Exploring the Feasibility of Low Cost Fault Injection Attacks on Sub-Threshold Devices through an example of a 65nm AES implementation

Alessandro Barenghi[1], Cédric Hocquet[2], David Bol[2], François-Xavier Standaert[2], Francesco Regazzoni[2,3], and Israel Koren[4]

[1]DEI - Politecnico di Milano, Milano, Italy. barenghi@elet.polimi.it
[2]ICTEAM Institute, Université catholique de Louvain, Louvain-la-Neuve, Belgium.
{first_name.last_name}@uclouvain.be
[3]ALaRI - University of Lugano, Lugano, Switzerland. regazzoni@alari.ch
[4]University of Massachusetts, Amherst, MA, USA. koren@ecs.umass.edu

Abstract. The continuous scaling of VLSI technology and the aggressive use of low power strategies (such as subthreshold voltage) make it possible to implement standard cryptographic primitives within the very limited circuit and power budget of RFID devices. On the other hand, such cryptographic implementations raise concerns regarding their vulnerability to both active and passive side channel attacks. In particular, when focusing on RFID targeted designs, it is important to evaluate their resistance to low cost physical attacks.

A common low cost fault injection attack is the one which is induced by insufficient supply voltage of the chip with the goal of causing setup time violations. This kind of fault attack relies on the possibility of gracefully degrading the performance of the chip. It is however, unclear whether this kind of low cost attack is feasible in the case of low voltage design since a reduction of the voltage may result in a catastrophic failure of the device rather than an isolated setup violation. Furthermore, the effect that process variations may have on the fault model used by the attacker and consequently the success probability of the attack, are unknown.

In this paper, we investigate these issues by evaluating the resistance to low cost fault injection attacks of chips implementing the AES cipher that were manufactured using a 65nm low power library and operate at subthreshold voltage. We show that it is possible to successfully breach the security of a custom implementation of the AES cipher. Our experiments have taken into account the expected process variations through testing of multiple samples of the chip. To the best of our knowledge, this work is the first attempt to explore the resistance against low cost fault injection attacks on devices that operate at subthreshold voltage and are very susceptible to process variations.

1 Introduction

Radio Frequency Identification (RFID) devices are nowadays used in a wide range of applications, such as health care, supply chain management, or pet identification [6]. Such a pervasive diffusion raises concerns regarding the privacy of the users as the RFID tags often store sensitive information. RFID devices have a very strict power and area budget, and as a result, incorporating the necessary support needed to guarantee privacy is a challenging task since the security primitives are often too costly in terms of area and power.

A particularly appealing solution to meet the above challenges is to exploit nanometer CMOS technologies, adopt known aggressive power saving techniques, and operate the device at a subthreshold voltage (with a typical supply voltage of 0.3V to 0.5V).

Using nanometer CMOS technologies, it is possible to implement standard cryptographic algorithms within the restricted available area. It is likely that future generations of RFID tags will be able to afford the cost of being manufactured in a more updated technology [3].

By using low power cell libraries and operating the device at a subthreshold voltage, it is possible to significantly reduce the power consumption but at a price of a considerably lower speed. However, this is acceptable since speed for RFIDs is not as crucial as for other applications.

A key concern for every secure cryptographic implementation is vulnerability to both active and passive side channel attacks. In the case of RFID designs, it is particularly important to evaluate the resistance of new implementations to low cost physical attacks such as fault injection attacks carried out by simply decreasing the supply voltage.

Although aggressive design techniques enable the use of standard ciphers and achieve very low power implementations, the security of the resulting circuit against such side channel attacks still remains unexplored. Implementing RFIDs in nanometer technology and operating them at subthreshold voltage raises two issues that do not have to be dealt with when current off-the-shelf components are used. First, such implementations may experience functional failures if the Vdd is reduced below the reference supply voltage [3]. Thus, it is not clear whether it is practically possible to reduce the voltage in order to generate only timing faults (violations of the flip-flops' setup time) which are typically injected to mount low cost fault attacks. Second, nanometer CMOS technologies are prone to process variations. As a result, almost every chip will have its own unique timing. Thus, it is unclear whether fault injection attacks can be carried out in a systematic way.

