Differential Fault Analysis on A.E.S

Pierre Dusart[1], Gilles Letourneux[2], and Olivier Vivolo[2]

[1] LACO (URM CNRS n°6090), Faculté des Sciences & Techniques,
123, avenue Albert THOMAS, 87060 Limoges, France
dusart@unilim.fr,
http://www.unilim.fr/laco
[2] E.D.S.I.
Atalis 1, 1, rue de Paris, 35510 Cesson-Sévigné, France
development@edsi-smartcards.com.fr

Abstract. DFA is no new attack. It was first used by Biham and Shamir who took unfair advantage of DES Feistel structure to carry it out. This structure is not present in AES. Nevertheless, is DFA able to attack AES another way? This article aims at setting out a means of applying DFA to AES that exploits AES internal structure. We can break an AES128 key with ten faulty messages within a few minutes.

1 Introduction

In September 1996, Boneh, Demillo, and Lipton [5] from Bellcore disclosed information about a new type of cryptanalytic attack which exploits computational errors to find cryptographic keys. Their attack is applicable to public key cryptosystems such as RSA, excluding secret key algorithms. In [4], E. Biham & A.Shamir extended this attack to various secret key cryptosystems such as DES, and called it Differential Fault Analysis (DFA). They applied the differential cryptanalysis to Data Encryption Standard (DES) within the frame of hardware fault model.

Since that time 56-bit key has been too short to be secure and worldwide competition between secret key cryptosystems has been raging. The standard which was to replace DES standard had to fulfill the following requirements: be a symmetric cryptosystem with 128 to 256 key sizes, easy to implement with hardware and resilient to linear and differential cryptanalyses. On Oct. 2, 2000, NIST chose Rijndael as Advanced Encryption Standard (AES).

We further assume that the attacker is in possession of a tamperproof-device, so that he can repeat the experiment with the same plaintext and key without applying external physical effects. As a result, he obtains two ciphertexts derived from the same (unknown) plaintext and key, among which one is correct and the other the result of a computation corrupted by a single error occuring during the computation.

J. Zhou, M. Yung, Y. Han (Eds.): ACNS 2003, LNCS 2846, pp. 293–306, 2003.

The major criticism of DFA was about its putting into practice possibilities until some authors [3] proved it to be possible. They introduced a fault while the program related to AES was running. A sealed tamperproof device when exposed to certain physical phenomena (e.g., ionizing or microwave radiation) is very likely to cause a fault to happen at a bit in any of the registers at an intermediate stage during the cryptographic computation. In practice, more than one bit can be altered. Whenever the attacker applies DFA attack to the DES, making the most of DES Feistel structure, he knows both differential input and output of the targeted SBox.

When applying DFA to DES, using the Feistel structure of DES, the attacker knows the differential input and output of the target SBox. It is necessary that the attacker should know these differentials to discover a round key byte. With AES, the situation is different because only output differential is known to the attacker. There is no finding immediately the error that alters substitution input. On the other hand, the set of values possibly taken on by the error slipped in can be determined. Knowing that is still not enough. Given that the fault introduced can possibly take 127 values, the round key byte concerned can take as many as 256 values. Thus AES is immune to the classical differential analysis attack. We intend to reduce the set of values possibly taken on by the error introduced assuming that it spreads over at least two distinct bytes used in the SubBytes operation performed through the ciphering process. The error introduced in each SBox input, can possibly take 127 values. As they originate in the same error, only their intersection deserves further consideration. This way the number of possibly committed errors is reduced by half (for a generic case, these sets are different). Either round key byte included in the target Sbox can then take 128 possible values. Key K_{N_r} value can be found by repeating an error at the same state byte. In the end, we proved that AES is vulnerable to differential fault analysis. We implemented the attack on a personal computer. Our analysis program extracted the full AES-128 key by analysing less than 50 ciphertexts.

The document is organized as follows. After briefly descripting AES, we will list a number of DFA-based attack models and then show how to quickly find out the set of values the last round key is likely to take. In the appendix, we illustrate our attack with an example.

The authors thank Joan Daemen for his valuable comments on the article. We are grateful to Cédric Hasard for his help.

2 Brief Description of AES

In this article, we give a description of AES slightly different from [1] as we use a matrix on $GF(2^8)$ to describe a state. Nevertheless, we keep using the notations of [1].

The AES is a block cipher with block length to 128 bits, and support key lengths N_k of 128, 192 or 256 bits. The AES is a key-iterated block cipher : it consists in repeating the application of a round transformation to the state. The number of rounds is represented by N_r and depends on the key length ($Nr = 10$ for 128 bits, $Nr = 12$ for 192 bits and $N_r = 14$ for 256 bits). The AES transforms a state, noted $S \in M_4(GF(2^8))$, (i.e. S is a 4x4 matrix with its coefficients in $GF(2^8)$) into another state in $M_4(GF(2^8))$. The key K is used for generating $N_r + 1$ round keys noted $K_i \in M_4(GF(2^8))$ $(i = 0, 1, \ldots, N_r)$. With AES, a round of an encryption is composed of four main operations: AddRoundKey, MixColumns, SubBytes, ShiftRows.

Remark 1. The representation chosen in [1] of $GF(2^8)$ is $GF(2)[X]/ < m >$, where $< m >$ is the ideal generated by the irreducible polynomial $m \in GF(2)[X]$, $m = x^8 + x^4 + x^3 + x + 1$.

