Security Lab – Introduction to Fault Injection and Differential Fault Analysis

This laboratory involves hands-on experimentation by making use of Differential Fault Analysis (DFA) techniques on an Advanced Encryption Standard (AES) implementation coded in C, running on an STM32 ARM-based microcontroller.

The main goal will be to introduce in practice the Electromagnetic Fault Injection (EMFI) on an STM32 - Nucleo Board running an embedded AES program. Then, to discover how it is possible to exploit faulty results for finding the secret key used inside the program.

This laboratory has two parts:

1- The practical step on an EMFI bench: You will discover how it is possible to inject an EMFI fault into the encryption process of an STM32 - Nucleo board.
2- The data processing step: You will exploit the faulty result to find the secret key used inside the embedded program.

Task 1: Preparing the NUCLEO board for an EMFI attack

1.1. You are already familiar with this board, used in the SCA Lab. We need to displace the trigger to perform an attack on the 8th or the 9th round AddRoundKey (ARK).

1.2. To trigger the oscilloscope, use the HAL library commands:

```
HAL_GPIO_WritePin(LED2_GPIO_PORT, TRIGGER_PIN_D8, GPIO_PIN_SET);
HAL_GPIO_WritePin(LED2_GPIO_PORT, TRIGGER_PIN_D8, GPIO_PIN_RESET);
```

In your opinion, where are the relevant positions of these two commands inside the code? Explain why in your report.

1.3. How long is the duration of the Rounds 8, 9 and 10 together?

1.4. When is the appropriate time to attack the 8th round ARK?

1.5. How many faulty bytes occur in the ciphertext if we inject a single-byte fault into the 8th round ARK?

Task 2: AES Execution on MATLAB

2.1. Open the AES_update folder and look at the files. To encrypt plaintext with a chosen key, use the command Cipher(key,In). Example:

```
 >> Cipher('00112233445566778899aabbccddeeff',
'000102030405060708090a0b0c0d0e0f')
```

Task 3: Differential fault Analysis for finding the secret key

```
$C \leftarrow M$
$C \leftarrow C \oplus K$
$RC = 1$
while $(RC < R_{max})$ do
    $C \leftarrow \mathtt{SB}(C)$
    $C \leftarrow \mathtt{SR}(C)$
    $C \leftarrow \mathtt{MC}(C)$
    $C \leftarrow C \oplus K_{RC}$
    $RC \leftarrow RC + 1$
end while
$C \leftarrow \mathtt{SB}(C)$
$C \leftarrow \mathtt{SR}(C)$
$C \leftarrow C \oplus R_{max}$
```

This algorithm refers to a software AES implementation.

In the AES-128 mode, the R_{max} value is equal to 10.

By an EMFI attack, at the end of the 8th round, the jump to the comparison instruction is skipped. The oscilloscope signals show that the encryption process is shortened one round (i.e. 9 rounds are executed instead of 10 rounds).

In your opinion, which round is suppressed?

During the experiments, we discovered that this attack can be **repeated** for several encryptions. For three different plaintexts, we performed the correct encryption. Then, we repeated the attack and obtained the corresponding faulty encryptions. Three plaintexts and their corresponding correct and faulty ciphertexts are reported here (and also in the file data.txt):

```
% Ma: 1st Plaintext, Ca: Corresponding correct ciphertext, Da:
Corresponding faulty ciphertext
Ma = '000102030405060708090a0b0c0d0e0f'
Ca = '50fe67cc996d32b6da0937e99bafec60'
Da = 'df48310c723bb363f6de952645e7aec3'

% Mb: 2nd Plaintext, Cb: Corresponding correct ciphertext, Db:
Corresponding faulty ciphertext
Mb = '3243f6a8885a308d313198a2e0370734'
Cb = '3925841d02dc09fbdc118597196a0b32'
Db = '577abf0e3ba2c205acaf4610218fcf33'

% Mc: 3rd Plaintext, Cc: Corresponding correct ciphertext, Dc:
Corresponding faulty ciphertext
Mc = 'ffeeddccbbaa99887766554433221100'
Cc = '2f49671c7ab81b2f435d9b650e35b8c1'
Dc = '378da346dc567473c462386312d232ce'
```

We know that:

$$C^a = \mathtt{SR} \circ \mathtt{SB}[\mathtt{MC} \circ \mathtt{SR} \circ \mathtt{SB}(M_8^a) \oplus K_9] \oplus K_{10}$$

$$D^a = \mathtt{SR} \circ \mathtt{SB}(M_8^a) \oplus K_{10}$$

Therefore, by combining the two previous equations, we get:

$$\mathtt{SB}^{-1} \circ \mathtt{SR}^{-1}(C^a \oplus K_{10}) = \mathtt{MC}[D^a \oplus K_{10}] \oplus K_9$$

Then, by expressing the relation between C_b and D_b similarly and by xoring it to the previous equation, we obtain a new equation without K_9:

$$\mathtt{SB}^{-1} \circ \mathtt{SR}^{-1}(C^a \oplus K_{10}) \oplus \mathtt{SB}^{-1} \circ \mathtt{SR}^{-1}(C^b \oplus K_{10}) = \mathtt{MC}(D^a \oplus D^b)$$

C_a, C_b, D_a and D_b are known values.

3.2. How many hypotheses exist for one byte of the K_{10} **without** using the above equation?

3.3. Now, write your own MATLAB function to find the correct hypothesis for each byte of K_{10}, by only using the last equation and without using the plaintexts. Include your function in your report.

3.4. Do you find the correct hypothesis for each byte of K_{10}? If not, what is the number of hypotheses **with** using the above equation?

3.5. How the plaintexts can be useful here?

3.6. How is it possible to enter the third corresponding correct and faulty ciphertexts? May you write a new equation similar to the above equation using C_a, C_c, D_a and D_c?

3.7. If you enter D_a and D_b inside your MATLAB calculations, how can the result for finding the correct hypotheses be improved?

Task 4: Countermeasures

4.1. Do you have any idea for a countermeasure?