In this paper, we answer these questions by performing a practical evaluation of the susceptibility of a subthreshold voltage implementation of AES (designed to be incorporated in an RFID), to low cost fault injection attacks. The considered design has a data path of 8 bit and is implemented using a 65nm low power library. In our experiments we slowly lowered the supply voltage to evaluate the susceptibility of the chip and to quantify the required precision of the power supply generator. Then, we conducted a similar set of experiments on 5 dies implementing the same functionality, to explore the effects of process variations on the susceptibility to fault attacks. To the best of our knowledge, this work is the first one that focuses on a practical evaluation of the resistance against fault injection attacks on subthreshold low power circuits.

The remainder of the paper is organized as follows. In Section 2 we briefly describe the AES algorithm and present several previously proposed fault attacks on AES. We then describe the architecture of our AES design in Section 3. Section 4 describes the attack technique chosen for our evaluation. Finally, the measurement setup and the results of our experiments are reported in Section 5.

2 Background

To practically evaluate the security level provided by a given implementation of a cipher, it is necessary to consider a number of attacks that an attacker can mount when granted physical access to the device. These attacks, commonly known as side channel attacks, rely on either measuring circuit parameters during the regular functioning of the device (e.g., power consumption, EM emissions) or actively perturbing the computation. This paper will focus on the second type of side channel attacks, the so-called fault injection attacks. These attacks rely on inducing non catastrophic faults into the computation (in order to obtain a faulty result), and analyzing the difference between the correct and wrong outputs. Since the induced faults only affect a part of the computation it is possible for the attacker to use the difference between the outputs to infer the secret key.

2.1 The AES Cipher

The cipher considered in this work is the Advanced Encryption Standard [14], due to its wide adoption and the subtantial cryptanalytical scrutiny it has undergone in the last 12 years. The selected variants of the Rijndael [5] algorithm as the AES standard support a plaintext block size of 128 bits and three key sizes of 128, 192 and 256 bits.

The AES cipher is based on the iteration of a round function composed of four primitives :SUBBYTES , SHIFTROWS , MIXCOLUMNS and ADDROUNDKEY . The number of times the round function is iterated, N_r, is 10, 12 or 14 times depending on the length of the key employed. The only exceptions to the repetition of the four primitives is the fact that the last round of the encryption is missing the MIXCOLUMNS primitive. Moreover, an extra ADDROUNDKEY is performed before the first round as a pre-whitening of the input state.

The inner state of the AES cipher after each round r, denoted by S_r, can be represented as a 4×4 matrix, where each element is 8 bit wide. We denote the n-th byte, counting from left to right, from top to bottom as S_n^r

Each primitive of the AES cipher contributes to either adding confusion or diffusion effects to the cipher, or to add a dependency on the value of the key. The SUBBYTES primitive is a non linear mapping over $\mathbb{Z}_{2^8}$ that introduces a non linear confusion effect. This mapping is applied to a single byte at a time, S_n^r and can be implemented either as a lookup table or computed on the fly. The SHIFTROWS primitive provides a row-wise diffusion effect to the inner state of the AES cipher. It rotates the four rows of the state S_r by 0,1,2 or 3 byte positions, respectively, while the values of the rotated bytes remain unaltered. The MIXCOLUMNS primitive provides column wise diffusion of the cipher state by considering the column as a vector of values over $\mathbb{Z}_{2^8}$, and multiplying the vector by a constant matrix. This operation linearly combines in an invertible way the contents of the four bytes of a column. The last operation, the ADDROUNDKEY primitive, combines the state of the AES cipher with a 4×4 key matrix through bitwise exclusive or.

Since the ADDROUNDKEY primitive is repeated $N_r + 1$ times, there is a need to expand the initial key provided by the user into $N_r + 1$ round keys through a key expansion routine. The AES key expansion routine combines the initial key through bitwise xor additions and application of the SUBBYTES primitive. The key schedule process is non-destructive, i.e., all the operations performed are bijective. As a result, if a person is in possession of 4,6 or 8 contiguous 32 bit words of the key schedule, he is able to reconstruct the full 128, 192 or 256 bit secret key.

