Differential Fault Analysis on A.E.S.

O. Vivolo[*], G. Letourneux[†], P. Dusart[‡]

01/10/2002

Abstract

This paper investigates Differential Fault Analysis (DFA) on the Advanced Encryption Standard (AES). Following the Bellcore fault model [2], we assume the injection of a single-byte error during the final rounds of encryption. We show how such faults propagate through AES transformations and demonstrate that the last round key can be derived with only a small number of faulty ciphertexts. From this information, the original cipher key can be efficiently reconstructed. Numerical simulations confirm the practicality of the attack, requiring fewer than ten faulty outputs to recover a complete subkey. Our results highlight the vulnerability of AES implementations in tamper-resistant devices, such as smart cards, when exposed to fault injections. These findings emphasize the importance of incorporating robust error-detection and fault-resilient countermeasures in secure hardware.

The first author served as the primary writer of the manuscript. The second author, working as his intern, conducted the initial numerical experiments. Finally, the third author contributed as the final reviewer, ensuring the correction of errors. In the earlier versions of the article, the authors were listed in alphabetical order.

Contents

[*]Orange SA, 2 Av. de Belle Fontaine, 35510 Cesson-Sévigné, France
[†]E.D.S.I. Atalis 1, 1, rue de Paris, 35510 Cesson-Sévigné, France
[‡]XLIM-DMI Faculté des Sciences et Techniques, 123 avenue Albert Thomas, 87060 Limoges Cedex, France

1 Introduction

The attack follows the Bellcore fundamental assumption that by exposing a sealed tamperproof device such as a smart card to certain physical effects (e.g., ionizing or microwave radiation), one can induce with reasonable probability a fault at a random bit location in one of the registers at some random intermediate stage in the cryptographic computation. Both the bit location and the round number are unknown to the attacker.

We further assume that the attacker is in physical possesion of the tamperproof-device, so that he can repeat the experiment with the same cleartext and key but without applying the external physical effects. As a result, he obtains two ciphertexts derived from the same (unknown) cleartext and key, where one of the ciphertexts is correct and the other is the result of a computation corrupted by a single error during the computation. For the sake of simplicity, we assume that one byte of the state before the MixColumn transformation of the nine round is replaced by a unknown value. This attack finds the last subkey. Once this subkey is known, we can find easily the key. The induced fault is going to be propagated by the MixColumn and spread on 4 bytes of the state. For each byte is possible to find a set of possible value of induced fault, and then a set of possible values for the roundkey 10.

We have implemented this attack on a personal computer. Our analysis program found the whole last subkey given less than 10 ciphertexts.

2 The description of the AES

In this article, we use a different description from original AES submission FIPS PUB 197 [1], we descibe AES using matrix on $GF(2^8)$ but we keep 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 of the repeated application of a round transformation on the state. The number of rounds is denoted 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 matrix 4x4 with its coefficients in $GF(2^8)$) to another state in $M_4(GF(2^8))$. The key K is expanded in N_r sub-keys noted $K_i \in M_4(GF(2^8))$.

A round of an encryption with AES is composed of four main operations :

1. AddRoundKey
2. MixColumn
3. SubBytes
4. ShiftRows

2.1 Representation chosen for $GF(2^8)$

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

2.2 Notation used in this article

We use four notations for representing an element in $GF(2^8)$, which equivalent to one another:

1. $x^7 + x^6 + x^4 + x^2$, the polynomial notation
2. $\{11010100\}$, the binary notation
3. 'D4', the hexadecimal notation
4. 212, the decimal notation

2.3 AddRoundKey for i^{th} round

The AddRoundKey transformation consist in an addition of matrices in $M_4(GF(2^8))$ between the state and the sub-key of the i^{th} round. We denote by $S_{i,A}$ the state after the i^{th} AddRoundKey.

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

2.4 SubByte for i^{th} round

The SubByte transformation consist to apply on each element of the matrix S an elementary transformation s. We denote by $S_{i,Su}$ the state after the i^{th} SubByte.

$$\begin{array}{rcl} M_4(GF(2^8)) & \longrightarrow & M_4(GF(2^8)) \\ S = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \\ S[4] & S[8] & S[12] & S[16] \end{pmatrix} & \longmapsto & S_{i,Su} = \begin{pmatrix} s(S[1]) & s(S[5]) & s(S[9]) & s(S[13]) \\ s(S[2]) & s(S[6]) & s(S[10]) & s(S[14]) \\ s(S[3]) & s(S[7]) & s(S[11]) & s(S[15]) \\ s(S[4]) & s(S[8]) & s(S[12]) & s(S[16]) \end{pmatrix}, \end{array}$$

where s is the non linear application defined by

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

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} is see as a $GF(2)$-vector. The value of $b =$ '63'$\in GF(2^8)$ and

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

2.5 MixColumn for i^{th} round

The MixColumn transformation consist in a multiplication of matrices in $M_4(GF(2^8))$, between the state and a fixed matrix A_0 of $M_4(GF(2^8))$. We denote by $S_{i,M}$ the state after the i^{th} MixColumn.

