Title of the Experiment: Fault Injection Attacks

Group Members' Names: Sujan Kumar Saha, Pankaj Bhowmik

Date: 11/12/2019

Abstract

The objective of the experiment is to analyze Fault Injection attacks in order to compromise the security of the system. In this experiment, we aim to perform a systematic approach that makes the advanced encryption standard (AES) vulnerable. AES is considered one of the strongest algorithms in cryptographic applications. The hardware application of AES cannot be considered as secured. The encryption key can be obtained by applying some glitches in the system and then perform some brute force analysis. If the encryption key can be extracted, then the ciphertext becomes vulnerable. However, there are several glitching methods, for instance, voltage glitching, clock glitching, EM disturbances and Flash glitching. However, the scope of the experiment is limited to the first two glitches.

Experiment Details

Goals: The goal of this experiment is to understand the fault injection methods and implement the voltage glitch and clock glitch methods to extract the key.

Experimental Setup: We used the MAX10 FPGA on Hardware Hacking (HaHa) board, USB-blaster, Quartus Prime Software, Analog Discovery, and Matlab.

Experiment Steps:

1. We run the AES algorithm in Quartus. With the help of given codes, we design the top module. To observe the ciphertext, to instantiate a ram where each word has 128 bits.
2. We power up the HAHA board and then download the bitstream into the MAX10 FPGA.
3. We checked the behavioral functionalities by applying the given states and keys to the top module and then observed them in the in system memory content editor.
4. We are in group 5, and extract the ciphertext for the assigned state and key.
5. The sof file for group 5 is then loaded to the FPGA to extract the ciphertext. The plain text and the key is unknown.
6. The voltage glitch is applied to the system and we collect the ciphertext for all 128 possible patterns. The ciphers are then exported into hex files and then accumulate 128 values in a text file. These values represent faulty cipher values.
7. We implemented the Matlab code of DPA analysis and run the code to find the key.

Results and Analysis:

Part I:

1. In the first part of this experiment, we are assigned to implement the AES, where different sub-modules were given. We designed the top module and instantiate a ram. Since the output of the model is 128 bit, the word length of the ram in 128. Besides, the AES takes input from a clock and runs at 50Mhz. We assign pin_88 on the haha board as the system clock. There are some states and keys are given. The implemented AES successfully generates the correct ciphertext according to the given result. The snaps of the in system memory content editor is given below.