2.2 Related Works

A number of fault injection attacks on the AES cipher have been reported. Although some of them were not experimentally validated at the time they were presented [4, 9, 11, 13], several other attacks were successfully mounted on real world implementations. For example, in [10] the authors were able to mount a successful attack by causing temporary brown outs and glitches on the supply line of an 8 bit microcontroller. In [18], Schmidt et al. attacked implementation of AES by blanking selectively the memory where the SBoxes are held. In [15], Peacham et al. describe a successful attack mounted using laser induced fault injection, on an AES implementation in a commercial grade smart card, that did not include any countermeasures against fault attacks. Another technique which has proven effective in inducing controlled fault is by causing setup time violations by lowering the supply voltage below the level the circuit was designed for. In [19], Selmane et al. report the effects of attacking a commercial grade ASIC

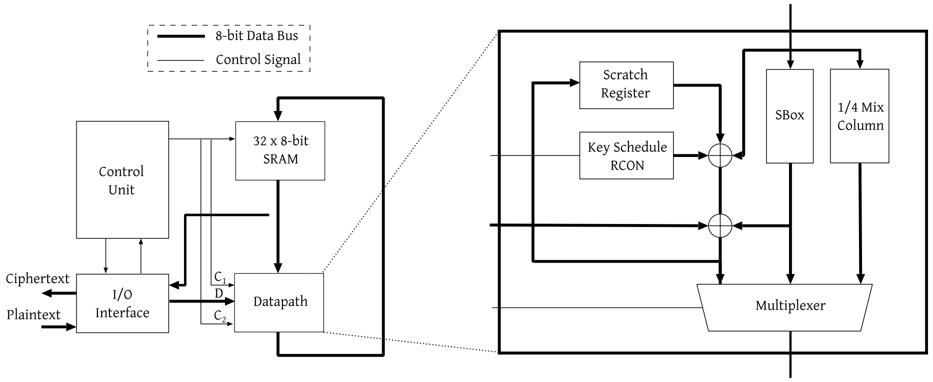

Fig. 1. Block diagram of the AES module proposed by Feldhofer *et al.* [8]

implementation of AES in a smart card, using this fault induction technique, while in [2] the authors successfully applied the technique to a full ARM9 core running a software implementation of AES.

3 Target Architecture

In this section we describe the architecture of the low area - low power AES design that we have used in our experiments. Since our objective is to evaluate the effectiveness of low cost fault injection attacks mounted on RFID devices, the base architecture must satisfy strict area and power requirements.

An AES architecture that is suitable for our purposes is the one proposed by Feldhofer *et al.* [8] that has an 8-bit datapath and supports only a single key size of 128 bits. The block diagram of the considered design is shown in Figure 1. The selected design includes three components: a module for computing the non linear transformation (S-box) that is used by the *SubBytes* operation and by the key expansion routine, a module for computing one quarter of the *MixColumn* operation per clock cycle, and a module to implement the round key addition. The *ShiftRows* operation is performed by accessing the register in an appropriate way.

The AES design which we have implemented is similar to the one described above except for the S-Box. A lightweight S-box implementation was proposed by Satoh *et al.* [17]. It requires to transform the input data into the composite Galois field $\mathbb{Z}(((2^2)^2)^2)$, invert it there efficiently and finally transform it back to $\mathbb{Z}(2^8)$. Such a design, which already resulted in a low gate count, was further optimized by Mentes *et al.* in [12]. We have therefore, selected this small footprint design for implementing the non linear transformation in AES.

The HDL code of the described AES design was synthesized for a target clock frequency of 100 KHz using *Synopsys Design Compiler*. The target library was the ST Microelectronics 65nm LP CMOS technology with seven interconnect metal layers. Considering the fact that the device would operate in the subthreshold regime, we manually removed from the target library the cells which may not operate correctly, i.e., the ones with the longest transistor stack, such as NAND3 or NOR3. We set the timing condition to the worst-case of SS process corner (slow NMOS and slow PMOS transistors), low temperature (-15°C), and operating supply voltage of 0.4V to achieve the desired 100 KHz clock frequency. The optimized design was then placed and routed using *Cadence SoC Encounter* and manufactured. All the dies were encapsulated in a 44 pin Ceramic Quad Flat Package (CQFP) and were tested to verify the correct execution of encryption and decryption. All the tested dies were found to operate correctly at a frequency of 1.3MHz with a power supply of 0.45V and 400KHz at 0.4V. However, by relaxing the clock frequency, it is possible to correctly operate the AES circuit at the voltage of 0.25V, which is the functional limit of the design.