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

where A_0 is defined by

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

2.6 ShiftRows for i^{th} round

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

$$\begin{array}{ccc} M_4(GF(2^8)) & \longrightarrow & M_4(GF(2^8)) \\ S = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \\ S[4] & S[8] & S[12] & S[16] \end{pmatrix} & \longmapsto & S_{i,Sh} = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[6] & S[10] & S[14] & S[2] \\ S[11] & S[15] & S[3] & S[7] \\ S[16] & S[4] & S[8] & S[12] \end{pmatrix}. \end{array}$$

3 The description of the attack on computation of AES

First, we are going describe an attack on AES in a simple case and after that we will see how we can generalize this attack. The goal of the attack is to recover the key K_{Nr}. Once we discover the key K_{Nr}, it is easy to get the key K, see appendix A.

3.1 Principle of the attack

We suppose that we can change a single element of the state after the ShiftRow of the $N_r - 1$ round and we know the index of the faulty element of state (this last supposition can be omitted, it is so easier to explain the mechanism). The new value of the element of the state is supposed unknown. The fault ε is propagated over four bytes on the output state. For each modified elements on the output state, we find a set of possible fault ε. Moreover we can intersect the possible values ε for these four sets, we obtain a small set thus reducing the number of required ciphertext for the full analysis. Finally for each fault, we deduce some possible values of four elements of the last roundkey. Repeating ciphertexts, we find four bytes of roundkey 10.

This attack still works even with more general assumptions on the fault locations, such as faults without knowing the fault locations before the 9^{th} MixColumn transformation. We also expect that faults in round 8 (before the 8^{th} MixColumn transformation) might be useful for the analysis, thus growing the number of required ciphertext for the full analysis. With our example, we need to ten ciphertexts to get four bytes of roundkey 10, when we do not assume on the fault locations.

3.2 Example

We use the same example than Appendix B of [1]. The following diagram shows the values in the final States array as the Cipher progresses 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 fault propagation appears in gray color and in hexadecimal notation:

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 Mixcolumn

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 injected error in the state, give four errors in the final state.

3.3 How the injected error act on the final state

We denote by F the faulty state. Now we describe each step from the $N_r - 1^{th}$ MixColumn to the end, and assume that we replace the first element of the state by an unknwon value. Let $\varepsilon \in GF(2^8) - \{0\}$ defined by

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

3.3.1 Fault modification

Obviously

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

3.3.2 Effect on MixColumn

$$F_{N_r-1,M} = S_{N_r-1,M} + A_0. \begin{pmatrix} \varepsilon & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = S_{N_r-1,M} + \begin{pmatrix} 2.\varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ 3.\varepsilon & 0 & 0 & 0 \end{pmatrix}.$$

3.3.3 Effect on AddRoundKey

$$F_{N_r-1,A} = S_{N_r-1,A} + \begin{pmatrix} 2.\varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ \varepsilon & 0 & 0 & 0 \\ 3.\varepsilon & 0 & 0 & 0 \end{pmatrix}.$$

3.3.4 Effect on last SubBytes

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

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

3.3.5 Effect after last ShiftRows

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

3.3.6 Effect after last AddRoundKey

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

$F_{N_r,A}$ is the faulty output for a cipher. Comparing the states $F_{N_r,A}$ and $S_{N_r,A}$, it is easy to get the values of $\varepsilon_0^{'}, \varepsilon_1^{'}, \varepsilon_2^{'}$ and $\varepsilon_3^{'}$.

3.4 Example

Always, in hexadecimal notation, we find

Output with faults

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

$\oplus$

Output without fault

39	02	DC	19
25	DC	11	6A
84	09	85	0B
1D	FB	97	32

$=$

Error

E7	00	00	00
00	00	00	51
00	00	47	00
00	99	00	00

The differential faults are $\varepsilon_0' =$ 'E7', $\varepsilon_1' =$ '51', $\varepsilon_2' =$ '47' and $\varepsilon_3' =$ '99'.