Remark 2. We use three notations, equivalent to one another, to represent an element in $GF(2^8)$:

- $x^7 + x^6 + x^4 + x^2$, the polynomial notation
- $\{11010100\}_b$, the binary notation
- 'D4', the hexadecimal notation

2.1 AddRoundKey for i^{th} Round

The AddRoundKey transformation consists in adding up matrices in $M_4(GF(2^8))$ between the state and the round key of the i^{th} round. We represent by $S_{i,A}$ the state after the i^{th} AddRoundKey.

$$\begin{aligned} M_4(GF(2^8)) &\longrightarrow M_4(GF(2^8)) \\ S &\longmapsto S_{i,A} = S + K_i \end{aligned}$$

2.2 SubBytes for i^{th} Round

The SubBytes transformation consists in applying to each element of the matrix S an elementary transformation s. We represent by $S_{i,Su}$ the state after the i^{th} SubBytes.

$$\begin{gathered} M_4(GF(2^8)) \longrightarrow M_4(GF(2^8)) \\ S = \begin{pmatrix} S[0] & S[4] & S[8] & S[12] \\ S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \end{pmatrix} \longmapsto S_{i,Su} = \begin{pmatrix} s(S[0]) & s(S[4]) & s(S[8]) & s(S[12]) \\ s(S[1]) & s(S[5]) & s(S[9]) & s(S[13]) \\ s(S[2]) & s(S[6]) & s(S[10]) & s(S[14]) \\ s(S[3]) & s(S[7]) & s(S[11]) & s(S[15]) \end{pmatrix} \end{gathered}$$

where s is the non linear application defined by

$$\begin{aligned} GF(2^8) &\longrightarrow GF(2^8) \\ x &\longmapsto s(x) = \begin{cases} a * x^{-1} + b, \text{ if } x \neq 0, \\ b, \text{ if } x = 0. \end{cases} \end{aligned}$$

a is a linear invertible application over $GF(2)$, $a \in M_8(GF(2))$, $*$ is the multiplication of matrices over $GF(2)$ and $x^{-1} = \{b_0 b_1 ... b_7\}_b$ is seen as a $GF(2)$-vector equal to the transposition of the vector $(b_0, \cdots, b_7)$. The value of $b =$ '63'$\in GF(2^8)$ and

$$a = \begin{pmatrix} 1\,0\,0\,0\,1\,1\,1\,1 \\ 1\,1\,0\,0\,0\,1\,1\,1 \\ 1\,1\,1\,0\,0\,0\,1\,1 \\ 1\,1\,1\,1\,0\,0\,0\,1 \\ 1\,1\,1\,1\,1\,0\,0\,0 \\ 0\,1\,1\,1\,1\,1\,0\,0 \\ 0\,0\,1\,1\,1\,1\,1\,0 \\ 0\,0\,0\,1\,1\,1\,1\,1 \end{pmatrix}.$$

2.3 MixColumns for i^{th} Round

The MixColumns transformation consists in multiplying the state by a fixed matrix A_0 of $M_4(GF(2^8))$. We represent by $S_{i,M}$ the state after the i^{th} MixColumns.

$$\begin{aligned} M_4(GF(2^8)) &\longrightarrow M_4(GF(2^8)) \\ S &\longmapsto S_{i,M} = A_0.S, \end{aligned}$$

where A_0 is defined by

$$A_0 = \begin{pmatrix} \text{'02'} & \text{'03'} & \text{'01'} & \text{'01'} \\ \text{'01'} & \text{'02'} & \text{'03'} & \text{'01'} \\ \text{'01'} & \text{'01'} & \text{'02'} & \text{'03'} \\ \text{'03'} & \text{'01'} & \text{'01'} & \text{'02'} \end{pmatrix}.$$

2.4 ShiftRows for i^{th} Round

The ShiftRows transformation is a byte transposition that cyclically shifts the rows of the state with different offsets. We represent by $S_{i,Sh}$ the state after the i^{th} ShiftRows.

$$\begin{gathered} M_4(GF(2^8)) \longrightarrow M_4(GF(2^8)) \\ S = \begin{pmatrix} S[0] & S[4] & S[8] & S[12] \\ S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \end{pmatrix} \longmapsto S_{i,Sh} = \begin{pmatrix} S[0] & S[4] & S[8] & S[12] \\ S[5] & S[9] & S[13] & S[1] \\ S[10] & S[14] & S[2] & S[6] \\ S[15] & S[3] & S[7] & S[11] \end{pmatrix}. \end{gathered}$$

3 Attacks on Computation of AES

We describe a list of possible DFA attacks on AES. The attacker is able to introduce a fault into the AES computation process and find out the cryptographic operation output. In all of those attack patterns, one fault means any error possibly several bits long, standing at a byte of the state The goal of these attacks is to find out the key K_{Nr} (and K_{Nr-1} in the case of AES 192 and 256 bits) and hence the key K ([1] for interested readers).

3.1 Models of Attack

All these attacks are based on the basic attack pattern.

Basic attack after the $N_r - 2^{th}$ MixColumns and before the $N_r - 1^{th}$ MixColumns. We introduce a random fault into a definite byte in the state known to the attacker between $N_r - 2^{th}$ MixColumns and the $N_r - 1^{th}$ MixColumns. The fault introduced into the state hits four bytes through last MixColumns. Through SubBytes operation, these four bytes interact with four bytes in ${N_r}^{th}$ round key and result in four differential faults ε'. From each of the four differential faults, we define the set $S_{c,\varepsilon'}$ of values possibly taken on by the initial fault. As the four faults originated in the same initial fault, its real value belongs to the set made up by the intersection of the four previous sets. According to proposition 5, 63 elements at the most constitute that intersection. It stems from proposition 6 that each of the 4 bytes of key K_{Nr} may possibly take 128 values. By iterating the introduction of a fault, the set of values possibly taken on by the keys is reduced by half. After having introduced five faults running, we discover four bytes of the round key. Applying this technique to another row or column, we can manage to discover for other bytes of the last round key. In using about 20 pairs (distorted ciphered output and correct ciphered output), we extract the AES key in full.