4 Chosen Attack Methodology

In this section we present the attack methodology we used in this work. Dusart et al. [7] have claimed that it is possible to successfully retrieve the whole secret key of an AES-128 cipher, through the injection of byte-wide faults during the regular functioning of the cipher. The attack proposed relies on the injec-

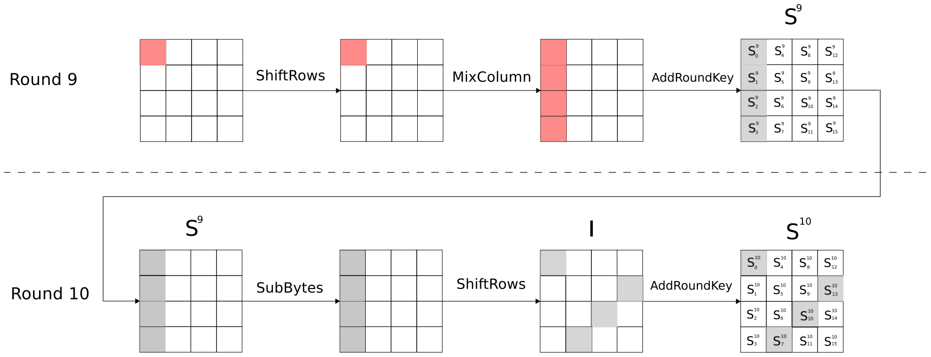
Round 9
ShiftRows
MixColumn
AddRoundKey
S^9
S_0^9 S_4^9 S_8^9 S_{12}^9
S_1^9 S_5^9 S_9^9 S_{13}^9
S_2^9 S_6^9 S_{10}^9 S_{14}^9
S_3^9 S_7^9 S_{11}^9 S_{15}^9
Round 10
S^9
SubBytes
ShiftRows
I
AddRoundKey
S^{10}
S_0^{10} S_4^{10} S_8^{10} S_{12}^{10}
S_1^{10} S_5^{10} S_9^{10} S_{13}^{10}
S_2^{10} S_6^{10} S_{10}^{10} S_{14}^{10}
S_3^{10} S_7^{10} S_{11}^{10} S_{15}^{10}

Fig. 2. Effects of the propagation of a single fault injected between the MixColumns operations of the eight and ninth round

tion of a single byte fault between the MIXCOLUMNS operation of the eighth round and the MIXCOLUMNS of the ninth round, as depicted in Figure 2. Due to the lack of the MIXCOLUMNS operation during the tenth round, the effect of the fault is spread only over 4 of the 16 bytes of the state. Since the key addition is performed byte-wise, the values of these 4 bytes are influenced only by 4 bytes of the last round key. Exploiting this fact and assuming that the injected fault has corrupted only one byte, the attacker may proceed to recover the 4 bytes of the key starting from a correct and faulty ciphertext. The attacker makes an hypothesis on the unknown part of the key and proceeds to invert the effect of the last ADDROUNDKEY on the part of the cipher that was affected by the fault (greyed out in the figure) obtaining 4 values belonging to the state marked as I in Figure 2. This operation is performed on both the erroneous and the correct values of the ciphertext, yielding 2 groups of 4 byte values. Subsequently, the attacker proceeds to invert the effect of both the SHIFTROWS and SUBBYTES primitives, since their effect is fully known, obtaining successfully a faulty and a correct hypothetical values for four bytes of the state S^9, denoted respectively $\widetilde{w} = \{\widetilde{S^9_0}, \widetilde{S^9_1}, \widetilde{S^9_2}, \widetilde{S^9_3}\}$ and $w = \{S^9_0, S^9_1, S^9_2, S^9_3\}$. Bypassing the effect of the ADDROUNDKEY operation, to further roll back the cipher, the attacker computes the exclusive or of $\widetilde{w}$ and w. Doing so, the effect of the ADDROUNDKEY function is cancelled since in computing $\delta = w \oplus \widetilde{w}$ the key values are added twice. This allows the attacker to effectively compute the difference between the correct and the erroneous state of the cipher right before the MIXCOLUMNS operation. Since the MIXCOLUMNS operation is linear with respect to the exclusive or, it is possible to map the four byte difference δ into the difference before the operation simply through multiplying the value by the inverse of the matrix employed in the regular MIXCOLUMNS . At this point, the attacker may check if the obtained difference is actually composed of a single byte, as the fault model required by the attack mandates, or not. Depending on whether the difference matches the fault model or not, the attacker can decide if the key hypothesis made at the beginning of this rollback procedure is a valid one or not.

