Antonella Cioffi

This thesis focuses on the study of the Internet of Things devices security against two typologies of side-channel attacks (SCAs): power measurements-based and timing measurements-based attacks. In particular, three aspects of the IoT security against SCA are faced: (i) improvement of power attack efficiency, (ii) improvement of Vulnerability Assessments tests, and (iii) techniques for robustness evaluation of cryptographic algorithms.

The improvement of power attack efficiency is aimed at decreasing the effort and the time needed to find the secret key of the cryptographic algorithm. To this aim, an enhanced version of the power attack was realized by increasing the signal-to-noise ratio (SNR) of the power traces before applying statistical operations. The proposed low-pass filter allows to discover the entire key unlike the lack of pre-processing with the same number of traces. Moreover, the filter is able to reduce the number of traces needed for a successful attack. Another method was proposed to improve the attack efficiency. It consists of a novel optimization technique by relying on fractional experimental design able to identify the parameter values maximizing the number of correctly discovered key bytes with the minimum cost of the attack in terms of time and resources. The proposed method also contributes to the improvement of metrological aspects such as the reduction of repeatability and reproducibility uncertainty.

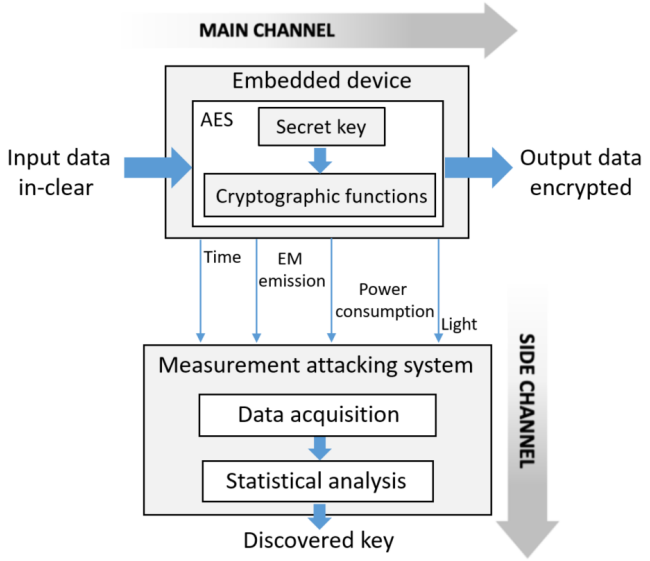
For the improvement of Vulnerability Assessments tests, needed to evaluate the robustness of IoT devices respect to the side-channel attacks, a method for assessing the security at varying the power countermeasures is proposed. The method provides a metric to express the effectiveness of a countermeasure in straightening the IoT transducers security, and it is able to identify the most effective countermeasure. As concerns the techniques for the robustness evaluation of cryptographic algorithms, an experimental test of robustness to timing attack is reported for the asymmetric cryptographic algorithm most used in Internet of Things framework, i.e. the Elliptic Curve Digital Signature Algorithm (ECDSA). Its robustness is demonstrated experimentally by measuring the execution times of the cryptographic algorithm and applying a classical method proposed in literature, based on the use of a mathematical object known as lattice.

Enhanced attacks and security assessment based on leakage measurements in embedded devices

2021 - XXXIV

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

ENHANCED ATTACKS AND SECURITY ASSESSMENT BASED ON LEAKAGE MEASUREMENTS IN EMBEDDED DEVICES



ANTONELLA CIOFFI

PH.D. IN INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

PH.D. THESIS

IN

INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

ENHANCED ATTACKS AND SECURITY ASSESSMENT BASED ON LEAKAGE MEASUREMENTS IN EMBEDDED DEVICES

ANTONELLA CIOFFI

TUTOR: PROF. PASQUALE ARPAIA

COORDINATOR: PROF. DANIELE RICCIO

XXXIV CICLO

SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL'INFORMAZIONE

UNIVERSITY OF NAPLES FEDERICO II

PH.D. THESIS
IN
INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Enhanced attacks and security assessment based on leakage measurements in embedded devices

Supervisors:
Prof. Pasquale Arpaia
Dr. Francesco Bonavolontà
Ing. Vincenzo Pascariello

Candidate:
Antonella Cioffi

2021 - 2022

"The mind is not a vessel to be filled,
but a fire to be kindled"
cit. Plutarco

Abstract

This thesis focuses on the study of the Internet of Things devices security against two typologies of side-channel attacks (SCAs): power measurements-based and timing measurements-based attacks. In particular, three aspects of the IoT security against SCA are faced: (i) improvement of power attack efficiency, (ii) improvement of *Vulnerability Assessments* tests, and (iii) techniques for robustness evaluation of cryptographic algorithms.

