cipher back up to the output of the 11th round with the knowledge of the last round key. Hence, DFA can be applied on a shorter AES algorithm to find K^{11}, the penultimate round becoming the last one.

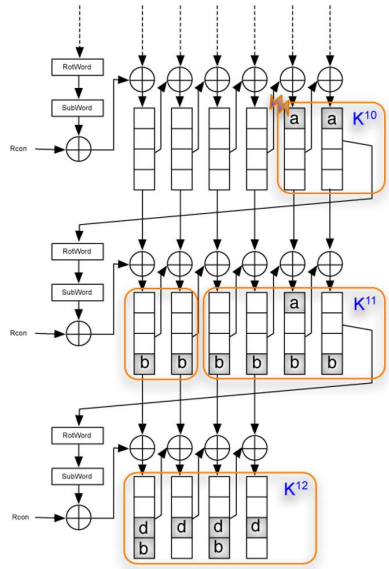


Fig. 8. Propagation of an error induced at K^{10} in the key schedule of AES-192.

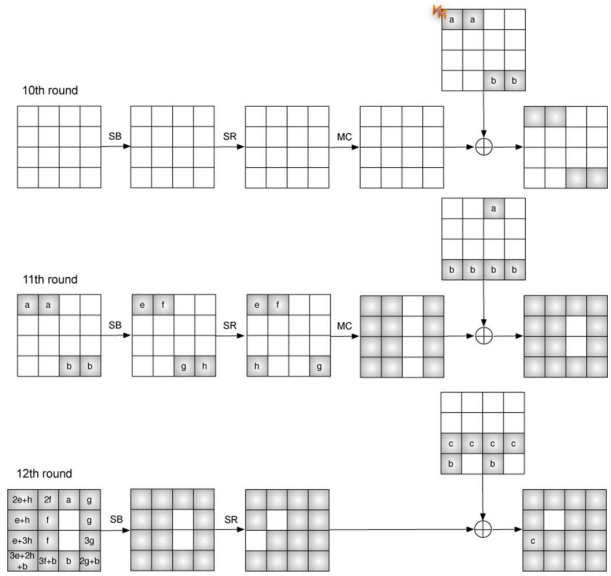


Fig. 9. Propagation of an error induced at K^{10} in the last three rounds of AES-192.

A. Fault Model

We assume that the attacker obtains two faulty ciphertexts by inducing a random byte fault at the first column of the tenth round of AES Key Schedule, as shown in Fig. 8. We also assume that she can obtain two faulty ciphertexts by inducing a random byte fault one round earlier.

B. Attack

We start with finding the last round key and assume that the first byte of the tenth round key is corrupted, as shown in Fig. 8. We denote by a the random faulty value. The nonlinear *SubBytes* transforms the faulty value a to a new value b that is unpredictable. The b can be expressed from a as follows:

$$b = \mathbf{SB}(K_1^{10}) \oplus \mathbf{SB}(K_1^{10} \oplus a). \tag{10}$$

The b is again transformed to a new value c as follows:

$$c = \mathbf{SB}\left(K_{15}^{11}\right) \oplus \mathbf{SB}\left(K_{15}^{11} \oplus b\right). \tag{11}$$

The corrupted round keys affect the AES States, as shown in Fig. 9. We explain out attack step by step as follows:

1) Step 1. Find c_1 and c_2: We can easily find c_i, $i = 1, 2$ from the correct and faulty ciphertext as follows:

$$c_i = \mathbf{C_8^i} \oplus \mathbf{C_8^i}^*.$$

2) Step 2. Find K_3^{12}, K_6^{12} and K_9^{12}: We start with S_3^{12}, S_7^{12}, and S_{11}^{12} as these bytes are affected by only one faulty value g. We construct the differential equations as follows:

$$\begin{aligned}
&\mathbf{SB^{-1}}\left(\mathbf{C_3^i} \oplus K_3^{12}\right) \oplus \mathbf{SB^{-1}}(\mathbf{C_3^i}^* \oplus K_3^{12}) \\
&\quad = \Delta S_3^{12} = g_i \\
&\mathbf{SB^{-1}}\left(\mathbf{C_6^i} \oplus K_6^{12}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_6^i}^* \oplus K_6^{12}\right) \\
&\quad = \Delta S_7^{12} = g_i \\
&\mathbf{SB^{-1}}\left(\mathbf{C_9^i} \oplus K_9^{12}\right) \oplus \mathbf{SB^{-1}}\left(\mathbf{C_9^i}^* \oplus K_9^{12} \oplus c_i\right) \\
&\quad = \Delta S_{11}^{12} = 3 \cdot g_i.
\end{aligned}$$

With two pairs of correct and faulty ciphertexts, we can find three key bytes and g_i's.