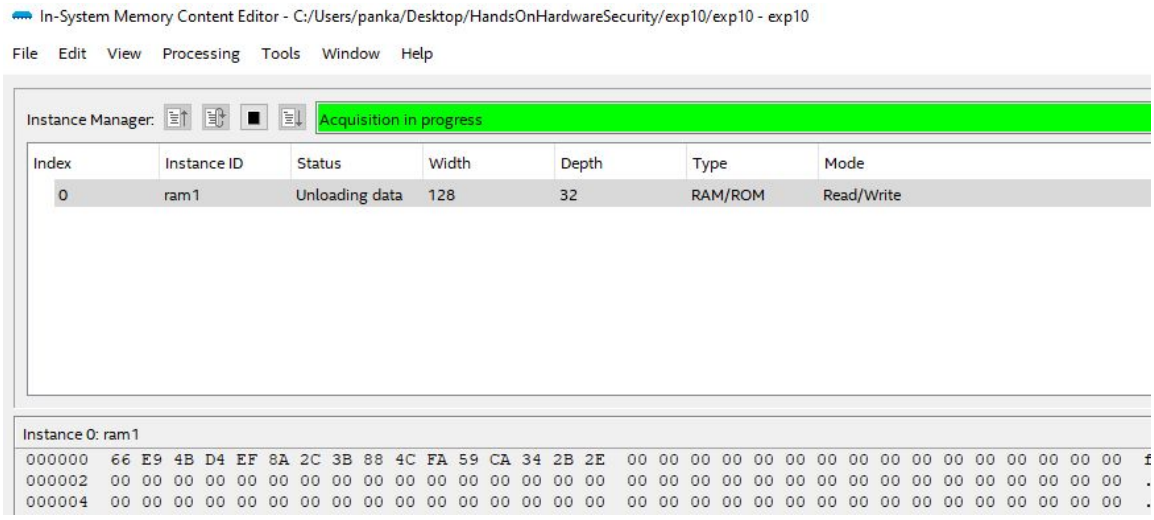

Fig1: Ciphertext : 66e94bd4ef8a2c3b884cfa59ca342b2e

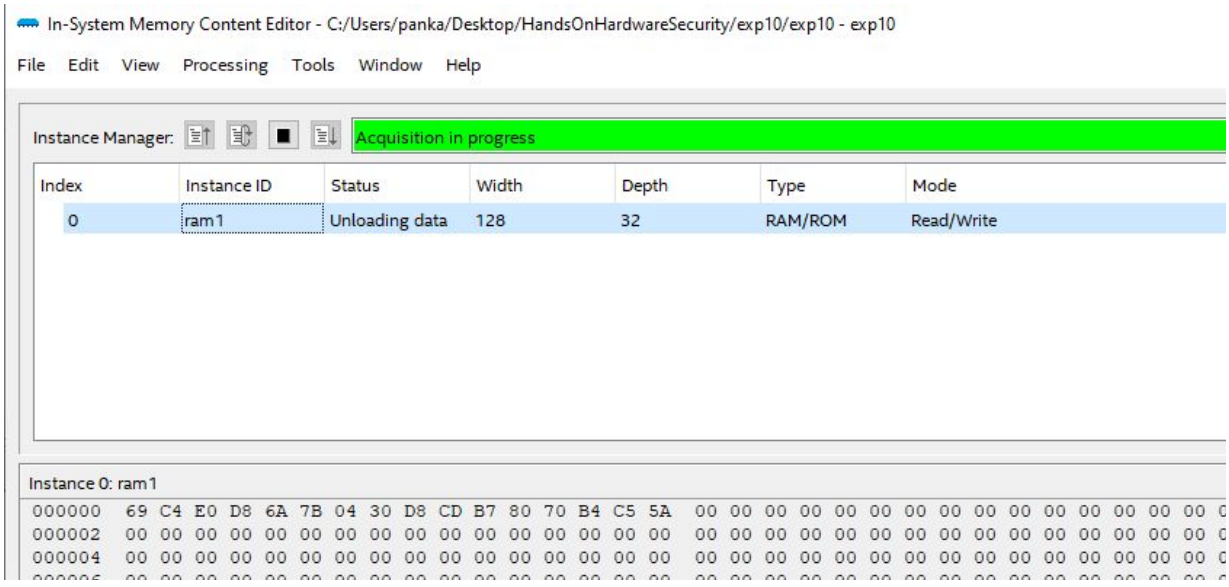

Fig2: Ciphertext : 69c4e0d86a7b0430d8cdb78070b4c55a

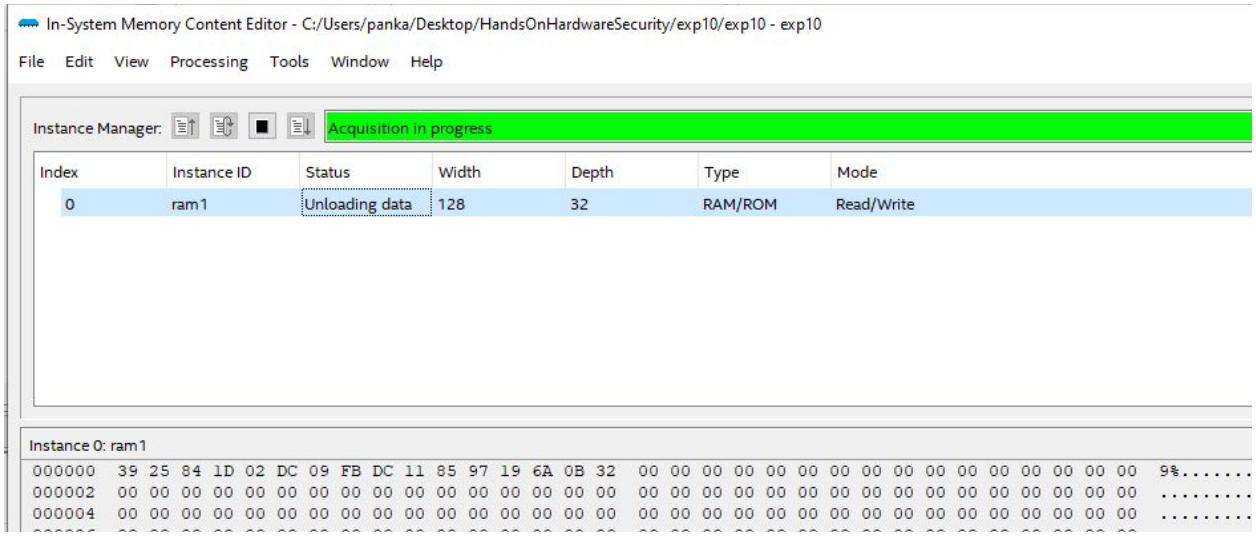

Fig3: Ciphertext : 3925841d02dc09fbdc118597196a0b32

2. After that, we put 3925841d02dc09fbdc118597196a0b32 as state value and 5dba9b93c355a576600a3e1ede5c396f as the key value. These values are assigned to group 5. And we get the cipher as **4E47BD5F9CB969F7B6B2AA3E02879FF2**. This snap of the in system memory content editor is given below.

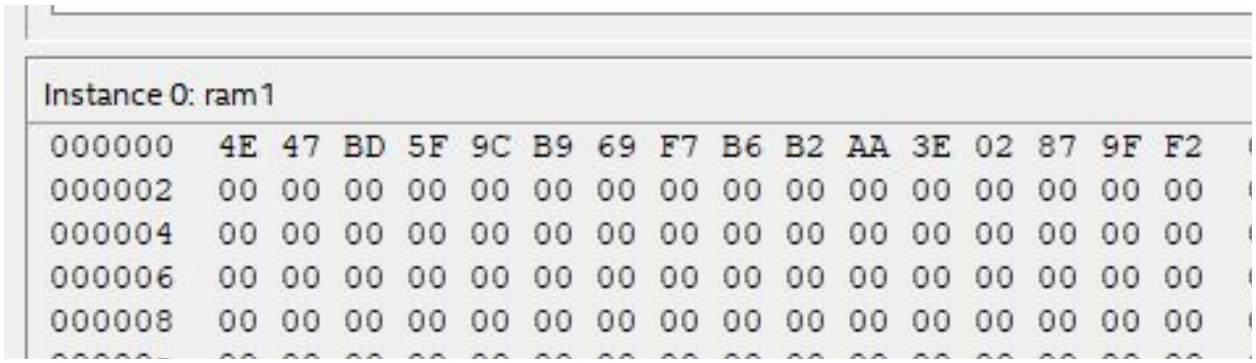

Fig3: Ciphertext : 4e47bd5f9cb969f7b6b2aa3e02879ff2

Part II:

1. Our group's C is 'F3EFE64049AC3AAB5030757154F79EF2'.
2. We choose to inject a fault in the last round because this is the weakest round of AES algorithm. The last round does not have Mix column step which is very hard to break. The difference between last round and other rounds is that Mix column step which has multiplication with polynomial modulo function.
3. The byte position change of the *shiftrow* operation is given below. The byte at i-th position goes to j-th position after *shiftrow* operation.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
j	1	6	11	16	5	10	15	4	9	14	3	8	13	2	7	12

4. We have implemented the Matlab code to find Mi9, M9 and K10. The code and Dj value files are added with the report.
5. Our group's M9 value is 'F7CDFEC0FBFDCDE0DFF8E3CCE8D2FCF9'.
6. Our group's K10 value is '9BBBF7D946ED8A11CE85CE90CF4A23B9'.
7. Unfortunately, we did not have enough time to calculate the K value.

Summary and discussion:

The experiment is about fault injection attack where we have learned how to find the key of AES encryption algorithm by injecting a fault at the last round. We faced difficulty to find the K because of the complex reverse operation of key expanding part. Other than that we were able to extract upto K10. But, overall the experiment was a good way to learn fault injection attack.