The improvement of power attack efficiency is aimed at decreasing the effort and the time needed to find the secret key of the cryptographic algorithm. To this aim, an enhanced version of the power attack was realized by increasing the signal-to-noise ratio (SNR) of the power traces before applying statistical operations. The proposed low-pass filter allows to discover the entire key unlike the lack of pre-processing with the same number of traces. Moreover, the filter is able to reduce the number of traces needed for a successful attack. Another method was proposed to improve the attack efficiency. It consists of a novel optimization technique by relying on fractional experimental design able to identify the parameter values maximizing the number of correctly discovered key bytes with the minimum cost of the attack in terms of time and resources. The proposed method also contributes to the improvement of metrological aspects such as the reduction of repeatability and reproducibility uncertainty.

For the improvement of *Vulnerability Assessments* tests, needed to evaluate the robustness of IoT devices respect to the side-channel attacks, a method for assessing the security at varying the power countermeasures is proposed. The method provides a metric to express the effectiveness of a countermeasure in straightening the IoT transducers security, and it is able to identify the most effective countermeasure.

As concerns the techniques for the robustness evaluation of cryptographic algorithms, an experimental test of robustness to timing attack is reported for the asymmetric cryptographic algorithm most used in Internet of Things framework, i.e. the Elliptic Curve Digital Signature Algorithm (ECDSA). Its robustness is demonstrated experimentally by measuring the execution times of the cryptographic algorithm and applying a classical method proposed in literature, based on the use of a mathematical object known as lattice.

Keywords: Security, Internet of Things (IoT), side-channel attack (SCA), measurement, cryptography.

Cover Image: Block diagram of leakage measurement-based attacks on embedded systems.

Sommario

Questa tesi si concentra sullo studio della sicurezza dei dispositivi Internet of Things contro due tipologie di attacchi side-channel (SCA): attacchi basati su misurazioni di potenza e attacchi basati su misurazioni di tempo. In particolare, vengono affrontati tre aspetti della sicurezza IoT contro gli attacchi sopra menzionati: (i) miglioramento dell'efficienza degli attacchi di potenza, (ii) miglioramento dei test di *Vulnerability Assessments* e (iii) tecniche per la valutazione della robustezza degli algoritmi crittografici.

Il miglioramento dell'efficienza dell'attacco di potenza è finalizzato a ridurre lo sforzo e il tempo necessari per trovare la chiave segreta dell'algoritmo crittografico. A tal fine, è stata realizzata una versione potenziata dell'attacco di potenza aumentando il rapporto segnale/rumore (SNR) delle tracce di potenza prima di applicare operazioni statistiche. Il filtro passa-basso proposto consente di scoprire l'intera chiave rispetto al caso di assenza di pre-elaborazione a parità del numero di tracce. Inoltre, il filtro è in grado di ridurre il numero di tracce necessarie per un attacco riuscito. Un altro metodo è stato proposto per migliorare l'efficienza dell'attacco. Questo consiste in una nuova tecnica di ottimizzazione basata su un disegno sperimentale frazionario in grado di identificare i valori dei parametri massimizzando il numero di byte di chiave correttamente scoperti con il minimo costo dell'attacco in termini di tempo e risorse. Il metodo proposto contribuisce anche al miglioramento di aspetti metrologici quali la riduzione della ripetibilità e dell'incertezza di riproducibilità.

Per il miglioramento dei test di *Vulnerability Assessments*, necessari per valutare la robustezza dei dispositivi IoT rispetto agli attacchi side-channel, viene proposto un metodo per valutare la sicurezza al variare delle contromisure di potenza. Il metodo fornisce una metrica per esprimere l'efficacia di una contromisura nel rafforzare la sicurezza dei trasduttori IoT, ed è in grado di identificare la contromisura più efficace.