3) Step 3. Find K_1^{12}, K_4^{12}, and K_{11}^{12}: Similar to the previous analysis, we construct the differential equations at S_1^{12}, S_5^{12}, and S_9^{12} and find K_1^{12}, K_4^{12}, and K_{11}^{12}.

4) Step 4. Find K_{12}^{12} and K_{14}^{12}: We construct the differential equations at S_{13}^{12} and S_{15}^{12} as follows:

$$\begin{aligned}
&\mathbf{SB}^{-1}(\mathbf{C_{12}^i} \oplus K_{12}^{12}) \oplus \mathbf{SB^{-1}}(\mathbf{C_{12}^i}^* \oplus K_{12}^{12} \oplus b_i) \\
&= \Delta S_{15}^{12} = 2 \cdot g_i \oplus b_i \qquad (12) \\
&\mathbf{SB}^{-1}(\mathbf{C_{14}^i} \oplus K_{14}^{12}) \oplus \mathbf{SB^{-1}}(\mathbf{C_{14}^i}^* \oplus K_{14}^{12} \oplus b_i) \\
&= \Delta S_{13}^{12} = 3 \cdot f_i \oplus b_i. \qquad (13)
\end{aligned}$$

In (12), b_1, b_2, and K_{12}^{12} are unknown. Hence, we can find 256 candidates for them satisfying (12). We eliminate wrong candidates again with (13) and finally get one quadruplet for $(K_{12}^{12}, K_{14}^{12}, b_1, b_2)$.

5) Step 5. Find K_{15}^{12}: As b_i's are found in the previous step, we can find K_{15}^{12} satisfying the following equation:

$$\mathbf{SB^{-1}}(\mathbf{C_{15}^i} \oplus K_{15}^{12}) \oplus \mathbf{SB^{-1}}(\mathbf{C_{15}^i}^* \oplus K_{15}^{12}) = \Delta S_{14}^{12} = b_i.$$

6) Step 6. Find K_0^{12}, K_7^{12}, K_{10}^{12}, and K_{13}^{12}: At the first column of S^{12}, the following equations hold:

$$\begin{aligned}
\Delta S_0^{12} &= 2 \cdot e_i \oplus h_i \\
\Delta S_4^{12} &= e_i \oplus h_i \\
\Delta S_8^{12} &= e_i \oplus 3 \cdot h_i \\
\Delta S_{12}^{12} &= 3 \cdot e_i \oplus 2 \cdot h_i \oplus b_i.
\end{aligned}$$

As we know b_i's, we can obtain the following equations by eliminating e_i and h_i:

$$\begin{aligned}
\Delta S_0^{12} \oplus \Delta S_8^{12} &= \Delta S_{12}^{12} \oplus b_i \\
2 \cdot \Delta S_0^{12} \oplus 3 \cdot \Delta S_8^{12} &= 7 \cdot \Delta S_4^{12}.
\end{aligned}$$

Therefore, with two pairs of correct and faulty ciphertexts, we can find K_0^{12}, K_7^{12}, K_{10}^{12}, and K_{13}^{12}.

7) Step 7. Find the Three Remaining Bytes of k^{12} and k^{11}: We have found 13 bytes of K^{12} enabling us to find additional five bytes of K^{11} by AES-192 Key Schedule [see Fig. 10(d)]. We can find one more byte K_{15}^{11} from (11) as we know b and

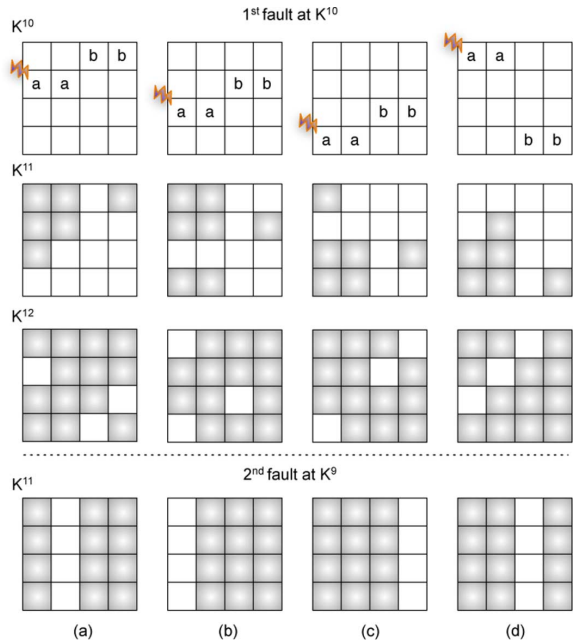


Fig. 10. Key bytes found by DFA according to the location of the fault in AES-192.

