A Security Comparison between AES-128 and AES-256 FPGA implementations against DPA attacks

Virginia Zúñiga González[1], Erica Tena-Sanchez*[1,2], Antonio J. Acosta[1,3]
[1]*Instituto de Microelectrónica de Sevilla (CSIC / University of Seville)*
[2]*Department of Electronics Technology, Escuela Politécnica Superior, University of Seville*
[3]*Department of Electronics and Electromagnetism, Physics Faculty, University of Seville*
Seville, Spain
*erica@imse-cnm.csic.es

***Abstract*—As the AES is the standard symmetric cipher selected by NIST, is the best-known and the most widely used block cipher. Consequently, security threats are constantly rising and increasingly powerful. With the addition of the upcoming scenario of quantum computing, these threats have become a front-line concern in the crypto-community. Although is claimed that using larger key sizes in symmetric key algorithms for implementing quantum-resistant implementations is enough to counteract brute force attacks, this paper shows that both AES-128 and AES-256 are vulnerable to Power Analysis attacks. This paper presents a security comparison against Differential Power Analysis (DPA) attacks over both AES 128-256. Through experimental attacks in FPGA AES implementations, results show that although AES-256 reaches a greater level of security than AES-128, is still vulnerable to this kind of attack. Specifically, we have obtained 75% of the bytes needed to find the original key for AES-128 while only 28.125% for AES-256 by performing the same attack.**

***Index Terms*—Security comparison, AES cipher, Differential Power Analysis attack, MTD, FPGA**

I. Introduction

Cyber-security plays an essential role, from business to general public safety. Cryptography is currently used to encrypt bank cards, smartphones, access control car lock systems or restricted areas, and even pay with payment prepaid telephone cards among many others. Unlike guarded computers and cloud servers, most embedded and cyber-physical IoT devices are physically accessible to exploit vulnerabilities and extract sensitive information from these devices. As a result, the risk of compromising secret information is significantly higher due to the potential for direct manipulation by malicious individuals. With such a forecast, cryptographic hardware is expected to increase demand for energy efficiency, but above all hardware reliability and security [1].

Encryption systems serve the purpose of scrambling data and keeping them safe from unauthorized access. For this reason, the US National Institute of Standards and Technology (NIST) indicates the implementation of approved conventional cryptographic standards. At the beginning of the century, NIST declared the symmetric block cipher Advanced Encryption Standard (AES) as the standard, becoming the most widely used cipher in different areas of application to this day [2]. Through this process, NIST and the international scientific community discussed the mandatory requirements of the ciphers. Under these specifications, the AES must have a 128-bit block size, three key lengths options can be supported: 128, 192, and 256-bit, security relative to others, and efficiency in software and hardware [3].

In view of compromising security by incorporating these encryption systems into electronic devices, extensive research has been studied. Those have demonstrated the existence of connections between power consumption, electromagnetic emissions, thermal patterns, and other phenomena with the encryption processes during encryption. This area of exploration, known as Side-Channel Analysis (SCA), has remained dynamic over the past two decades [4]. Nowadays, this field of study has been active in finding ways to evaluate the security of cipher implementations, exploiting them to recover encryption keys, and protecting implementations from attacks.

In particular, the attacks using the relation between the power consumed by the device and the data being computed, have attracted significant attention because of their effectiveness [5]. The power consumed during calculation operations is directly associated with the processed data, meaning that power measurements contain valuable information about the circuit's switching. Remarkably, even the effects of a single transistor can manifest as subtle correlations in power measurements using Differential Power Analysis (DPA). In other words, this attack takes advantage of the fact that it frequently is feasible to compromise the system by utilizing statistical techniques specifically designed for the target algorithm.

Lastly, the use of quantum computing is now of great relevance in defining the security of a system. A quantum key-recovery attack refers to a method that retrieves the key more rapidly than an exhaustive search, without attempting all key possibilities. In [3] authors affirm that while asymmetric algorithms, like the well-known RSA, are susceptible to attacks making use of quantum computers, symmetric algorithms exhibit significantly higher resilience. Moreover, a quantum computer utilizing Grover's algorithm could execute a full

979-8-3503-0385-8/23/$31.00 ©2023 European Union

key search on a cipher with a keyspace of 2^n elements in $2^{(n/2)}$ steps. Therefore, key lengths of more than 128 bits are required to guarantee resistance against quantum computer attacks. This realization also served as the impetus behind the insistence on 192-bit and 256-bit key lengths for AES in actual applications. An analysis for the first time of the post-quantum security of AES is shown in [6]. Although AES-256 is considered quantum-resistant against brute-force attacks, due to its architecture, it can be vulnerable to DPA attacks. In this paper we analyze these vulnerabilities on unprotected implementations in FPGA, comparing the security results with AES-128.