Main attack after the $N_r - 2^{th}$ MixColumns and before the $N_r - 1^{th}$ MixColumns. We introduce a random fault into the state at a point unknown to the attacker. That is a generalization of the previous attack. Assuming that the fault may occur at four different places allows us to find out the key K_{Nr}. We thus put ourselves under the conditions of the previous attack so as to determine four sets of Key K_{Nr} possible values. It is necessary that the attack should be repeated using from 40 to 50 distinct pairs (distorted ciphered output and correct ciphered output) for the complete key to be extracted.

Attack after the $N_r - 3^{th}$ MixColumns and before the $N_r - 2^{th}$ MixColumns. We introduce a random fault into the state at a point unknown to the attacker. $N_r - 2^{th}$ ShiftRows spreads each of the four faults over another column of the state. The last MixColumns propagates each fault contained in a column to the whole of it. We can apply the result of the basic attack to every fault contained in a column in order or determine the sets of values possibly taken on by every byte of the last round key. In such a case ten faults are required to get key K_{Nr}.

Attack on hardware device. It is possible to apply the previous patterns of attack to hardware device. Suppose that you can physically modify a hardware AES device. Firstly, compute the outputs from around ten random plaintexts with an AES device. Secondly, modify for instance the component design by

cutting wires lying between two bytes and grounding them (or Vcc) temporarily two rounds before the process ends. It amounts to having a byte of round $N_r - 2$ with a '00' (or 'FF') value. Compute another time the same plaintexts as previously with the tampered device. When the input is a random plaintext, the error generated is a random one. Proceeding along as set out in the previous paragraph, we can extract key K_{Nr}.

3.2 Basic Principle of Attacks

Let us analyse basic attack, we denote by F the erroneous state. Now we are going to describe each step of the state from the $N_r - 1^{th}$ MixColumns to the end. Assume that we replace the first element of the state by an unknown value. Let $\varepsilon \in GF(2^8) - \{0\}$, and get

$$F_{N_r-1,Sh}[0] = S_{N_r-1,Sh}[0] + \varepsilon.$$

$$F_{N_r-1,Sh} = S_{N_r-1,Sh} + \begin{pmatrix} \varepsilon & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

$$F_{N_r-1,M} = S_{N_r-1,M} + A_0.\begin{pmatrix} \varepsilon & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = S_{N_r-1,M} + \begin{pmatrix} \text{'02'}.\varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \text{'03'}.\varepsilon & 0 & 0 & 0 \end{pmatrix}.$$

$$F_{N_r-1,A} = S_{N_r-1,A} + \begin{pmatrix} \text{'02'}.\varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \text{'03'}.\varepsilon & 0 & 0 & 0 \end{pmatrix}.$$

We can define $\varepsilon_0^{'}$, $\varepsilon_1^{'}$, $\varepsilon_2^{'}$, $\varepsilon_3^{'}$ (the differential faults) by the equations

$$\begin{cases} s(x_0 + \text{'02'}.\varepsilon) = s(x_0) + \varepsilon'_0 \\ s(x_1 + \varepsilon) = s(x_1) + \varepsilon'_1 \\ s(x_2 + \varepsilon) = s(x_2) + \varepsilon'_2 \\ s(x_3 + \text{'03'}.\varepsilon) = s(x_3) + \varepsilon'_3 \end{cases} \tag{1}$$

Consequently

$$F_{N_r,Su} = S_{N_r,Su} + \begin{pmatrix} \varepsilon_0^{'} & 0 & 0 & 0 \\ \varepsilon_1^{'} & 0 & 0 & 0 \\ \varepsilon_2^{'} & 0 & 0 & 0 \\ \varepsilon_3^{'} & 0 & 0 & 0 \end{pmatrix}.$$

$$F_{N_r,Sh} = S_{N_r,Sh} + \begin{pmatrix} \varepsilon_0^{'} & 0 & 0 & 0 \\ 0 & 0 & 0 & \varepsilon_1^{'} \\ 0 & 0 & \varepsilon_2^{'} & 0 \\ 0 & \varepsilon_3^{'} & 0 & 0 \end{pmatrix}.$$

$$F_{N_r,A} = S_{N_r,A} + \begin{pmatrix} \varepsilon_0^{'} & 0 & 0 & 0 \\ 0 & 0 & 0 & \varepsilon_1^{'} \\ 0 & 0 & \varepsilon_2^{'} & 0 \\ 0 & \varepsilon_3^{'} & 0 & 0 \end{pmatrix}.$$

$F_{N_r,A}$ is the erroneous output of a cipher. Comparing the states $F_{N_r,A}$ with $S_{N_r,A}$, the values of $\varepsilon_0^{'}$, $\varepsilon_1^{'}$, $\varepsilon_2^{'}$ and $\varepsilon_3^{'}$ can be easily found.

The only operation that could give a clue about the key K_{N_r} is the last SubBytes transformation. Consequently we have four equations where $x_0, x_1, x_2, x_3, \varepsilon$ are unknown variables. We want to solve the system of equations (in x_i and ε) (1). All these equations belong to a generalized equation :

$$s(x + c.\varepsilon) + s(x) = \varepsilon', \tag{2}$$

where $c =$'01', '02' or '03'. Let us analyse it.