Per quanto riguarda le tecniche per la valutazione della robustezza degli algoritmi crittografici, si riporta un test sperimentale di robustezza al timing attack per l'algoritmo crittografico asimmetrico più utilizzato nel framework Internet of Things, ovvero l'Elliptic Curve Digital Signature Algorithm (ECDSA). La sua robustezza è dimostrata sperimentalmente misurando i tempi di esecuzione dell'algoritmo crittografico e applicando un metodo classico proposto in letteratura, basato sull'utilizzo di un oggetto matematico noto come *lattice*.

Keywords: Sicurezza, Internet delle cose (IoT), attacco side-channel (SCA), misurazione, crittografia.

Cover Image: Diagramma a blocchi degli attacchi basati su misurazioni di perdite dai sistemi embedded.

List of publications

Journal publications:

(i) Pasquale Arpaia, Francesco Bonavolontá, and Antonella Cioffi; *"Problems of the advanced encryption standard in protecting Internet of Things sensor networks."* **Measurement**, 161 (September 2020).
doi:10.1016/j.measurement.2020.107853

(ii) Pasquale Arpaia, Francesco Bonavolontá, Antonella Cioffi and Nicola Moccaldi; *"Power Measurement-Based Vulnerability Assessment of IoT Medical Devices at Varying Countermeasures for Cybersecurity"*. **IEEE Transactions on Instrumentation and Measurement**, vol. 70, pp. 1-9, (June 2021).
doi: 10.1109/TIM.2021.3088491.

(iii) Pasquale Arpaia, Francesco Bonavolontá, Antonella Cioffi and Nicola Moccaldi; *"Reproducibility enhancement by optimized Power Analysis attacks in Vulnerability Assessment of IoT transducers"* **IEEE Transactions on Instrumentation and Measurement**, vol. 70, pp. 1-8, (August 2021).
doi: 10.1109/TIM.2021.3107610.

Conference proceedings:

(i) Pasquale Arpaia, Francesco Bonavolontà, and Antonella Cioffi; *"Security vulnerability in Internet of Things sensor networks protected by Advanced Encryption Standard."* **IEEE International Workshop on Metrology for Industry 4.0 IoT** (June 2020).
doi: 10.1109/MetroInd4.0IoT48571.2020.9138236

(ii) Leopoldo Angrisani, Pasquale Arpaia, Francesco Bonavolontà, and Antonella Cioffi; *"Experimental test of ECDSA digital signature robustness from timing-lattice attack"* **IEEE International Instrumentation and Measurement Technology Conference (I2MTC)** (May 2020), pp. 1-6
doi: 10.1109/I2MTC43012.2020.9129144.

Contents

List of Abbreviations

ADC	**A**nalog to **D**igital **C**onverter
AES	**A**dvanced **E**ncryption **S**tandard
ANOM	**AN**alysis **o**f **M**ean
ANOVA	**AN**alysis **o**f **VA**riance
APDU	**A**pplication **P**rotocol **D**ata **U**nit
CPA	**C**orrelation **P**ower **A**nalysis
DES	**D**ata **E**ncryption **S**tandard
DPA	**D**ifferential **P**ower **A**nalysis
ECDSA	**E**lliptic **C**urve **D**igital **S**ignature **A**lgorithm
HD	**H**amming **D**istance
HW	**H**amming **W**eight
IoT	**I**nternet **o**f **T**hings
MAC	**M**essage **A**uthentication **C**ode
NIST	**N**ational **I**nstitute of **S**tandards and **T**echnology
SCA	**S**ide **C**hannel **A**ttack
SNR	**S**ignal to **N**oise **Ratio**
SPA	**S**imple **P**ower **A**nalysis