The organization of the paper is as follows. In Section II, we introduce the AES architecture and discuss the differences between its versions with 128 and 256 bits key lengths. Section III summarizes all the vulnerabilities of AES-128 and AES-256. Section IV shows the set-up lab and results. Finally, in section V, the conclusions are given.

II. Description of AES-128 and AES-256 Architecture and Operation

The AES is a symmetric key cryptographic (SKC) algorithm. In SKC, the same key is used for encryption and decryption ($KeyA = KeyB$) and both sender and receiver know the value of the key [1]. On the other side, it encrypts a whole data block in every iteration. So a given *plaintext* will always result in the same *ciphertext* using the same key. As mentioned above, the AES key size varies between 128, 192, and 256 bits but its structure operates with 128-bit data blocks for both the key and plaintext.

This cipher consists of layers that handle 128-bits data blocks throughout the data path as shown in Fig. 1. Each encryption round is made up of four different layers: Substitution, Shiftrow, MixColumn, and Key Addition layer. Moreover, the last round does not make use of the MixColumn transformation, which makes the encryption and decryption scheme symmetric [3]. The substitution layer applies the known AES SBox look-up table transformation on each byte. Shiftrow rotates to the left of every one of the rows of the state matrix. The first row of the AES state does not change, while the second row rotates one position, the third row rotates two and, the last rotates three positions. After that, the Mixcolumn layer multiplies the state by a matrix of fixed data. At last, the Key Addition layer only consists of an XOR operation to add the key in the process.

The addition of the key in the process is as follows. Before the first round, *plaintext* is mixed with the first 128 bits of the original key, K_0 using the Key Addition layer. On every round, the Key Addition layer is responsible for mixing the current 16-byte state matrix with a subkey that has been gotten from the original key in the Key Schedule process. The Key Schedule takes the original input key (in 128-bit blocks) and generates subkeys building what is known as the expanded key. The number of subkeys is equal to the number of rounds plus one, due to the first Key Addition layer. Thus, for AES-128, the number of rounds is *nr* = 10, and there are 11 subkeys.

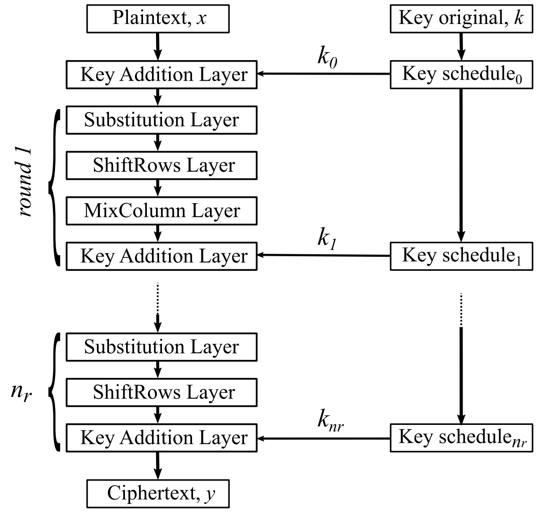


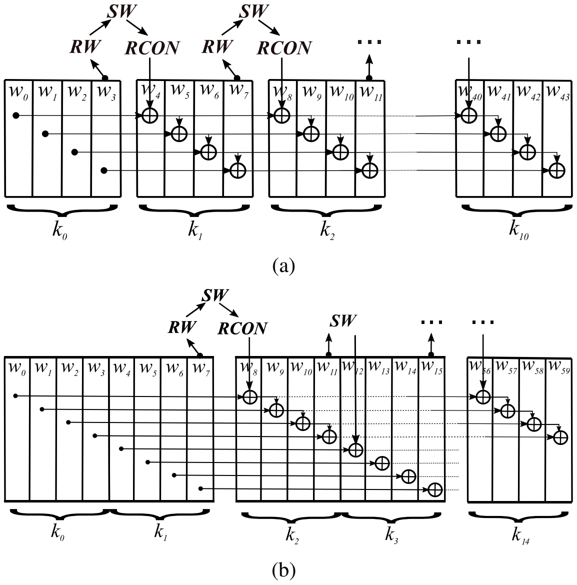
Fig. 1: AES encryption algorithm diagram.