c. We obtain additional two faulty ciphertexts by inducing a random byte fault at the first column of the ninth round of AES Key Schedule. Similar to the second part of DFA on AES-256 in Section IV, we eliminate the final round with the knowledge of the last round key. However, we know only three diagonals of K^{12} enabling us to find only three columns of K^{11} regardless of the location of the fault at K^9. As shown in Fig. 10, the number of bytes we can find depends on the location of the fault at K^{10}. Four bytes (three bytes of K^{12} and one byte of K^{11}) are missing when the fault impacts the third byte [see Fig. 10(b)]. In other cases, 5, 6, or 7 bytes are missing, respectively. If we have two more pairs by inducing a random byte fault at the tenth round of AES Key Schedule, we can find all bytes of K^{12}. Therefore, we can conclude that we can find AES-192 key with six pairs of correct and faulty ciphertexts. Furthermore, we can find the key with four pairs enabling an easy exhaustive key search of 2^{32} in one out of four cases.

VI. Simulation Result and Comparison

We implemented our attacks on a PC with a 3.20-GHz Intel with 8GB memory using Python 3.1.2. It takes 42, 21, and 11 min on average to find the AES-128, AES-192, and AES-256 key, respectively.

We summarize our DFA on AES-128, AES-192, and AES-256 in Tables III–V. Compared with the existing ones, our attacks have the following advantages. First our attacks use one-byte fault model, whereas the previous attacks assume that a fault impacts several bytes at once, which is impractical in 8-bit or 16-bit architecture. Second, we reduce the number of pairs required to find the key. Our DFA on AES-128 needs two pairs, which is the best without exhaustive search. The attack in [19] can find AES-128 key with two pairs but requires exhaustive search of 2^{32} candidates, and hence the knowledge of the plaintexts is indispensable. However, our attack can be performed only with ciphertexts. For the attacks on AES-192 and AES-256, our methods require four times less pairs than those in [13]. Finally, our attacks induce faults one round earlier compared to the existing ones. A general countermeasure against DFA is a repetition of the last few rounds to check the errors. Therefore, our attacks show that at least the last three rounds for AES-128 and four rounds for the other variants should be protected.

TABLE III
Comparison With Existing DFA on AES-128 Key Schedule

Reference	Fault Impact	Fault Location	No. of faults
[25]	four bytes	9^{th} round	12
[31]	four bytes	9^{th} round	7
[19]	three bytes	9^{th} round	4
This article	one byte	8^{th} round	2

TABLE IV
Comparison With Existing DFA on AES-192 Key Schedule

Reference	Fault Impact	Fault Location	No. of faults
[13]	one byte	11^{th} and 10^{th} rounds	16
This article	one byte	10^{th} and 9^{th} rounds	4 or 6

TABLE V
Comparison With Existing DFA on AES-256 Key Schedule

Reference	Fault Impact	Fault Location	No. of faults
[13]	one byte	13^{th} and 12^{th} rounds	16
This article	one byte	12^{th} and 11^{th} rounds	4

VII. Conclusion

Due to its complexity, DFA on AES Key Schedule require more pairs of correct and faulty ciphertexts than DFA on AES State. Furthermore, they have used a multibyte fault model, which is impractical in 8-bit or 16-bit architecture. In this paper we proposed new DFA on AES-128, AES-192, and AES-256. Our attacks reduce the number of pairs of correct and faulty ciphertexts based on a one-byte fault model. Furthermore, our attacks induce faults one round earlier than the existing ones. Hence, it is shown that one more round should be protected to prevent DFA on AES.

The general countermeasure against DFA on the AES State is to recompute the last few rounds of AES and compare it with the original output. However, if the key schedule is not redone for recomputation, it cannot prevent DFA on the AES Key Schedule. Therefore, we can conclude that the key scheduling process as well as encryption process should be protected against fault attacks.

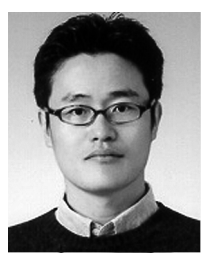
Acknowledgment

The author would like to thank the reviewers for their comments and suggestions. He would also like to thank G. Avoine for helpful comments.