Remark 3. The map defined by $x \mapsto x^{-1}$ in $GF(2^8)$ is differentially 2 or 4 uniform [7]. This map has other favorable cryptographic properties: large distance from affine functions, high non-linear order and efficient computability.

Definition 1. *Consider the linear application in $GF(2)$:*

$$\begin{aligned} l : GF(2^8) &\longrightarrow GF(2^8) \\ x &\longmapsto x^2 + x \end{aligned}$$

Let us represent by $E_1 = Im(l)$ the $GF(2)$-vector space image of l. We have $dim_{GF(2)}(E_1) = 7$. If $\theta \in E_1$, then there are two solutions $x_1, x_2 \in GF(2^8)$ to the equation $x^2 + x = \theta$, and the solutions satisfy the equation $x_2 = x_1 + 1$.

Definition 2. *Let $\lambda \in GF(2^8)$, $\lambda \neq 0$ and define ϕ_λ a $GF(2)$-vector spaces isomorphism:*

$$\begin{aligned} \phi_\lambda : GF(2^8) &\longrightarrow GF(2^8) \\ x &\longmapsto \lambda.x \end{aligned}$$

and let $E_\lambda = Im(\phi_\lambda|_{E_1})$ be the $GF(2)$-vector space image of ϕ_λ restricted to E_1. Moreover $dim_{GF(2)}(E_\lambda) = 7$.

Proposition 1. *There is a bijective application ϕ between $E_1^*(= E_1 - \{0\})$ and $S_{c,\varepsilon'}$.*

$$\begin{aligned} \phi : E_1^* &\longrightarrow S_{c,\varepsilon'} \\ t &\longmapsto (c(a^{-1} * \varepsilon').t)^{-1}. \end{aligned}$$

$S_{c,\varepsilon'}$ have 127 elements.

Proof. Let $\varepsilon \in S_{c,\varepsilon'}$, then $\exists x \in GF(2^8)$ such that (2) holds.
Let us assume $x \neq 0$ and $x \neq c.\varepsilon$, we get

$$x^2 + c.\varepsilon.x = (a^{-1} * \varepsilon')^{-1}.c.\varepsilon.$$

We represent by $t = x.(c.\varepsilon)^{-1} \in GF(2^8) - \{0\}$, then we have

$$t^2 + t = (a^{-1} * \varepsilon')^{-1}.(c.\varepsilon)^{-1}. \tag{3}$$

Therefore $(a^{-1} * \varepsilon')^{-1}(c.\varepsilon)^{-1} \in E_1^*$. Reciprocally for $\theta \in E_1^*$ we can define $(a^{-1} * \varepsilon')^{-1}.(c.\theta)^{-1} \in S_{c,\varepsilon'}$.
Let us assume $x = 0$ or $x = c.\varepsilon$, (2) becomes $a * (c.\varepsilon)^{-1} = \varepsilon'$. We obtain $\varepsilon = ((a^{-1} * \varepsilon').c)^{-1}$. This case is included in the previous one because $1 \in E_1^*$. We observe that when $\theta = 1$, four solutions in x to the equation (2) can be found. In brief, a bijection map exists between E_1^* and $S_{c,\varepsilon'}$:

$$\begin{array}{ccccc} E_1^* & \xrightarrow{\phi_\lambda} & E_\lambda - \{0\} & \longrightarrow & S_{c,\varepsilon'} \\ t & \longmapsto & \lambda.t & \longmapsto & (\lambda.t)^{-1}. \end{array}$$

where $\lambda = c(a^{-1} * \varepsilon')$.

Proposition 2. *The following statements hold for* $\lambda_1, \lambda_2 \in GF(2^8) - \{0\}$*:*

$$dim_{GF(2)}(E_{\lambda_1} \cap E_{\lambda_2}) = \begin{cases} 7 & \text{If } \lambda_1 = \lambda_2 \\ 6 & \text{Otherwise} \end{cases}$$

Proof. Proving that following lemma 1 holds true is enough to prove Proposition 2 holds true too.

Lemma 1. *For* $\lambda_1, \lambda_2 \in GF(2^8) - \{0\}$*, we get*

$$E_{\lambda_1} = E_{\lambda_2} \Longleftrightarrow \lambda_1 = \lambda_2.$$

Proof. This lemma is equivalent to this proposition: for $\lambda \in GF(2^8) - \{0\}$,

$$E_\lambda = E_1 \Longleftrightarrow \lambda = 1.$$

Let us prove this statement and assume that $\lambda E_1 = E_1$. Remark that $E_1 = \{t = \{t_7 t_6 \cdots t_0\}_b \in GF(2^8) - \{0\} \ : \ t_7 = t_5\}$. Hence $\{1, x, x^2, x^3, x^4, x^6, x^5 + x^7\}$ is a basis of E_1. Let us multiply the basis vectors v_i of E_1 by $\lambda = \{\lambda_7 \cdots \lambda_0\}_b$. As $\lambda v_i \in E_1$, we have $(\lambda v_i)_7 = (\lambda v_i)_5$. We obtain 7 relations ($\lambda_7 = \lambda_5$, $\lambda_6 = \lambda_4$, $\lambda_5 = \lambda_3 + \lambda_7$, $\lambda_4 = \lambda_6 + \lambda_2 + \lambda_7$, $\lambda_7 + \lambda_3 = \lambda_5 + \lambda_1 + \lambda_6$, $\lambda_5 + \lambda_1 = \lambda_3 + \lambda_4$, $\lambda_6 + \lambda_5 = \lambda_7 + \lambda_3$). We solve this system to obtain $\lambda_7 = \lambda_6 = \lambda_5 = \lambda_4 = \lambda_3 = \lambda_2 = \lambda_1 = 0$. The solution $\lambda = 0$ is not right. We can infer that $\lambda = 1$.