Meanwhile, AES-256 requires a total of *nr* = 14 rounds with 15 subkeys, each of 128 bits. Accordingly, a subkey has four words so the keys expanded have 44 and 60 words respectively.

The AES encryption algorithm employs different Key Schedules tailored for its three key sizes. Although sharing many similarities, this is what differentiates the AES-128 and AES-256 algorithms from each other, in addition to increasing the number of rounds. The AES Key Schedule is word-oriented, where every subkey is stored in a key expansion array W. For AES-128, the Key Schedule scheme is shown in Fig. 2(a). The last Ws of the K_0 to K_9 will be modified with the following transformations. *RotWord* (RW) rotates each column element one position up in the 4-byte array. After, *SubWord* (SW) apply, like the Substitution layer, the SBox look-up table on each byte. Finally, *RCON* does a XOR operation between the first byte of the column and a constant byte. This *RCON* value is known and changes with each iteration. For the AES-256, a fairly similar approach is carried out but it exists some variations. Notice that the original key fills the first two subkeys, K_0 and K_1, instead of only the first one. In other words, the subkey for the first AES round, K_0, is formed by the array elements W_0, W_1, W_2, W_3, the first four bytes of the original key. Then, the second subkey, K_1, is the last four bytes archived as the elements W_4, W_5, W_6, W_7. Thus, only seven *RCON* coefficients are used instead of ten as AES-128 does. Despite this, the main distinction between both processes is applying *SubWord* onto the fourth column of each of the two subkeys to compute the next word, for example, as is illustrated in Fig. 2(b) for W_{11} and W_{12} [10]. Unlike AES-128, this means that not all Key Schedule rounds are the same.

III. AES-128 and AES-256 Security, Vulnerabilities and Attacks

Classical cryptanalysis on AES reveals that the key size is crucial in the security of the cipher. While for AES-128, there are no known attacks faster than exhaustive search, AES-256


SW
RW
RCON
SW
RW
RCON
...
...
w_0 w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8 w_9 w_{10} w_{11} w_{40} w_{41} w_{42} w_{43}
k_0 k_1 k_2 k_{10}
SW
RW
RCON
SW
...
...
w_0 w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8 w_9 w_{10} w_{11} w_{12} w_{13} w_{14} w_{15} w_{56} w_{57} w_{58} w_{59}
k_0 k_1 k_2 k_3 k_{14}

(a)

(b)

Fig. 2: Key Schedule architecture for (a)AES-128 and (b)AES-256.

was shown to be breakable by attacks that require $2^{99.5}$ time in [7]. Consequently, AES is no longer regarded as theoretically secure. However, the pressing question facing us all is the extent to which it is practically insecure.

Although the security level of the ciphers was primarily based on their mathematical formulation and key length in the past, the implementation of these algorithms in electronic devices is more complex. Physical attacks can occur at different stages of the circuit design phase by simulation or at different levels (algorithm level, block level, or transistor level cells), even experimental on the circuit already realized in an ASIC or FPGA. In addition, it is important to differentiate whether security is measured to detect possible vulnerabilities and to be able to apply countermeasures to avoid information leakage (for example SCA or Fault Injection [1]) or simply to demonstrate that a system is vulnerable, regardless of the exact details of the leakage point or conditions (TVLA [8]). It is easy to assume that the former is a greater threat since information can be obtained so our focus will be directed towards them.

Various attack strategies exhibit significant differences in terms of cost, time, equipment, and expertise. They can be classified based on whether they are active/passive or invasive/non-invasive. Invasive attacks involve manipulating the device, while non-invasive attacks gather information without altering the device. Passive attacks reveal the secret key by the cryptographic device operates in the correct way during encryption, whereas active attacks modify the functionality. Invasive attacks, whether passive or active, are powerful but expensive and can cause irreparable damage to the cryptographic device. On the other hand, non-invasive attacks, requiring minimal resources, pose a substantial threat with high effectiveness. Due to this, researchers show great interest in studying the security against these latest attacks called Side-channel Attacks (SCA). SCAs exploit the existence of dependence between power consumption, electromagnetic emissions, thermal patterns, and other phenomena with the encryption data of the device [1].