References

[1] *Data Encryption Standard*, NIST FIPS PUB 46-2, National Institute of Standard and Technology, 1993.
[2] *Advanced Encryption Standard*, NIST FIPS PUB 197, National Institute of Standard and Technology, 2001.
[3] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The sorcerer's apprentice guide to fault attacks," in *Proc. Fault Diagnosis and Tolerance in Cryptography in Association With DSN 2004—Int. Conf. Dependable Systems and Networks*, 2004, pp. 330–342.
[4] A. Barenghi, G. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi, "Low voltage fault attacks to AES and RSA on general purpose processors," *IACR*, 2010, eprint archive, 2010-130.

[5] E. Biham, L. Granboulan, and P. Q. Nguyen, "Impossible fault analysis of RC4 and differential fault analysis of RC4," in *Proc. Fast Software Encryption: 12th Int. Workshop (FSE 2005)*, 2005, vol. 3557, Lecture Notes in Computer Science, pp. 359–367, Springer.
[6] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proc. 17th Annu. Int. Cryptology Conf. Advances in Cryptology (CRYPTO '97)*, 1997, vol. 1294, Lecture Notes in Computer Science, pp. 513–525, Springer.
[7] J. Blömer and J.-P. Seifert, "Fault based cryptanalysis of the advanced encryption standard (AES)," in *Proc. 7th Int. Conf. Financial Cryptography (FC 2003)*, 2003, vol. 2742, Lecture Notes in Computer Science, pp. 162–181, Springer.
[8] D. Brumley and D. Boneh, "Remote timing attacks are practical," *Comput. Netw.*, vol. 48, no. 5, pp. 701–716, 2005.
[9] C.-N. Chen and S.-M. Yen, "Differential fault analysis on AES key schedule and some countermeasures," in *Proc. 8th Australasian Conf. Information Security and Privacy (ACISP 2003)*, 2003, vol. 2727, Lecture Notes in Computer Science, pp. 118–129, Springer.
[10] H. Chen, W. Wu, and D. Feng, "Differential fault analysis on CLEFIA," in *Proc. 9th Int. Conf. Information and Communications Security (ICICS 2007)*, 2007, vol. 4861, Lecture Notes in Computer Science, pp. 284–295, Springer.
[11] J.-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, and J.-L. Willems, "A practical implementation of the timing attack," in *Proc. Int. Conf. Smart Card Research and Applications (CARDIS '98)*, 1998, vol. 1820, Lecture Notes in Computer Science, pp. 167–182, Springer.
[12] P. Dusart, G. Letourneux, and O. Vivolo, "Differential fault analysis on AES," in *Proc. First Int. Conf. Applied Cryptography and Network Security (ACNS 2003)*, 2003, vol. 2846, Lecture Notes in Computer Science, pp. 293–306, Springer.
[13] N. Floissac and Y. L'Hyver, "From AES-128 to AES-192 and AES-256, How to adapt differential fault analysis attacks," *IACR*, 2010, eprint archive, 2010-396.
[14] T. Fukunaga and J. Takahashi, "Practical fault attack on a cryptographic LSI with ISO/IEC 18033-3 block ciphers," in *Proc. 6th Int. Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC 2009)*, 2009, pp. 84–92, IEEE Computer Society.
[15] C. Giraud, "DFA on AES," in *Proc. 4th Int. Conf. Advanced Encryption Standard—AES (AES 2004)*, 2005, vol. 3373, Lecture Notes in Computer Science, pp. 27–41, Springer.
[16] L. Hemme, "A differential fault attack against early rounds of (Triple-) DES," in *Proc. 6th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES 2004)*, 2004, vol. 3156, Lecture Notes in Computer Science, pp. 254–267, Springer.
[17] J. J. Hoch and A. Shamir, "Fault analysis of stream ciphers," in *Proc. 6th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES 2004)*, 2004, vol. 3156, Lecture Notes in Computer Science, pp. 240–253, Springer.
[18] C. H. Kim, "Differential fault analysis against AES-192 and AES-256 with minimal faults," in *Proc. 7th Int. Workshop Fault Diagnosis and Tolerance in Cryptography (FDTC 2010)*, 2010, pp. 3–9, IEEE Computer Society.
[19] C. H. Kim and J.-J. Quisquater, "New differential fault analysis on AES key schedule: Two faults are enough," in *Proc. 8th IFIP WG 8.8/11.2 Int. Conf. (CARDIS 2008)*, 2008, vol. 5189, Lecture Notes in Computer Science, pp. 48–60, Springer.