The attacker iterates the same difference analysis procedure for all the possible 2^{32} values of the four unknown bytes of the key and stores only the ones which actually produce a single byte difference before the last MIXCOLUMNS operation when elaborated through the aforementioned procedure. A single sweep of this procedure yields roughly a thousand valid candidates for the 32-bit wide key slice, and may be repeated if more than one faulty ciphertext caused by a single byte fault is available to the attacker. With a second sweep of the procedure the number of key candidates is reduced to one with a reasonably high probability [16].

Since it is possible for the attacker to discern, looking at which bytes are affected by the faults, which slice of the key is the one under consideration, it is possible to reconstruct the whole last round key with 4 faults and a brute force effort of $1000^4 \sim 2^{40}$ AES encryptions, which takes about a couple of minutes on a modern desktop computer, or with 8 faults and no brute force effort at all.

Further reduction of the number of faults needed for the attack allows to employ a single fault happening between the MIXCOLUMNS operation of the seventh and the MIXCOLUMNS of the eight round as four single-byte faults happening simultaneously on each column of the state before the last MIXCOLUMNS is executed, thanks to the diffusing effect of the SHIFTROWS operation of the ninth round. This way, it is possible to retrieve the whole last round key of the cipher with a single fault and a modest brute force effort or two faults and no brute force effort. The main drawback of this method is that it is not possible to determine whether a faulty ciphertext has been originated by a fault complying with the required timing hypothesis, since the whole ciphertext value is altered. Nonetheless, if a fault which does not match the fault hypothesis is employed in the key recovery procedure, the number of valid key candidates drops to zero during the first iteration of the procedure, thus allowing the attacker to be aware of the issue.

After performing the aforementioned procedure, the attacker has all the bits of the last round key and is thus able to reconstruct the whole original key, in case a key size of 128 bit is employed. If larger key sizes are used, it is necessary to recover more key material in order to successfully break the cipher. An extension of the aforementioned attacks is reported in [2], and needs twice the number of faults in order to recover the whole AES-256 key. The fault model assumed is the same, but the attacker is required to inject faults also a round before where the aforementioned attack takes place.

5 Experimental Results

This section presents the measurement setup used and the experiments conducted in order to profile the behavior of the low power AES implementation and investigate the feasibility of low cost fault injection attacks based on voltage throttling.

5.1 Measurement Setup

The packaged chip was mounted on a suitable socket and a dedicated PCB was built. The chip under test was connected to a Keithley K236 power supply, which is sufficiently precise to allow reducing the supply voltage by as little as 0.1mV, as was needed for our purposes. The power supply was connected to the power pin of the chip under test. All the tested AES circuits were clocked at a frequency of 1.3MHz by means of an external clock generator and each encryption required 1100 clock cycles (less than 1ms). Note that these characteristics are within the typical range of RFID systems.

To carry out the experiments we connected the inputs/outputs of the PCB to a logic generator/analyzer, the National Instruments NI6552. The entire acquisition system, including the scaling of the supply voltage, was controlled by a Labview 7.1 program, allowing to automate the acquisition.

5.2 Performing the Attacks

We first wanted to ascertain whether it is possible to gracefully degrade the working of the chip through lowering the supply voltage by a small amount. The key intuition behind this attack methodology is that the signal lines of the circuit representing the critical paths for a specific portion should fail first when the feeding voltage is lowered. This effect is caused by the slower rising rate of the gates, which may fail to drive the longest delay paths within the timings enforced by the clock. These experiments were repeated on different samples of the same chip in order to investigate the effects of fabrication process variations.