In this paper, we will focus on power consumption attacks and, concretely the well-known Differential Power Analysis (DPA) because has received a lot of attention from researchers, for instance for the AES in [9]. Introduced by Kocher et al. in [5], DPA automatically locates correlated regions in a power consumption trace, the attack can be automated and little or no information about the target implementation is required.

The goal of a DPA attack is to reveal the secret key of cryptographic devices using a large number of measured power consumption traces from the devices while encrypting or decrypting data. In practice to execute a DPA attack, an adversary initially observes and captures m encryption operations or power traces $T[1 : m]$, each containing n samples. Furthermore, the attacker obtains recordings of the ciphertexts $y[1 : m]$, plaintexts $x[1 : m]$, or both. The analysis is focused in an intermediate operation of the cipher, where the secret key is operated with a known value. Thus, depending on this, is common to see attach on the first or the last round, where the plaintexts or ciphertext respectively are known. Once the operation of interest has been selected and the power consumption traces measured, the next step in the DPA attack is to calculate the intermediate hypothetical values for each of the possible keys. These hypothetical values are then mapped to power consumption values using a power model. The choice of the power model will largely determine the success of the attack. In this work, we use the $Hamming{-}DistanceModel$. To conclude, the H matrix is compared with the power consumption traces measured, T. The maximum correlation value of both matrices should be the correct key. In Fig. 4, we can see the steps in a complete DPA attack on the first round.

The attack on the first round tries to find K_0 while the attack on the last round will get the last subkey of the expanded key K_{nr}. For the second of the attacks, a second step is necessary which is to reverse the Key Schedule process until the original key is obtained. The difficulty of reversing the key programming process depends on the key length. Attacking the AES-128 involves discovering only one of the subkeys of the expanded key. If it is an intermediate or last subkey, the original key k_0 is gotten by reversing the Key Schedule process. On the other hand, reversing the process for AES-256 requires knowledge of two consecutive subkeys, see Fig. 2. This fact, it often necessitates the execution of two consecutive interdependent attacks [3], [10], [11]. The first attack follows a similar approach as that used for AES-128, but a second attack, that is directly contingent on the success of the first one, must be done. Such a configuration can significantly increase the complexity for potential attackers, particularly when they need to carry out new signal acquisitions using specific inputs built based on the first part of the recovered key. Any error or uncertainty in the initial attack further complicates the process of key recovery. In [11], an attack is exposed to guess parts of the two consecutive round keys K_{14} and K_{13}.

Alternative suggestions like [10] whose contribution is proving that the complexity can be reduced to two independent attacks by targeting the first and last round keys separately. They demonstrate that the available information is sufficient to recover the main key, or at least a very small list of potential candidates, with minimal exploratory effort. Even so, this study implies knowing 32 bytes of the expanded key instead of 16 for AES-128, making it more difficult and time-consuming for the attacker.

IV. Case of study and results

The security of both ciphers has been studied experimentally. The AES-implemented design is joint for both versions for a fair comparison, with key lengths of 128 and 256 bits, and we will refer to this one as AES128-256 [12]. The performance of the AES implementation is analyzed in terms of timing and resource occupation using the device Spartan-6 Xilinx device. Some design specifications are in Table I. The AES128-256 uses 3644 LUTs in total so requires 7% of FPGA resources. The AES response time depends on the number of bits in the key and whether an encryption or decryption operation is performed. There is no dependency on the specific value of the key nor on the value of the data to be encrypted or decrypted. The number of clock cycles in the function of the selected operation is shown as well.

TABLE I: Resources required for the implementation of AES128-256 in Spartan-6 XC6SLX75-2CSG484.