[20] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. 19th Annu. Int. Cryptology Conf. Advances in Cryptology (CRYPTO '99)*, 1999, vol. 1666, Lecture Notes in Computer Science, pp. 388–397, Springer.
[21] W. Li, D. Gu, and Y. Wang, "Differential fault analysis on the contracting UFN structure, with application to SMS4 and Macguffin," *J. Syst. Software*, vol. 82, no. 2, pp. 346–354, 2009.
[22] W. Li, D. Gu, Y. Wang, J. Li, and Z. Liu, "An extension of differential fault analysis on AES," in *Proc. Int. Conf. Network and System Security*, Los Alamitos, CA, 2009, pp. 443–446, IEEE Computer Society.
[23] A. Moradi, M. T. M. Shalmani, and M. Salmasizadeh, "A generalized method of differential fault attack against AES cryptosystem," in *Proc. 8th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES 2006)*, 2006, vol. 4249, Lecture Notes in Computer Science, pp. 91–100, Springer-Verlag.
[24] D. Mukhopadhyay, "An improved fault based attack of the advanced encryption standard," in *Proc. AFRICACRYPT 2009*, 2009, vol. 5580, Lecture Notes in Computer Science, pp. 421–434.
[25] D. Peacham and B. Thomas, A DFA Attack Against the AES Key Schedule 2006 [Online]. Available: http://www.siven-ture.com/pdfs/AES/KeySchedule/DFA/whitepaper.pdf, SiVenture White Paper 001, 26, Oct.
[26] R. C.-W. Phan and S.-M. Yen, "Amplifying side-channel attacks with techniques from block cipher cryptanalysis," in *Proc. 7th IFIP WG 8.8/11.2 Int. Conf. Smart Card Research and Advanced Applications (CARDIS 2006)*, 2006, vol. 3928, Lecture Notes in Computer Science, pp. 135–150, Springer.
[27] G. Piret and J.-J. Quisquater, "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD," in *Proc. 5th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES 2003)*, 2003, vol. 2779, Lecture Notes in Computer Science, pp. 77–88, Springer.
[28] J.-J. Quisquater and D. Samyde, "Electromagnetic analysis (EMA): Measures and counter-measures for smart cards," in *Proc. Int. Conf. Research in Smart Cards Smart Card Programming and Security (Esmart 2001)*, 2001, vol. 2140, Lecture Notes in Computer Science, pp. 200–210, Springer.
[29] M. Rivain, "Differential fault analysis on DES middle rounds," in *Proc. 11th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES 2009)*, 2009, vol. 5747, Lecture Notes in Computer Science, pp. 457–469.
[30] D. Saha, D. Mukhopadhyay, and D. RoyChowdhury, "A diagonal fault attack on the advanced encryption standard," *IACR*, 2009, eprint archive, 2009-581, Springer.
[31] J. Takahashi and T. Fukunaga, "Differential fault analysis on the AES key schedule," *IACR*, eprint archive, 2007-480.
[32] J. Takahashi and T. Fukunaga, "Improved differential fault analysis on CLEFIA," in *Proc. Fifth Int. Workshop on Fault Diagnosis and Tolerance in Cryptography, 2008 (FDTC 2008)*, 2008, pp. 25–34, IEEE Computer Society.
[33] J. Takahashi and T. Fukunaga, "Differential fault analysis on AES with 192 and 256-bit keys," *IACR*, 2010, eprint archive, 2010-023.
[34] J. Takahashi, T. Fukunaga, and K. Yamakoshi, "DFA mechanism on the AES key schedule," in *Proc. 4th Int. Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC 2007)*, 2007, pp. 62–74, IEEE Computer Society.
[35] M. Tunstall and D. Mukhopadhyay, "Differential fault analysis of the advanced encryption standard using a single fault," *IACR*, eprint archive, 2009-575.



Chong Hee Kim received the B.S. degree from Kyungpook National University, Republic of Korea, in 1997, and the M.S. and Ph.D. degrees from Pohang University of Science and Technology (POSTECH), Republic of Korea in 1999 and 2004, respectively.

He worked for Samsung Electronics Co., LTD and NXP Semiconductors N.V. He is currently senior researcher at Université catholique de Louvain, Belgium. His recent research interests include RFID secure protocols and security of the embedded systems such as side channel analysis and fault attacks.