The first experiment with the objective of finding out whether the ASIC degrades gracefully, was conducted by testing how many encryptions the chip was able to perform while lowering the voltage by 0.1 mV after each test. The voltage reduction was carefully carried out in order to exactly identify the voltage level at which it was possible to have only setup time violations but no functional errors which may cause the circuit to behave differently in response to the attack. At each voltage step, we collected the results of ten thousand encryption operations and compared them to the correct one. Figure 3 depicts the results of the experiments in terms of percentage of faulty ciphertexts versus functioning voltage. As can be seen from the figure, there is a 0.8 mV interval where the fault occurrence is limited to less than 10% of the outputs and the fault occurrences gradually increase while lowering further the voltage. The 0.8 mV zone where the fault occurrence is particularly limited is relevant when performing fault attacks as it maximizes the likelihood of inserting a single fault in the whole computation, as opposed to the catastrophic behavior shown when the supply voltage is considerably lowered as reported in [1, 19]. The 0.8 mV interval in supply voltage is well within the reach of the precision of the employed tunable power supply, thus we expect to be able to insert successfully exploitable faults.

To verify the impact of process variations on the position and width of the sensitive voltage range, we tested different samples of the same chip. Figure 4 reports the results of conducting the same campaign on five different sample

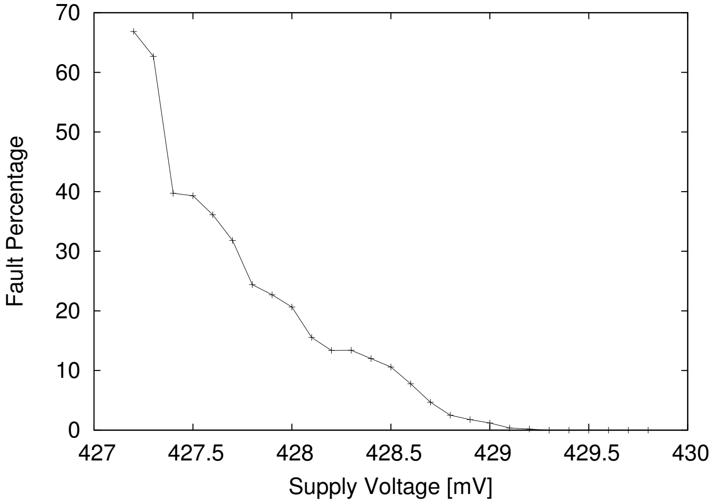

Fig. 3. Percentage of faulty outputs at different supply voltages for a single sample chip

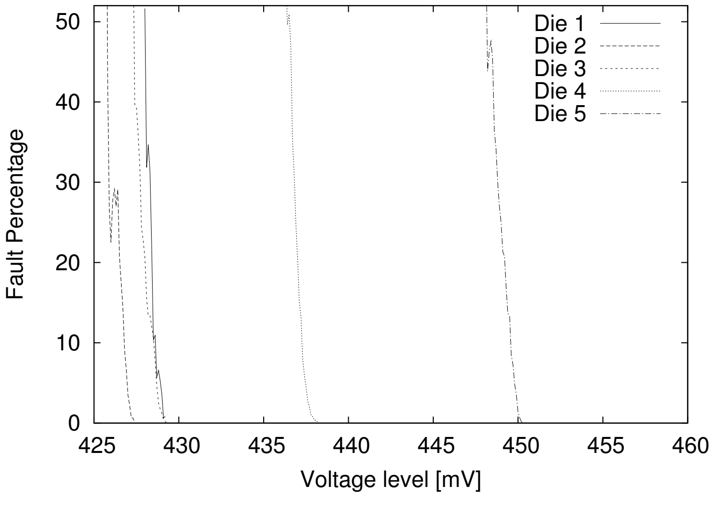

Fig. 4. Comparison of the voltage threshold of the first appearance of faults among different samples of the same chip

chips implementing the same AES design. As it can be clearly seen, process variations strongly affect the offset of the fault injection threshold for the sampled chip. However, it can be noticed that the rate of the degradation of the circuit performance is the same for all the samples. This in turn implies that, regardless of the different offsets in the fault onset zone, it is always possible to inject successfully single byte faults with the same low cost equipment.