Feature	AES-128	AES-256
Encryption clock cycles	11	15
Decryption clock cycles	21	22
Encryption Throughput at 100 MHz (Mbps)	1163,64	853,33
Decryption Throughput at 100 MHz (Mbps)	609,52	581,81
Maximum frequency (MHz)	104,264	
LUTS	3644 (7%)	

As an experimental setup, we propose the scheme shown in Fig. 3, where the used equipment is: computer, power supply, oscilloscope, and SAKURA-G FPGA board (a specific board designed for SCA attacks) [13]. For trace acquisition, the oscilloscope Keysight InfiniiVision DSOX3054T is used, with 4 G/samples and a bandwidth of 500MHz. The computer controls the instruments, communicates with the AES implementation in the SAKURA-G FPGA, and processes the data to carry out the DPA attacks. And, the power supply Keysight E36312A supplies the SAKURA-G board precisely.

The attack performed reveals byte by byte of the attacked subkey to minimize the computation of the operation. For this purpose, we attacked in the last round as it is more vulnerable to these attacks. The followed attack scheme is shown in Fig. 4. We can see in Fig. 5(a) and Fig. 6(a), a revealed subyte for calculating a peak correlation over the last round for AES-128 and AES-256 respectively. Due to AES-256 traces having more sample points, because there are more encryption rounds, trace acquisition is slower than for AES-128. A longer key means more data to send, receive, and process. The security achieved for each attack is measured using the Measurements to Disclose (MTD) the key, which determines the minimum number of input patterns needed to retrieve the secret key. As usual metrics, it is shown in Fig. 5(b) and Fig. 6(b) the correlation versus the number of power consumption traces (zoomed 1 to 20k traces). The correct subyte can be visualized by distinguishing the correlation bold line. This value increases the more traces are used for the attack while the rest of the hypothetical subytes values converge to zero.

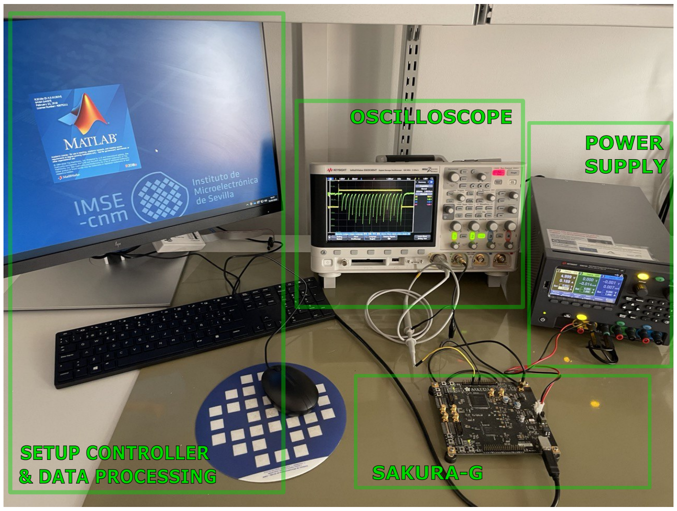

OSCILLOSCOPE
POWER SUPPLY
SETUP CONTROLLER & DATA PROCESSING
SAKURA-G
MATLAB
IMSE -cnm
Instituto de Microelectrónica de Sevilla

Fig. 3: Experimental setup scheme.

The attacks have been made using a total of 100.000 power consumption traces, for random plaintexts. Being aware that the used key influences the MTD value, we attacked using three different keys. This allows us to compare the values more efficiently. The first keys for both implementations, key_1, are a public NIST key test while the rest are pseudo-random.

The aim of this work is to compare the security of both AES in relation to how many bytes (subytes) are known versus the power consumption trace number needed. For this purpose, we based on the results in Table II where the MTD values (x1000) are summarized for each subyte, key used, and cipher. The symbol $++$ specifies that the correct subyte value has not been obtained with the complete data set for this test, i.e. it needs a larger number of traces. The number of the subyte obtained for each key and implementation is shown in bold in the last row of this table.

Based on these data, we can draw the following conclusions. If we pay attention to the MTD for each subkey we can observe that for AES-256 more traces on average are needed. The AES-256 includes a higher number of operations which causes its power consumption to be affected by more switching at the circuit cell level. This situation can make the dependence of the consumption value and the processed data more easily masked. Consequently, the correlation obtained in the DPA attack between hypothetical and real power consumption values is lower. A higher MTD reinforces the small dependence and makes it more likely to get the subyte. It is important to note that there is a strong dependence on the key used. The value

