VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

M-Tech Integrated Software Engineering

Information and System Security- SWE3002

Review – 3

Differential Fault Analysis on AES

Team Members:

S.No	Name	Reg.No
1	16MIS0114	R SAI KIRAN
2	16MIS0133	M SAI DEEPU
3	16MIS0254	PRASANNA KUMAR

Submitted To:

Prof. Srinivas Koppu

Slot: E1+TE1

School Of Information Technology and Engineering

April-2019

Abstract:

Hardware designers put a great effort to invest a significant design while implementing cryptographic algorithms onto an embedded device to match the demands of algorithm. When it comes to design which are present in a potential hostile environment another challenge arises that is the design should be resistant to attacks based on physical properties of implementation. Therefore, this makes the hardware designer to worry a little when it comes to designing. This project explains about the fault attacks and countermeasures to help the designer to protect the design against this type of implementation attacks. We analyse the different types of fault attacks and classify the existing countermeasure for the respective fault attack. We also discuss about the effectiveness and efficiency of the counter measure. The result of the project is to provide a set of countermeasures, which provide sufficient security level to meet the limits of embedded devices.

Keywords: fault attack, implementation attacks, countermeasures

Introduction:

Differential Fault Analysis is a type of side channel attack in the field of cryptography. Fault attacks and passive side channel attacks like power or Electromagnet analysis are powerful attacks on implementations of cryptographic algorithms. Fault attacks consist in using hardware malfunction to infer secrets from the target's faulty behaviour or outputs. These active attacks can be performed in different physical manners. Differential fault analysis exploits differential information between correct and faulty cipher texts to retrieve the secret key. Because of symmetric cryptography, the regular way to deal with securing an execution against these faults is to use masking and redundancy mechanism. Therefore, we need to provide countermeasures for all types of fault attacks.

Fault attacks come in many varieties. Some fault attacks, for example, the differential fault attack misuse the spread of a fault from the point of injection to the output, and requires sets of right and faulty cipher texts. We present a generic variety of countermeasure against fault attacks known as fault space

transformation based countermeasures. Any counter measure against fault attack has two major phases – fault detection and fault nullification. The first step is uses some form of spatiotemporal redundancy to detect the occurrence of fault, and the second phase attempts to either decrease or randomize the effect of fault. Some well-known countermeasure strategies in the present literature include the use of redundancy like temporal, spatial, information and hybrid.

Contributions:

The major contributions of the paper are:

- This paper provides the information about the types of faults involved in AES and its countermeasures. It describes a smart way of counter attacking the faults.

- This paper shows the different types of attacks on AES, which changes the values of round counters.

- This paper introduces a generalized counter measures framework against DFA and DFIA using the concept of fault space transformation.

- The paper shows the requirement of such a countermeasure by showing actual DFIA attacks using practically achievable biased fault models on the trivial temporal and spatial redundancy countermeasures, connected to AES-128 implementations.

- In this paper, we demonstrate that SIFA attacks are considerably more dominant than anticipated. Our main contribution is to show that SIFA is not just independent of the degree of redundancy but also independent of the degree of masking.

Differential Fault Analysis:

In DFA, they inject a random fault with certain known spatio-temporal characteristics and analyses faulty and fault free cipher texts to find the secret key. It is powerful that it can recover the complete 128-bit key of AES with just single fault injection.

Attacks on the round counter value:

This attack scenario changes the round counter during AES execution Therefore it changes the index of the current executing round. Any change in the RC value often leads to a change in the total number of executed rounds, by adding, suppressing or even repetitively executing several rounds:

DIFFERENT CHANGES IN ROUND COUNTER VALUES (Proposed Method)

Experiment1:

If $RC \oplus e < RC \Rightarrow$ Round addition or repetitive execution of several rounds.

For instance: if RC=7 and e =2 then $RC \oplus e=5$ and the AES execution will be:

R0. . . R5-R6-R5-R6-R7. . . R10

The rounds 5 and 6 will be executed twice and the total number of executed rounds will be incremented to 12.