After characterizing the graceful degradation of the chips, in terms of fault occurrence frequency, we moved on to investigate the actual fault pattern to discover if single byte faults were present in the erroneous ciphertexts which can be collected. To this end, we collected roughly 670K faulty ciphertexts while running the chip under test within the 10% faulty ciphertext region mentioned before. In order to uniformly stress the AES implementation, the plaintexts used during this campaign were selected from the NIST standard AES test vectors. The goal of the first analysis was to understand the variance of the fault patterns. By examining the faulty ciphertexts produced by the device, we found out that the errors induced by setup time violations caused only 822 unique faults. This implies that the positions where the faults occur are very regular, since there is an approximate repetition rate of 1000 for each fault pattern. The fault repetition rate was uniform with respect to the different plaintexts.

The last step in the characterization of attacks on this AES implementation was to find out how many faults out of the ones obtained were practically usable in order to carry out the attack. To this end, we analyzed the difference between the correct encryption process and the faulty one by rolling back the encryption process for the faulty ciphertexts. The inner states of the cipher at the beginning of each round obtained in this way were compared with the correct values and all the per-state differences were analysed. Recall that this approach does not impact the practical feasibility of the attack, since all the mentioned fault attack techniques to AES are able to successfully discard faults which do not fit the correct fault pattern.

The differential analysis of the faulty ciphertexts has shown that, out of the 822 unique faulty ciphertexts, 166 were single byte faults on a state of the cipher, thus resulting in 20% of the injected faults fitting the desired fault model (single byte modification). The fault patterns in the byte indicate that the byte is actually randomly modified with no particular sensitivity of a specific bit in the byte.

The last step to confirm the feasibility of mounting fault injection attacks on the chip is to verify that the faults are hitting the specific round positions required by the attack of [7]. Figure 5 shows the distribution of the single byte faults in the state of the cipher. As it is possible to notice, the faults are reasonably uniformly distributed, albeit with a tendency towards the last round and the middle ones. This different sensitivity to single byte faults can be ascribed

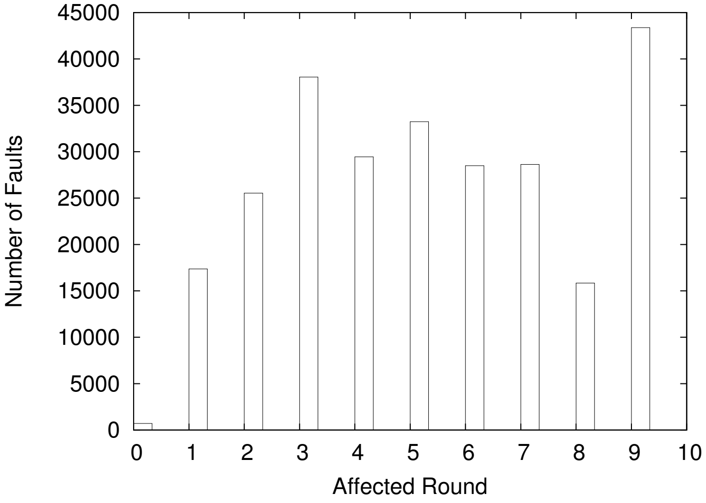

Fig. 5. Per round distribution of the single byte faults on the states of the cipher. Round 0 indicates the faults occurring before the cipher started, allegedly during the load operations

to a larger number of faults hitting the control unit, with respect to the ones for a specific round. This issue may be caused by a particular sensitivity of some inner paths of the control unit to setup time violations. The very low fault rate for the first state of the cipher is to be attributed to the fact that the architecture has just loaded the values and has not performed any significant operation on the plaintext yet. These results show that it is possible to generate successfully the required faults in order to break the AES implementation under consideration. In particular, since the position of the single byte fault in the inner state is almost uniformly distributed, the hypotheses made in [7, 16] on the required number of faults hold.

The confirmation of the feasibility of an attack on the chip was obtained by executing the aforementioned attacks on an ad-hoc C software implementation of the attack algorithms on a Core 2 Quad Q6600 based desktop. The observed key retrieval times were in the range of a few minutes, as expected from this attack technique.