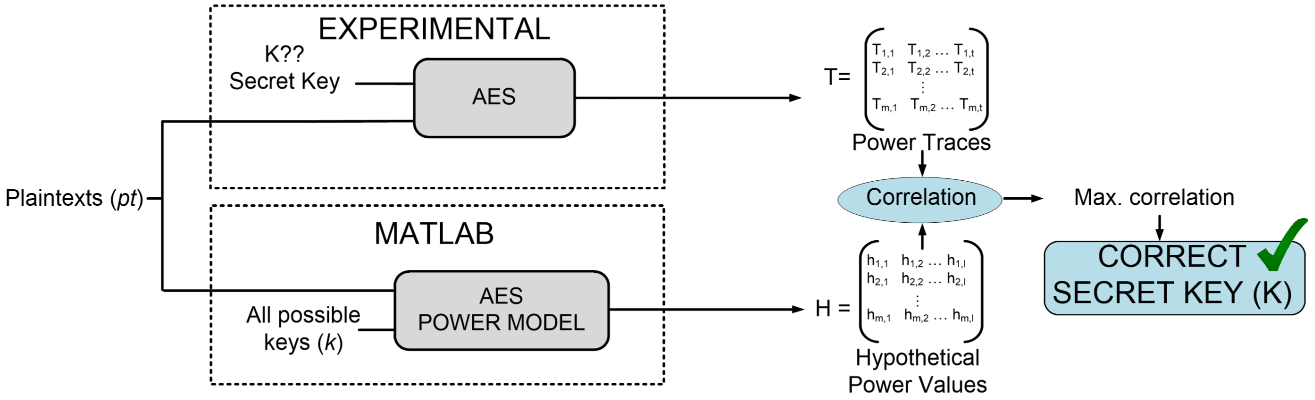
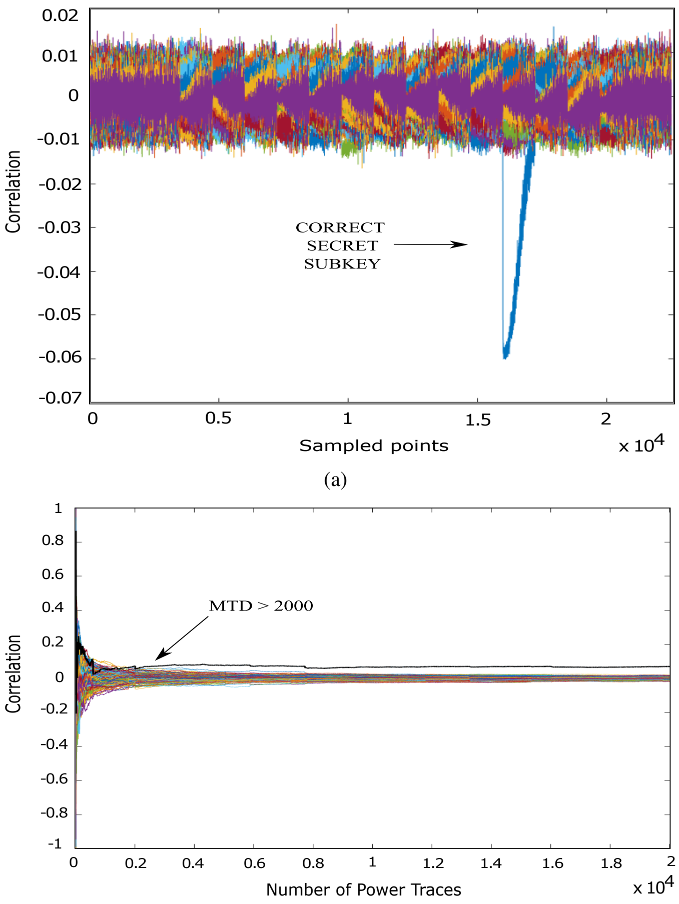