Proposition 3. *For* $\lambda_1, \lambda_2, \lambda_3 \in GF(2^8) - \{0\}$*, we get:*

$$dim_{GF(2)}(E_{\lambda_1} \cap E_{\lambda_2} \cap E_{\lambda_3}) = \begin{cases} 7 & \text{If } \lambda_1 = \lambda_2 = \lambda_3 \\ 6 & \text{If } rank_{GF(2)}\{\lambda_1^{-1}, \lambda_2^{-1}, \lambda_3^{-1}\} = 2 \\ 5 & \text{Otherwise} \end{cases}$$

Proof. It follows from proposition 2 and this following lemma

Lemma 2. *For $\lambda_1, \lambda_2, \lambda_3 \in GF(2^8) - \{0\}$, we get*

$$E_{\lambda_1} \cap E_{\lambda_3} = E_{\lambda_2} \cap E_{\lambda_3} \Longleftrightarrow \lambda_3^{-1} = \lambda_1^{-1} + \lambda_2^{-1} \text{ or } \lambda_1 = \lambda_2.$$

Proof. 1. $\Leftarrow$

Let $x \in E_{\lambda_1} \cap E_{\lambda_3}$, then $\exists y, t \in E_1$ such that $x = \lambda_1.y = \lambda_3.t$.

$$y = \lambda_1^{-1}.\lambda_3.t = \lambda_2^{-1}.\lambda_3.t + t,$$

$$y - t = \lambda_2^{-1}.\lambda_3.t \in E_1,$$

and

$$x = \lambda_3.t = \lambda_2.(y - t) \in E_{\lambda_2}$$

2. $\Rightarrow$

Let us assume that $\lambda_1 \neq \lambda_2$, and show that $\forall t \in E_1,\ \lambda_3.(\lambda_1^{-1} + \lambda_2^{-1}).t \in E_1$. Let $x = \lambda_3.t \in E_{\lambda_3}$:

- If $x \in E_{\lambda_1}$ then $x \in E_{\lambda_2}$ and $\exists s_1, s_2 \in E_1$ so that $x = \lambda_1.s_1 = \lambda_2.s_2$ and we get $\lambda_3.(\lambda_1^{-1} + \lambda_2^{-1}).t = s_1 + s_2 \in E_1$.
- If $x \notin E_{\lambda_1}$ then $x \notin E_{\lambda_2}$ and we get $\lambda_1^{-1}.x \notin E_1$ and $\lambda_2^{-1}.x \notin E_1$. We have $\lambda_3.(\lambda_1^{-1} + \lambda_2^{-1}).t = \lambda_1^{-1}.x + \lambda_2^{-1}.x \in E_1$ (because $\forall u \notin E_1$ and $\forall v \notin E_1$ then $u + v \in E_1$).

We showed that $E_{\lambda_3.(\lambda_1^{-1}+\lambda_2^{-1})} = E_1$ and with the lemma 1 we get $\lambda_3^{-1} = \lambda_1^{-1} + \lambda_2^{-1}$.

Proposition 4. *Finally for $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \in GF(2^8) - \{0\}$, we get:*

$$dim_{GF(2)}(E_{\lambda_1} \cap E_{\lambda_2} \cap E_{\lambda_3} \cap E_{\lambda_4}) = \begin{cases} 7 & \text{If } \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 \\ 6 & \text{If } rank_{GF(2)}\{\lambda_1^{-1}, \lambda_2^{-1}, \lambda_3^{-1}, \lambda_4^{-1}\} = 2 \\ 5 & \text{If } rank_{GF(2)}\{\lambda_1^{-1}, \lambda_2^{-1}, \lambda_3^{-1}, \lambda_4^{-1}\} = 3 \\ 4 & \text{Otherwise} \end{cases}$$

Definition 3. *We define the set of solutions to (2) in ε by*

$$S_{c,\varepsilon'} = \left\{\varepsilon \in GF(2^8) :\ \exists x \in GF(2^8),\ s(x + c.\varepsilon) + s(x) = \varepsilon'\right\}.$$

Definition 4. *We considered four equations in a different way, but the fault introduced is common to these four equations. This is the reason why we introduce the set of possibly introduced faults S:*

$$\Pi = S_{2,\varepsilon'_0} \bigcap S_{1,\varepsilon'_1} \bigcap S_{1,\varepsilon'_2} \bigcap S_{3,\varepsilon'_3}.$$

Π has a smaller cardinal than $S_{c,\varepsilon}$. This allows one to specify more accurately the set of values possibly taken on the faults. Thus the key can be found out by introducing fewer faults.