Experiment 2:

If $RC \oplus e > RC$ with $RC < Rmax-1 \Rightarrow$ Round reduction.

For example: if RC=4 and e =2 then $RC \oplus e=6$ and the faulty AES execution will be: R0. . . R3-R6. . . R10

Therefore, the rounds 4 and 5 will be skipped and the total number of executed rounds will be reduced to 8.

Experiment3:

If $RC \oplus e > RC$ with $RC = Rmax - 1 \Rightarrow$ Round alteration: no change in the total number of rounds, but effects on AddRoundKey of the final round and maybe the penultimate round.

For instance: if RC=9 and e =2. then $RC \oplus e=11$ and AES execution will be:

R0. . . R8-Rm=11-Rf=12

Therefore, the total number of executed rounds will remain 10, but the penultimate round and the final round will use invalid round keys values (K11 and K12) during their ARK transformations.

Countermeasures against Fault Attacks:

Deployment of novel and improved fault attacks drives the evolution of countermeasures. Since there exist no generic countermeasures, which can prevent all attacks, the combination of different techniques is required to achieve a sufficient security level.

There are two major principles for protection of a cryptographic device against fault attacks

- Hardware countermeasures
- Design driven countermeasures

HARDWARE COUNTERMEASURES:

A prominent example of such countermeasures are passive and active shields. Passive shields are metal layers that cover a part of or the entire chip, thus preventing an optical fault injection or probing attacks. An active shield consists of a wire mesh that runs signals over the chip surface and detects any interruption on a wire. Furthermore, a chip can be equipped with light sensors and anomalous frequency detectors which detect optical fault injection and glitches in the clock signal.The main drawback of the hardware countermeasures is their cost. In addition, since such protection aims at preventing a specific method of fault injection, it does not provide long term protection, as novel fault injection methods are frequently developed.

DESIGN DRIVEN COUNTERMEASURES

There are two main principles used to construct design driven countermeasures:

(1) Employing redundancy to check whether the computation was tampered with, i.e. incorporate a fault check which detects and reports a fault,

(2) Design the implementation to be characteristic way to attacks.

Although the second method leads to a more efficient protection, it prevents only a limited set of attacks. Therefore, it is necessary to combine it with a fault detection mechanism in order to provide complete protection. Further, a fault can be detected in different parts of a processor:

1. input part

2. processing part

3. program flow.

Finally, an actual fault detection mechanism can be implemented in different ways.

Differential Fault Intensity Analysis (DFIA):

It is a group of fault attacks that combine principles of side channel analysis techniques. Such attacks require only faulty cipher texts and exploit the fact that the key dependent faulty state value has a strongly biased distribution for the correct key hypothesis.

Fault Injection Setup:

The fault injection set up consists of an FPGA (Spartan3A XC3S400A), a PC and an external arbitrary function generator (Tektronix AFG3252). The FPGA has a DUT (Device under Test) block, which is a time or hardware redundant AES implementation. Faults were injected using clock glitches and the fault intensity was controlled by increasing/decreasing the glitch frequency. The system had two clock signals - clkslow and clkfast, both of which were derived from an external clock signal clkext via a Xilinx Digital Clock Manager (DCM) module. The clkslow signal was used for fault-free operation of the DUT, while the clkfast signal was used to create the glitches for fault injection.

Applying Fault Space Transformation to AES – 128: (Counter measure)

We apply our proposed fault space transformation based countermeasure technique to protect AES-128 against biased fault attacks. For the transformation function W, we propose using the Rjindael MixColumns matrix for its ease of implementation.

Statistical Fault Attacks:

In SFA, we only require a collection of faulty cipher texts encrypted with the same key. Hence, SFA works with random and unknown plaintexts. Assuming that intermediate variables get uniformly distributed towards the last rounds for secure cryptographic primitives like AES, an attacker has to be able to induce faults which change the distribution of some intermediate values to be non-uniform.