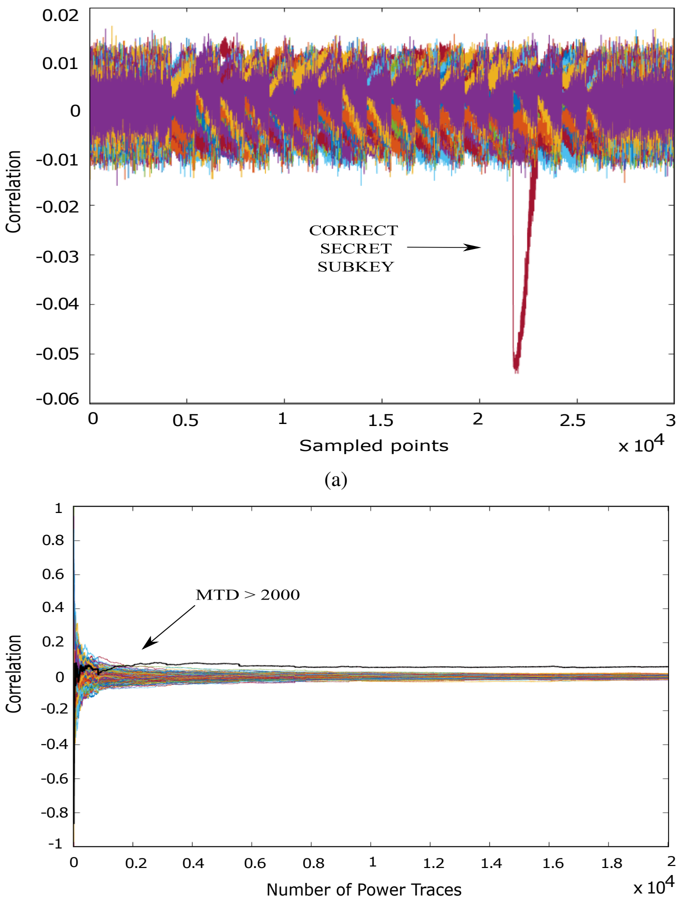
Fig. 4: DPA attack scheme.



(a)

(b)

Fig. 5: Correlation versus samples and their evolution related to using a higher number of traces for AES-128.



(a)

(b)

Fig. 6: Correlation versus samples and their evolution related to using a higher number of traces for AES-256.

of the key affects the security of the device. But it is hard to know which keys make the cipher more vulnerable.

The average number of uncovered subytes for AES-128 is 75% while for AES-256 it is 56.25%. It is important to note that this percentage is only based on the 16 bytes of the last round of expanded key. As it is explained in this study previously, knowing the expanded key is not enough for a complete AES-256 attack. We have to attack one more subkey, for example, the first round like in [10] to reduce key candidate values or the K_{13} to acquire the whole original key like in [11]. As a result, a significantly more complex algoritm DPA attack is required. It also exposes an additional 16 bytes to attack,

TABLE II: MTDs required to attack AES-128 and AES-256 key subytes.

Number of Subyte	AES-128			AES-256		
	Key1	*Key2*	*Key3*	*Key1*	*Key2*	*Key3*
Subyte1	90	30	++	20	++	++
Subyte2	4	8	90	++	40	4
Subyte3	50	30	10	++	++	20
Subyte4	4	2	2	4	70	++
Subyte5	80	4	++	4	++	++
Subyte6	4	2	40	100	20	20
Subyte7	2	20	++	++	++	20
Subyte8	++	++	4	++	30	++
Subyte9	++	8	++	++	6	++
Subyte10	80	4	++	++	++	30
Subyte11	30	8	30	++	4	4
Subyte12	6	20	40	8	++	10
Subyte13	++	4	++	70	70	20
Subyte14	++	2	30	20	70	20
Subyte15	20	60	++	60	++	++
Subyte16	10	10	30	10	8	++
# Subytes revealed	**12**	**15**	**9**	**9**	**9**	**9**

*MTDs are x1000 the represented values.

which prolongs the execution time. Hence, only 28.125% of the total of subytes needed for the AES-256 DPA attack is obtained.

Although longer keys ensure the necessary security levels to prevent brute force attacks in a post-quantum era, this paper has shown that they are still vulnerable to side-channel attacks. That is why the need to include hardware countermeasures to prevent such attacks. For this purpose, there is a wide range of possibilities that, depending on the requirements and limitations of the application, we can choose from and that can best fit our design [14].

V. Conclusions

This paper has presented a study about how the security varies between 128 or 256-bit key size architecture of the Standard cipher AES. Once assumed that a brute force attack is a threat for AES-128 while is not for AES-256, even in the presence of quantum computation, the next step is assessing the security once these encryption systems are implemented on a device because they become susceptible to physical attacks. For that, this paper presents a security comparison based on one of the more powerful attacks, the DPA attack.