3.5 Analysis on information bring by fault

The single operation could bring information about the key K_{N_r} is the last SubByte transformation. Consequently we have four equations where $x_0, x_1, x_2, x_3, \varepsilon$ are unknown variables. We want to solve the following equations (in x and ε) :

$$\begin{cases} s(x_0 + 2.\varepsilon) &= s(x_0) + \varepsilon_0' \\ s(x_1 + \varepsilon) &= s(x_1) + \varepsilon_1' \\ s(x_2 + \varepsilon) &= s(x_2) + \varepsilon_2' \\ s(x_3 + 3.\varepsilon) &= s(x_3) + \varepsilon_3' \end{cases}$$

All this equations belong to a generalized equation and let us analyse it

$$s(x + c.\varepsilon) + s(x) = \varepsilon', \tag{1}$$

where $c =$'01', '02' or '03'.

Definition 1 *We define the set of solutions of (1) 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 2 *Consider the linear application over $GF(2)$:*

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

Denote by $E_1 = Im(l)$ be the $GF(2)$-vector space image of l and $dim_{GF(2)}(E_1) = 7$. If $\theta \in E_1$, then there is two solutions $x_1, x_2 \in GF(2^8)$ of equations $x^2 + x = \theta$, and the solutions verify $x_2 = x_1 + 1$.

Definition 3 *Let $\lambda \in GF(2^8)$, $\lambda \neq 0$ and define ϕ_λ an isomorphism of $GF(2)$-vector spaces*

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

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 $S_{c,\varepsilon'}$.*

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

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

Proof: Let $\varepsilon \in S_{c,\varepsilon'}$, then $\exists x \in GF(2^8)$ such that (1) holds.
Assume $x \neq 0$ and $x \neq c.\varepsilon$, we get

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

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

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

Therefore $(a^{-1} * \varepsilon')^{-1}(c.\varepsilon)^{-1} \in E_1^*$. Reciprocally for $\theta \in E_1^*$ we can define $(a^{-1} * \varepsilon')^{-1}.(c.\theta)^{-1} \in S_{c,\varepsilon'}$.
Assume $x = 0$ or $x = c.\varepsilon$, (1) become $a * (c.\varepsilon)^{-1} = \varepsilon'$. We obtain $\varepsilon = ((a^{-1} * \varepsilon').c)^{-1}$, this case is included in the previous case because $1 \in E_1^*$. We see for the case $\theta = 1$, the equation (1) has four solutions in x. In summary, there exist a bijection map 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 $\lambda_1, \lambda_2 \in GF(2^8) - \{0\}$:*

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

Proof: This proof come from the following lemma : □

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 prove this assertion : for $\lambda \in GF(2^8) - \{0\}$,

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

Let us prove this statement and assume that $\lambda E_1 = E_1$ and $\lambda \neq 1$. Therefore $\lambda.1 = \lambda \in E_1$, also $\lambda \in E_1$, and $\lambda.\lambda = \lambda^2 \in E_1$. Consequently $\lambda^3, \lambda^4, \lambda^5, \lambda^6, \lambda^7, \lambda^8 = \lambda^4 + \lambda^3 + \lambda + 1 \in E_1$. The vectors family $(1, \lambda, \lambda^2, \cdots, \lambda^7)$ is spanning E_1 and the vectors are GF(2) linearly independent, therefore $dim_{GF(2)} E_1 = 8$. Contradiction. □

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

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

Proof: The proof come from proposition 2 and this following lemma □

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

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

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

   - If $x \in E_{\lambda_1}$ then $x \in E_{\lambda_2}$ therefore $\exists s_1, s_2 \in E_1$ such 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 $x \notin E_{\lambda_2}$ therefore 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 help of 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:*

$$\hat{E} = E_{\lambda_1} \cap E_{\lambda_2} \cap E_{\lambda_3} \cap E_{\lambda_4},$$

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

Definition 4 *We considered four equations in a different way, but the committed fault is common to these four equations, that is why we introduce the set of possible committed faults S :*

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

Moreover the cardinal of S is smaller than the cardinal of $S_{c,\varepsilon}$. It allows to reduce the space of the faults, and so to use fewer faultly calculations to go back up to the key.

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

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

Proposition 5 *For a differential fault ε', let $\varepsilon \in S \cap S_{c,\varepsilon'}$ be a fault value and define $\theta = ((a^{-1} * \varepsilon').c.\varepsilon)^{-1} \in E_1^*$ and α, β the two solutions (in $GF(2^8)$) of the equation $t^2 + t = \theta$. The possible values of key $K_{N_r}[i]$ (for some i, it is the index of element in the state) are*

- *If $\theta \neq 1$, then there are two possible values of $K_{N_r}[i]$*

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

- *If $\theta = 1$, then there are four possible values of $K_{N_r}[i]$*

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

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

Proof:

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

Once we get a solution x of (1), it is easy to get a possible value of $K_{N_r}[i]$. □

By applying this proposition to the four faulty elements of the state, we can deduce four sets of possible values for $K_{N_r}[0]$, $K_{N_r}[7]$, $K_{N_r}[10]$ and $K_{N_r}[13]$. Then by repeating the insertion of faults in a calculation, and by intersecting these four sets we arrive rather quickly has to have a single value for $K_{N_r}[0]$, $K_{N_r}[7]$, $K_{N_r}[10]$ and $K_{N_r}[13]$.