Proposition 5. *If two of the four following values* $2^{-1}.\varepsilon'_0,\ \varepsilon'_1,\ \varepsilon'_2,\ 3^{-1}.\varepsilon'_3$ *are not equal, we get*

$$Card\ \left(S_{2,\varepsilon'_0} \bigcap S_{1,\varepsilon'_1} \bigcap S_{1,\varepsilon'_2} \bigcap S_{3,\varepsilon'_3}\right) \leq 63.$$

Proposition 6. *For a differential fault* ε'*, let* $\varepsilon \in \Pi \cap S_{c,\varepsilon'}$ *be a fault value,* $\theta = ((a^{-1} * \varepsilon').c.\varepsilon)^{-1} \in E_1^*$ *and* α, β *the two solutions (in* $GF(2^8)$*) to the equation* $t^2 + t = \theta$*. The possible values of key* $K_{N_r}[i]$ *(for a certain* i*, being the index of element in the state) are*

- *If* $\theta \neq 1$ *then* $K_{N_r}[i]$ *can possibly take on two values*

$$K_{N_r}[i] = s(c.\varepsilon.\alpha) + F_{N_r,A}[i] \text{ or } K_{N_r}[i] = s(c.\varepsilon.\beta) + F_{N_r,A}[i]$$

- *If* $\theta = 1$ *then* $K_{N_r}[i]$ *can possibly take on four values*

$$K_{N_r}[i] = s(c.\varepsilon.\alpha) + F_{N_r,A}[i] \text{ or } K_{N_r}[i] = s(c.\varepsilon.\beta) + F_{N_r,A}[i]$$

$$\text{or } K_{N_r}[i] = b + F_{N_r,A}[i] \text{ or } K_{N_r}[i] = s(c.\varepsilon) + F_{N_r,A}[i]$$

Proof.
- If $\theta \neq 1$ then we know that $\theta \in E_1$, and there are two solutions α, β to $t^2 + t = \theta$. We can deduce two solutions from (2) noted $\{x_1, x_2\}$, where $x_1 = c.\varepsilon.\alpha$ and $x_2 = c.\varepsilon.\beta$.
- If $\theta = 1$, we know that $1 \in E_1$, and there are two solutions α, β to $t^2 + t = 1$. We can deduce two solutions from (2) noted $\{x_1, x_2\}$, where $x_1 = c.\varepsilon.\alpha$ and $x_2 = c.\varepsilon.\beta$. Moreover there are also two trivial solutions to (2): $x_3 = 0$ and $x_4 = c.\varepsilon$.

Once we get a solution x to (2), $K_{N_r}[i]$ value can be easily inferred.

By applying this proposition to the four erroneous elements of the state, we can deduce four sets of values that $K_{N_r}[0]$, $K_{N_r}[7]$, $K_{N_r}[10]$ and $K_{N_r}[13]$ can taken on. By introducing repeatedly a fault into a computation, and considering the intersection of those four sets we soon get the true value for $K_{N_r}[0]$, $K_{N_r}[7]$, $K_{N_r}[10]$ and $K_{N_r}[13]$.

3.3 Probability Complexity

We want to know how many pairs we need to crack the cipher.

Proposition 7. *In average, 9 pairs are required to find 4 bytes of the* K_{N_r} *round key. Alike, 11 pairs are required for the Basic attack, 9 for the Extended ones and 34 for the main one.*

Proof. Denote by Card K the cardinal of possible values taken on by any byte of K_{N_r}. Under propositions 4 and 6, supposing they have been distributed at random, probabilities are as follows:

- $\frac{256 \cdot 255 \cdot 254 \cdot 253}{256^4}$ that Card $K = 32$
- $\frac{C_4^2 \cdot 256 \cdot 255 \cdot 254}{256^4}$ that Card $K = 64$

– $\frac{(C_4^3+C_4^2/2)\cdot 256\cdot 255}{256^4}$ that Card $K = 128$
– $\frac{256}{256^4}$ that Card $K = 256$

On average, Card $K = 32.75146103$ when the position of the fault is known. In general, such information is not available so the four possibilities have to be tried: Card $K = 4 \cdot 32.75146103 = 131.0058441$. Each time, the set of possible values is divided by $\frac{256}{131.0058441} \approx 1.954111298$. To bring the number of possibilities down to one, $\ln 256 / \ln 1.954111298 \approx 8.277180940$ pairs are required. Hence, nine faulty ciphertexts have to be used to find 4 bytes of the K_{N_r} round key. In the Extended attacks, 16/4=4 more pairs are required but the four errors are treated simultaneously. Hence in these cases, nine faulty ciphertexts have to be used to find the whole K_{N_r} round key. In the basic attack, the errors have to be distribued all over the other bytes; 16/4=4 more pairs are required:

$$4\frac{\ln 256}{\ln \frac{256}{32.75146103}} \approx 10.78707691.$$

A Example

We will be using the same example as in Appendix B of [1]. The following diagram shows the values in the final states for a block length and a Cipher Key length of 16 bytes each (i.e., $Nb = 4$ and $Nk = 4$).

`Input=` '32 43 F6 A8 88 5A 30 8D 31 31 98 A2 E0 37 07 34'
`Cipher Key=` '2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C'
`Output=` '39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32'

The spreading of the fault is highlighted:

After ShiftRows 9

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Fault injected 1E

99	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

After MixColumns

7B	40	A3	4C
29	D4	70	9F
8A	E4	3A	42
CF	A5	A6	BC

$\oplus$ K_9

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6E

After AddRoundKey 9

D7	59	8B	1B
5E	2E	A1	C3
EC	38	13	42
3C	84	E7	D2

After SubBytes 10

0E	CB	3D	AF
58	31	32	2E
CE	07	7D	2C
EB	5F	94	B5

After ShiftRows 10

0E	CB	3D	AF
31	32	2E	58
7D	2C	CE	07
B5	EB	5F	94

$\oplus$ value of K_{10}

D0	C9	E1	B6
14	EE	3F	63
F9	25	0C	0C
A8	89	C8	A6

Output with Faults

DE	02	DC	19
25	DC	11	3B
84	09	C2	0B
1D	62	97	32

The error injected into the state, generates four further errors (differential faults) in the final state.