An AES128-256 design has been implemented in a Spartan-6 using a SAKURA-G board and an appropriate lab setup for this proposal. By conducting a DPA attack using a total of 100,000 traces and employing three random keys for each cipher, we observed that targeting individual bytes in the last encryption round revealed that the AES-256 exhibits higher resistance in unveiling the subkeys compared to AES-128. On average, subkeys need more power consumption traces to be revealed. Using the complete dataset for this test, only 56.25% of the subytes, 16 bytes in total, for AES-256 are attacked compared to 75% for AES-128. The conclusion is that the AES-256's increased number of operations leads to more switching at the circuit cell level, impacting its power consumption and making it more resistant to DPA attacks. Also, we must consider that with the fully known expanded key for the AES-128, it is possible to obtain the original key. Nevertheless, a more complex attack algorithm is needed to find another subkey for the AES-256 to recover its original key. This implies a more complex attack, more computation, a larger number of traces, and thus, more attack time.

Acknowledgment

This work has been funded by SPIRS (Secure Platform for ICT Systems Rooted at the Silicon Manfacturing Process) Project with Grant Agreement No. 952622 under the European Union's Horizon 2020 research and innovation programme. Authors want to thank Programa Operativo FEDER 2014-2020 and Consejería de Economía, Conocimiento, Empresas y Universidad de la Junta de Andalucía under Project US-1380823, and project Grant PID2020-116664RB-I00 funded by MCIN/AEI/10.13039/501100011033.

References

[1] Acosta, Antonio J., Tommaso Addabbo, and Erica Tena-Sánchez. ”Embedded electronic circuits for cryptography, hardware security and true random number generation: an overview.” International Journal of Circuit Theory and Applications 45.2: 145-169, 2017.

[2] Pub, NIST FIPS. ”197: Advanced encryption standard (AES).” Federal information processing standards publication 197.441 (2001): 0311.

[3] Paar, Christof, and Jan Pelzl. Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media, 2009.

[4] Randolph, Mark, and William Diehl, “Power side-channel attack analysis: A review of 20 years of study for the layman,” Cryptography, vol. 4, no 2, p. 15, 2020.

[5] Kocher, Paul, Joshua Jaffe, and Benjamin Jun. ”Differential power analysis.” Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19. Springer Berlin Heidelberg, 1999.

[6] Bonnetain, Xavier, María Naya-Plasencia, and André Schrottenloher. ”Quantum security analysis of AES.” IACR Transactions on Symmetric Cryptology 2019.2: 55-93, 2019.

[7] Biryukov, Alex, and Dmitry Khovratovich. ”Related-key cryptanalysis of the full AES-192 and AES-256.” Advances in Cryptology–ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15. Springer Berlin Heidelberg, 2009.

[8] J. Cooper, E. Demulder, G. Goodwill, J. Jaffe, and G. Kenworthy, ”Test Vector Leakage Assessment methodology in practice,” in International Cryptographic Module Conference, vol. 20, 2014.

[9] Mangard, Stefan, Norbert Pramstaller, and Elisabeth Oswald. ”Successfully attacking masked AES hardware implementations.” Cryptographic Hardware and Embedded Systems–CHES 2005: 7th International Workshop, Edinburgh, UK, August 29–September 1, 2005. Proceedings 7. Springer Berlin Heidelberg, 2005.

[10] Wurcker, Antoine. ”Ease of side-channel attacks on AES-192/256 by targeting extreme keys.” Cryptology ePrint Archive (2019).

[11] Moradi, Amir, Markus Kasper, and Christof Paar. ”Black-box side-channel attacks highlight the importance of countermeasures: An analysis of the Xilinx Virtex-4 and Virtex-5 bitstream encryption mechanism.” Topics in Cryptology–CT-RSA 2012: The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27–March 2, 2012. Proceedings. Springer Berlin Heidelberg, 2012.

[12] D2.1, “SPIRS Deliverable D.2.1, Initial Design of RoT Components,” December 2022.

[13] Guntur, Hendra, Jun Ishii, and Akashi Satoh. ”Side-channel attack user reference architecture board SAKURA-G.” 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE). IEEE, 2014.

[14] Tena-Sánchez, Erica, et al. ”Gate-level hardware countermeasure comparison against power analysis attacks.” Applied Sciences 12.5 (2022): 2390.