3.6 Example

Remember our example:

$$\begin{array}{rcl} s(x_0 \oplus 2.\varepsilon) & = & s(x_0) \oplus \text{'E7'} \\ s(x_1 \oplus \varepsilon) & = & s(x_1) \oplus \text{'51'} \\ s(x_2 \oplus \varepsilon) & = & s(x_2) \oplus \text{'47'} \\ s(x_3 \oplus 3.\varepsilon) & = & s(x_3) \oplus \text{'99'} \end{array}$$

We compute

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

We get

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

With five faults {'1E', 'E1', 'B3', '16', '9E'}, we obtain a correct and single value of $K_{10}[0]$, $K_{10}[7]$, $K_{10}[10]$, $K_{10}[13]$.

A Back to initial key with the last subkey

See [1] for additional informations about w and RotWord, Rcon and SubWord functions.

Let denote by $K_n[j]$ the j^{th} byte of the n^{th} roundkey and $w[i]$ as in [1]. We have

$$K_n = (w[N_k n], w[N_k n + 1], \cdots, w[N_k n + N_k - 1]).$$

We have the following relations (for $N_k = 4, 6$):
for $N_k \leqslant i < Nb * (Nr + 1), i \neq 0 \bmod N_k$,

$$\begin{aligned} w[i] &= w[i - N_k] \oplus w[i-1] \\ \text{i.e. } w[i - N_k] &= w[i] \oplus w[i-1] \end{aligned}$$

and for $i = 0 \bmod N_k$,

$$\begin{aligned} w[i] &= w[i - N_k] \oplus \text{SubWord}(\text{RotWord}(w[i-1])) \oplus \text{Rcon}[i/N_k] \\ \text{i.e. } w[i - N_k] &= w[i] \oplus \text{SubWord}(\text{RotWord}(w[i-1])) \oplus \text{Rcon}[i/N_k] \end{aligned}$$

Hence, we have
for $0 \leqslant i < Nb * (Nr + 1) - Nk, i \neq 0 \bmod N_k$,

$$w[i] = w[i + N_k] \oplus w[i + N_k - 1] \tag{3}$$

and for $i = 0 \bmod N_k$,

$$w[i] = w[i + N_k] \oplus \text{SubWord}(\text{RotWord}(w[i + N_k - 1])) \oplus \text{Rcon}[(i + N_k)/N_k] \tag{4}$$

With AES-256, you must add an Subword operation when $i \equiv 4 \bmod N_k$. So we can deduce previous key from endding subkey and step by step obtain K_0 with is the cipherkey.

Remark 1 *On AES-128, it is sufficient to know K_{10} to find the cipher key, but on AES-256, you must know K_{13} and K_{14}.*

```
RecoverKey(byte Finalkey[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
    word temp
    i = Nb * (Nr+1)-1
    j = Nk - 1
    while (j >= 0)
       w[i] = word(Finalkey[4*j], Finalkey[4*j+1],
                       Finalkey[4*j+2], Finalkey[4*j+3])
       i = i-1
       j = j-1
    end while
 {here, "i" must be equal to Nb * (Nr+1) - Nk - 1}
    while (i >= 0)
        temp = w[i+Nk-1]
        if (i mod Nk = 0)
            temp = SubWord(RotWord(temp)) xor Rcon[i/Nk+1]
        else if (Nk > 6 and i mod Nk = 4)
            temp = SubWord(temp)
        end if
        w[i] = w[i+Nk] xor temp
        i = i - 1
    end while
end
```

Figure 1: Pseudo Code for Key Recovery.

References

[1] National Institute of Standards and Technology, *FIPS PUB 197: Advanced Encryption Standard (AES)*, 2001. https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.197.pdf

[2] D. Boneh, R. A. DeMillo, R. J. Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*, EUROCRYPT'97. doi:https://doi.org/10.1007/3-540-69053-0_4

[3] E. Biham, A. Shamir, *Differential Fault Analysis of Secret Key Cryptosystems*, CRYPTO'97. doi:https://doi.org/10.1007/BFb0052259

[4] R. Anderson, M. G. Kuhn, *Tamper Resistance—A Cautionary Note*, 2nd USENIX Workshop on Electronic Commerce, 1996. https://www.usenix.org/publications/library/proceedings/ec96/full_papers/kuhn/