Output with faults					Output without fault					Error			
DE	02	DC	19	⊕	39	02	DC	19	=	E7	00	00	00
25	DC	11	3B		25	DC	11	6A		00	00	00	51
84	09	C2	0B		84	09	85	0B		00	00	47	00
1D	62	97	32		1D	FB	97	32		00	99	00	00

The differential faults are $\varepsilon'_0 =$ 'E7', $\varepsilon'_1 =$ '51', $\varepsilon'_2 =$ '47' and $\varepsilon'_3 =$ '99'. The following four equations have now to be worked out:

$$\begin{aligned} s(x_0 \oplus \text{'02'}.\varepsilon) &= s(x_0) \oplus \text{'E7'} \\ s(x_1 \oplus \varepsilon) &= s(x_1) \oplus \text{'51'} \\ s(x_2 \oplus \varepsilon) &= s(x_2) \oplus \text{'47'} \\ s(x_3 \oplus \text{'03'}.\varepsilon) &= s(x_3) \oplus \text{'99'} \end{aligned}$$

As defined previously,

$$E_1^* = \{\text{'01'..'1F','40'..'5F','A0'..'BF','E0'..'FF'}\}.$$

Let

$$\begin{aligned} \lambda_0 &= \text{'02'}.(a^{-1} * \text{'E7'}) = \text{'12'} \\ \lambda_1 &= \text{'01'}.(a^{-1} * \text{'51'}) = \text{'7C'} \\ \lambda_2 &= \text{'01'}.(a^{-1} * \text{'47'}) = \text{'65'} \\ \lambda_3 &= \text{'03'}.(a^{-1} * \text{'99'}) = \text{'B0'} \end{aligned}$$

We have a single linear relation over $GF(2)$ between $\lambda_0, \lambda_1, \lambda_2, \lambda_3$: $\lambda_0^{-1} \oplus \lambda_3^{-1} = \lambda_2^{-1}$. Therefore we get

$$\text{card}\left(S_{2,\text{'E7'}} \bigcap S_{1,\text{'51'}} \bigcap S_{1,\text{'47'}} \bigcap S_{3,\text{'99'}}\right) = 2^5 - 1 = 31.$$

Using the relation $S_{c,\varepsilon'} = \{(c.(a^{-1} * \varepsilon').t)^{-1}, \quad t \in E_1^*\}$, we can easily (and quickly!) compute

$$\begin{aligned} &S_{2,\text{'E7'}} \bigcap S_{1,\text{'51'}} \bigcap S_{1,\text{'47'}} \bigcap S_{3,\text{'99'}} \\ &= \{\text{'01', '04', '13', } \textbf{'1E'}\text{, '21', '27', '33', '3B', '48', '4D', '50', '53', '55', '5D',} \\ &\quad \text{'64', '65','7E', '7F', '80', '83', '8D', '8F', '93', 'A7', 'A8', 'A9', 'AB',} \\ &\quad \text{'B3', 'B8', 'C9', 'F6'}\} \end{aligned}$$

Using the proposition 6, we get a set of values possibly taken on by $K_{10}[0]$ (the true value is 'D0'):

$K_{10}[0] \in$ {'03', '06', '09', '0C', '10', '15', '1A', '1F', '21', '24', '2B', '2E', '32', '37', '38', '3D', '43', '46', '49', '4C', '50', '55', '5F', '61', '64', '6B', '6E', '72', '77', '78', '7D', '83', '86', '89', '8C', '90', '95', '9A', '9F', 'A1', 'A4', 'AB', 'AE', 'B2', 'B7', 'B8', 'C3', 'C6', 'C9', 'CC', **'D0'**, 'D5', 'DA', 'DF', 'E1', 'E4', 'EB', 'EE', 'F2', 'F7', 'F8', 'FD'}

By introducing a second fault into the very same place in the state, we reduce by half the set of values possibly taken on by $K_{10}[0]$. Introducing a fault five times over, we can find out the one and only true value of $K_{10}[0]$. Of course, we can also analyse the other three bytes $K_{10}[7]$, $K_{10}[10]$ and $K_{10}[13]$ as we analyse the first one. Doing so, we can find out the true values of 4 bytes in key K_{10}. In order to determine the other 4 bytes of the key K_{10}, we have to introduce a fault into any other place in the state and repeat the above described process.

B Example 2

We compute ten pairs of (correct/faulty) ciphertexts with an AES-128 program. We know that we injected a fault into a byte between the MixColumn7 and MixColumn8 operations. Still we know neither its position nor its value (for the readers interesting in repeating those computations, the fault injected replaces State[0] by 'FF' just before MixColumn8 in the following examples).

Correct CipherText	Faulty CipherText
'467A7363D54E58BB25B135FABFA0EA49'	'4F41429299FFBE374514034F07BF4B19'
'9EEE064F55D3B0F5DDC0002E33CDCBEE'	'DF0C7EBA22B9131D83ADE91D223ADD6F'
'5EB4F21A7493ED8EA431B8E6B73FA924'	'2A2B37C7B08482E43063040D357E7F92'
'1A6FC7471E2A43460AE4F29296CCB731'	'A83C77CE284BCAF64DDE12DF58D8B9DB'
'7711043CE69C252E7219FBB12371CD66'	'B7FF53C4D24FF23DF8618B229F8522CB'
'3253954160E455152D77F8A0748B0CEB'	'CA499E9FB8BC82E3120C489FACDC654D'
'538FFA5AD396AE973EDB8C50B44EC54C'	'5C655A7BDE74DED49BE0D36BF27662B8'
'1663332626442DA55F3362384FF1144B'	'51116D1D351518FC7021931A20AC49A0'
'F9CC9D6B31BC0EA27D4E239DBBC943CD'	'75EC4D4F1122E1B7F3F8AD578AA2CD11'
'8C7D0ABC6CDD13D0BD268469ED34FADB'	'672A2B22556974C304C8C7DCD499ABAD'