Unlike most traditional fault attacks, SFA requires a slightly different fault model. Assuming that intermediate variables are uniformly distributed towards the last rounds for secure cryptographic primitives like AES, an attacker has to be able to induce faults, which change the distribution of some intermediate values to be non-uniform.

Attack procedure:

We will focus only on the attack that targets the 9th round of AES. When changing the distribution of one byte of AES before the last Mix Columns, they showed that with these fault models, 4 bytes of the last round key could be recovered with high probability using the Squared Euclidean Imbalance (SEI).

Statistical Ineffective Fault Attacks (SIFA) allow an attacker to circumvent many popular fault countermeasures. As the name suggests, and in contrast to SFA, SIFA solely relies on exploiting faulted encryptions where the induced faults are ineffective and the obtained cipher texts are hence always correct. The basic observation of SIFA is that induced faults can lead to a non-uniform distribution of intermediate values for the cases where the fault has been ineffective, which in turn can be exploited in a key-recovery attack. The data complexity of the attack then depends on the strength of the bias in the targeted

intermediate variable and the necessary number of faulted encryptions to obtain sufficiently many ineffective samples.
SIFA is applicable just as easily for more than two redundant computations, since only one computation needs to be faulted. In fact, it has been shown in that SIFA is not only applicable against redundancy countermeasures, but also against infective countermeasures. Although in the latter case more faulted encryptions are necessary, the presented attacks are still efficient. Consider an AES with simple temporal redundancy and an attacker that is able to fault a certain byte in round 9 such that the faulted value follows some non-uniform distribution, which is not known to the attacker. If the attacker is faulting only one of the redundant computations, the attacker will eventually observe correct ciphertexts where the induced fault was ineffective.

Counter measures:

Self-Destruct.

The most radical approach of destroying the device as soon as a fault is detected is a valid countermeasure against any fault attack. However, this technique has a few downsides and limitations, including false positives and additional effort to reliably destroy a circuit

Correction.

A different approach to make use of redundancy is to correct the effect of a fault, for instance using error-correcting codes or simple majority voting. However, correction-based countermeasures usually can be reduced to the detection-based case using additional faults. How hard this is and which requirements this might have on the precision of the fault crucially depends on the implementation of the countermeasure

Infection.

In the application of SIFA on an infective countermeasure has been demonstrated. The employed dummy rounds in this countermeasure increase the needed number of faulted encryptions until the key can be recovered.

However, when aiming to prevent SIFA, dummy rounds that do not infect the state in the case of a fault should provide even more protection. Hence, we explore this countermeasure next.

Hiding.

The goal of hiding countermeasures is to reduce the attacker's knowledge of what is currently computed, and thus effectively decrease his precision when placing the fault. Examples include adding dummy rounds randomly between the relevant rounds, or shuffling the order of execution, for example the order in which the 16 AES S-boxes per round are executed. In the following, we analyze the case of dummy AES rounds in more detail, and show that the noise introduced this way quadratically increases the necessary number of faulty encryptions for the analysis.

Fault Detection Mechanisms:

A fault detection is the basis for most of the existing countermeasures against fault attacks. They incorporate a fault detection mechanism which detects and reports an error, thus preventing an erroneous result to be observed by an attacker. In what follows, we describe how fault detection mechanisms are used to protect basic parts of a generic processor: input, processing, and the control part.