6 Conclusions

In this paper, for the first time, the susceptibility to low cost fault injection attacks of a 65 nm subthreshold AES coprocessor specifically designed for RFID applications has been practically evaluated. We show that by using a precise power supply generator which allows power scaling in the range of 0.1mV, it is possible to inject the faults needed to recover the secret key. However, we noticed that compared to the usual situation, while attacking devices operating at subthreshold voltage, the ideal spot for the attack is located much closer to the operational voltage. Finally, we explored the effects of process variations on the ideal spot and noticed that its characteristic does not change when different chips are used. However, the location of the ideal spot is different for each chip.

Acknowledgements

This work was partially supported by the Walloon Region (E-USER and S@T Skywin projects) and by the Nanotera program (SecWear project). François-Xavier Standaert is an associate researched of the Belgian Fund for Scientific Research (FNRS-F.R.S.).

References

1. Alessandro Barenghi, Guido Bertoni, Emanuele Parrinello, and Gerardo Pelosi. Low Voltage Fault Attacks on the RSA Cryptosystem. In *Workshop on Fault Diagnosis and Tolerance in Cryptography*, pages 23–31, 2009.
2. Alessandro Barenghi, Guido M. Bertoni, Luca Breveglieri, Mauro Pellicioli, and Gerardo Pelosi. Low Voltage Fault Attacks to AES. In Mohammad Tehranipoor and Jim Plusquellic, editors, *HOST*. IEEE Computer Society, 2010.
3. D. Bol, R. Ambroise, D. Flandre, and J.D. Legat. Interests and limitations of technology scaling for subthreshold logic. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 17(10):1508–1519, 2009.
4. Chien-Ning Chen and Sung-Ming Yen. Differential fault analysis on aes key schedule and some countermeasures. *in Proc. Information Security and Privacy*, pages 217–217, 2003.
5. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.
6. R. Das and P. Harrop. RFID forecasts, players and opportunities 2011¡96¿-2021. *IDTechEx report*, 2010.
7. Pierre Dusart, Gilles Letourneux, and Olivier Vivolo. Differential Fault Analysis on A.E.S. *CoRR*, cs.CR/0301020, 2003.
8. M. Feldhofer, J. Wolkerstorfer, and V. Rijmen. AES implementation on a grain of sand. *Information Security, IEE Proceedings*, 152(1):13–20, 2005.
9. Christophe Giraud. DFA on AES. *Advanced Encryption Standard - AES, 4th International Conference*, 3373:27–41, 2005.
10. Michael Hutter, Thomas Plos, and Jörn-Marc Schmidt. Contact-Based Fault Injections and Power Analysis on RFID Tags. *in Proc. IEEE European Conference on Circuit Theory and Design*, pages 409–412, 2009.
11. Chong Hee Kim and Jean-Jacques Quisquater. New Differential Fault Analysis on AES Key Schedule: Two Faults Are Enough. *in Proc. International Conference on Smart Card Research and Advanced Applications*, pages 48–60, 2008.
12. Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. A systematic evaluation of compact hardware implementations for the rijndael s-box. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 323–333. Springer, 2005.

13. Amir Moradi, Mohammad T. Manzuri Shalmani, and Mahmoud Salmasizadeh. A generalized method of differential fault attack against AES cryptosystem. *in Proc. International Workshop on Cryptographic Hardware and Embedded Systems*, pages 91–100, 2006.
14. NIST. Announcing the advanced encryption standard aes. Technical report, Federal Information Processing Standards Publication 197, 2001.
15. David Peacham and Byron Thomas. A DFA attack against the AES key schedule. SiVenture Whitepaper, October 2006.
16. Gilles Piret and Jean-Jacques Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2779 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2003.
17. A. Satoh, S. Morioka, K. Takano, and S. Munetoh. A Compact Rijndael Hardware Architecture with S-Box Optimization. In *Proceedings of ASIACRYPT 2001*, number 2248 in LNCS, pages 239–254, 2000.
18. Jorn-Marc Schmidt, Michael Hutter, and Thomas Plos. Optical Fault Attacks on AES: A Threat in Violet. *in Proc. Workshop on Fault Diagnosis and Tolerance in Cryptography*, pages 13–22, 2009.
19. Nidhal Selmane, Sylvain Guilley, and Jean-Luc Danger. Practical Setup Time Violation Attacks on AES. In *EDCC-7 '08: Proceedings of the 2008 Seventh European Dependable Computing Conference*, pages 91–96, Washington, DC, USA, 2008. IEEE Computer Society.