The first pair shows an differential error result, which can be split into four matrices. When every column preceding MixColumn9 is injected with a fault, the matrices show as follows:

$$\begin{pmatrix} \text{'09'} & \text{'4C'} & \text{'60'} & \text{'B8'} \\ \text{'3B'} & \text{'B1'} & \text{'A5'} & \text{'1F'} \\ \text{'31'} & \text{'E6'} & \text{'36'} & \text{'A1'} \\ \text{'F1'} & \text{'8C'} & \text{'B5'} & \text{'50'} \end{pmatrix} = \begin{pmatrix} \text{'09'} & \text{'00'} & \text{'00'} & \text{'00'} \\ \text{'00'} & \text{'00'} & \text{'00'} & \text{'1F'} \\ \text{'00'} & \text{'00'} & \text{'36'} & \text{'00'} \\ \text{'00'} & \text{'8C'} & \text{'00'} & \text{'00'} \end{pmatrix} \oplus \begin{pmatrix} \text{'00'} & \text{'4C'} & \text{'00'} & \text{'00'} \\ \text{'3B'} & \text{'00'} & \text{'00'} & \text{'00'} \\ \text{'00'} & \text{'00'} & \text{'00'} & \text{'A1'} \\ \text{'00'} & \text{'00'} & \text{'B5'} & \text{'00'} \end{pmatrix}$$
$$\oplus \begin{pmatrix} \text{'00'} & \text{'00'} & \text{'60'} & \text{'00'} \\ \text{'00'} & \text{'B1'} & \text{'00'} & \text{'00'} \\ \text{'31'} & \text{'00'} & \text{'00'} & \text{'00'} \\ \text{'00'} & \text{'00'} & \text{'00'} & \text{'50'} \end{pmatrix} \oplus \begin{pmatrix} \text{'00'} & \text{'00'} & \text{'00'} & \text{'B8'} \\ \text{'00'} & \text{'00'} & \text{'A5'} & \text{'00'} \\ \text{'00'} & \text{'E6'} & \text{'00'} & \text{'00'} \\ \text{'F1'} & \text{'00'} & \text{'00'} & \text{'00'} \end{pmatrix}$$

By considering the first matrice, the possible values of the key can be reduced (for the first matrice, the error belongs to the first column). If the error lies in the first line: we have to compute the following

$$L_\varepsilon(1) = S_{2,\text{'09'}} \bigcap S_{1,\text{'1F'}} \bigcap S_{1,\text{'36'}} \bigcap S_{3,\text{'8C'}}$$

We have Card $L_\varepsilon(1) = 15$, hence, using $CipherText[0]$ ='46' and Proposition 6, we get the first set $PK_1[0]$ of values possibly taken on by $K_{10}[0]$.

If the error lies in the second line: we have to compute the following

$$L_\varepsilon(2) = S_{3,\text{'09'}} \bigcap S_{2,\text{'1F'}} \bigcap S_{1,\text{'36'}} \bigcap S_{1,\text{'8C'}}$$

We have Card $L_\varepsilon(2) = 15$, hence, using $CipherText[0]$ ='46' and Proposition 6, we get the second set $PK_2[0]$ of values possibly taken on by $K_{10}[0]$.

We go over the same steps in the cases where the error lies in the third or in the fourth line. We refer to the resulting sets by $PK_3[0]$ and $PK_4[0]$.

Finally, $K_{10}[0]$ lies in

$$PK_1[0] \cup PK_2[0] \cup PK_3[0] \cup PK_4[0]$$

which have 96 elements altogether. So we reduce the 256 values of $K_{10}[0]$ to only 96 possibilities.

We go over the same steps applied to the second, third and fourth matrices (which impact on the other bytes of K_{10}) and find

$$K_{10} = \text{'D014F9A8C9EE2589E13F0CC8B6630CA6'}$$

with ten faulty ciphertexts.

References

1. FIPS PUB 197 : *Avanced Encryption Standard*, `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`
2. Joan Daemen and Vincent Rijmen, *The Design of Rijndael, AES – The Advanced Encryption Standard*, Springer-Verlag 2002, (238 pp.).
3. Ross J. Anderson, Markus G. Kuhn: *Tamper Resistance – a Cautionary Note*, The Second USENIX Workshop on Electronic Commerce Proceedings, Oakland, California, November 18–21, 1996, pp 1–11, ISBN 1-880446-83-9.
4. E. Biham and A.Shamir, *Differential Fault Analysis of Secret Key Cryptosystems*, CS 0910, Proceedings of Crypto'97.
5. Boneh, DeMillo, and Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'97, pp. 37–51, 1997.
6. Joan Daemen, *Annex to AES Proposal Rijndael*, `http://www.esat.kuleuven.ac.be/~rijmen/rijndael/PropCorr.PDF`,1998.
7. K. Nyberg, *Differentially uniform mappings for cryptography*, Advances in Cryptology, Proceedings Eurocrypt'93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 55–64.
8. G. Letourneux, *Rapport de stage EDSI : Etude et implémentation de l'AES, Attaques DPA et DFA*, August 30, 2002.