1) Protection of Input Parameters:

In some protocols, an adversary is allowed to supply parameters that are used for the computation later on. If an implementation misses to check whether these parameters are valid, the adversary can choose bogus inputs to attack the system. If the inputs are not supplied externally, the system parameters are typically stored in non-volatile memory and are transferred to RAM for the computations. Either these parameters can also be targeted for attack, as errors that occur in public or secret parameters can be exploited to reveal secret information. Since such errors can be injected even before traditional countermeasures that protect the integrity of an algorithm can take effect, the validity of input parameters must be explicitly checked before being used for computations. A common way of implementing such checks is by adding a Cyclic Redundancy Check (CRC) to each system parameter. After a parameter is read for a computation, its CRC is computed and compared to the one stored in non-volatile memory. Error detection probability of the CRC is proportional to the amount of added redundancy. However, increased redundancy introduces additional hardware overhead, since more memory is required for its storage. Moreover, a module that computes CRCs has to be implemented. Therefore, the amount of redundancy is usually a result of a trade-off between the desired security level and the available hardware resources.

2) Protection of the Processing Part:

The processing part of a device is harder to protect than the parameters, since static codes are not sufficient. A fault injected in a device during the processing time may enable leakage of sensitive information. Furthermore, there is a big variety of attacks like modifying registers, disturbing the computation of the ALU, or modifying the program flow to miss instructions. In order to prevent such attacks, a method called CED is introduced. Concurrent error detection is a common mechanism used to prevent fault-based cryptanalysis. It is based on introducing redundancy to check the correctness of a computation.

Conclusion:

In order to construct a fault-attack resistant embedded device, a hardware designer has to understand their threat in the first place. Moreover, we explained how an injected fault propagates from the physical layer to the protocol layer and highlighted the parts of a generic processor, which are sensitive to faults. Hence, we provided an orientation to hardware designers for determining which attacks are a threat for their specific designs. In this paper we described about the various faults and their counter measures.

References:

[1] Patranabis, S., Chakraborty, A., Mukhopadhyay, D., & Chakrabarti, P. P. (2017). Fault space transformation: a generic approach to counter differential fault analysis and differential fault intensity analysis on aes-like block ciphers. *IEEE Transactions on Information Forensics and Security*, *12*(5), 1092-1102.

[2] Dobraunig, C., Eichlseder, M., Gross, H., Mangard, S., Mendel, F., & Primas, R. (2018, December). Statistical ineffective fault attacks on masked AES with fault countermeasures. In *International Conference on the Theory and Application of Cryptology and Information Security* (pp. 315-342). Springer, Cham.

[3] Fahd, S., Afzal, M., Abbas, H., Iqbal, W., & Waheed, S. (2018). Correlation power analysis of modes of encryption in AES and its countermeasures. *Future Generation Computer Systems*, *83*, 496-509.

[4] van Woudenberg, J., Breunesse, C. B., Velegalati, R., Yalla, P., & Gonzalez, S. (2017, December). Differential Fault Analysis Using Symbolic Execution. In *Proceedings of the 7th Software Security, Protection, and Reverse Engineering/Software Security and Protection Workshop* (p. 4). ACM.

[5] Dutertre, J. M., Mirbaha, A. P., Naccache, D., Ribotta, A. L., Tria, A., & Vaschalde, T. (2012, June). Fault round modification analysis of the advanced encryption standard. In *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on* (pp. 140-145). IEEE.

[6] Bae, K., Moon, S., Choi, D., Choi, Y., Choi, D. S., & Ha, J. (2011, November). Differential fault analysis on AES by round reduction. In *Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on* (pp. 607-612). IEEE.

[7] Karaklajić, D., Schmidt, J. M., & Verbauwhede, I. (2013). Hardware designer's guide to fault attacks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *21*(12), 2295-2306.

[8] Mirbaha, A. P., Dutertre, J. M., & Tria, A. (2013, October). Differential analysis of Round-Reduced AES faulty ciphertexts. In *Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2013 IEEE International Symposium on* (pp. 204-211). IEEE.

[9] Patranabis, S., Chakraborty, A., Mukhopadhyay, D., & Chakrabarti, P. P. (2017). Fault space transformation: a generic approach to counter differential fault analysis and differential fault intensity analysis on aes-like block ciphers. *IEEE Transactions on Information Forensics and Security*, *12*(5), 1092-1102.