Introduction

The progress in communication technology favoured the development of a new paradigm, known as *Internet of Things* (IoT), consisting in smart transducers, immersed in our everyday surrounding environment, that acquire data continuously and communicate with each other by wireless connection to support daily tasks [1]. Indeed, the main goal of the IoT is to improve the quality of life and work, by saving time and resources, as well as minimizing human intervention [2]. In the healthcare sector, for example, an IoT medical transducer (i) acquires patient's health data information, (ii) makes basic data processing, (iii) transmits or receives information by means of a wireless local area network or directly to/from the cloud, and (iv) can act as an actuator [3]. The IoT technologies applied to the healthcare improves the shift of health-related services (prevention, diagnosis, and treatment) from hospital to the patient's home [4], favouring reduction of costs [5], real-time updated information on health conditions, personalized and remotely treatments [6], and maximization of therapeutic results [7]. Many IoT medical transducers have been developed for elderly living assistance [8, 9], for people with chronic diseases [10, 11, 12], or in rehabilitation [13, 14]. The IoT technologies have been widely adopted also in fileds as environmental monitoring, transportation, farming, agriculture, payments and smart grid [15, 16, 17, 18, 19, 20]. Owing to the recent progress in 5G wireless technology, new infrastructures will allow the connection of a massive number of IoT devices, encouraging the development of new Internet-based applications. In fact, the faster communication and capacity providing by 5G wireless network will significantly contribute to increase the number of IoT devices and applications [21].

The enormously rising number of intelligent objects around us makes the IoT security more troublesome compared to other networks [22]. Indeed, security threats are not only addressed to servers or computers, but also to the IoT smart transducers which collect vital data to manage our buildings, our production, and our whole life. The impossibility to implement complex security schemes, due to limited resources of IoT transducers based on low-cost microcontrollers, as well as their location in unsupervised environment, open the doors to new types of attacks [23]. The attackers can gain access to sensor nodes to control and physically damage them, or to monitor and eavesdrop the wireless signals. Moreover, they can take advantage of internet connection to attack other devices.

Various security measures have been introduced in the IoT architecture [24], as lightweight cryptographic algorithms and authentication protocols. The formers encrypt the transmitted signals, while the latters identify the devices before communication. The problem of developing cryptographic algorithms for devices with limited resources is still open. In fact, Google has recently disclosed the results of the Project *Adiantum*, an innovation in cryptography specific for

low-end processors, that do not have hardware support for Advanced Encryption Standard (AES), which is the most used cryptographic algorithm [25]. AES is more reliable compared to other solutions (e.g., PRESENT and CLEFIA) for its safety to attacks and its robust security.

Although these algorithms are considered as mathematically secure, their implementation is vulnerable to attacks, due to information leaked by actual computers and microchips. In 1996 and in 1999, Paul Kocher highlighted that attacks based on measurements of execution time [26] and power consumption [27] can reveal sensitive information in spite of cryptographic algorithms. This measurement-based attack is identified as *side-channel attack* (SCA), since it consists in measuring unintended computation effects (so-called *leakages*), including timing, I/O data exchange, power consumption, or electromagnetic emission. These measurements allow to retrieve useful information related to the cryptographic algorithm, and even to disclose the secret key.

For smart transducers sharing the same secret key, the key recovery from one device enables an attacker to eavesdrop or to insert altered messages into the communication of other connected transducers. In this way, the confidentiality and the integrity of the whole system are compromised. This can results in serious security problems. Indeed, in healthcare sector, the lack of confidentiality and integrity can lead to the loss of sensitive patient information and incorrect data introduction causing misdiagnosis or misadministration of drugs. For the above reasons, the design of an IoT transducer requires a deep hardware security threats analysis.

The side-channel attacks are intensively studied by researchers and test laboratories. As for the attacks that have received by far the most attention in recent years, i.e. power analysis attacks, the scientific literature presents (i) methods to improve the attack efficiency [28, 29, 30], (ii) techniques to make cryptographic algorithms robust against the power attacks [31, 32, 33], and (iii) tests, namely *Vulnerability Assessments*, to evaluate the robustness of IoT devices respect to the side-channel attacks. In fact, on the one hand, the very appealing sensitive and personal information encourages attackers to break a cryptographic system in order to make profits, on the other hand, the security offered by such systems must be increased and tested in order to ensure confidentiality.

In this thesis all the aspects related to the security of IoT devices have been faced. In particular, after a first chapter aimed to introduce the cryptography and side-channel attacks, the subsequently chapters face different problems related to the IoT security. The problems faced in the Chapters are as follows:

- Chapter 1: **Introduction to cryptography**. Firstly, an overview on the main goals of the cryptography are illustrated, then, the focus is moved to side-channel attacks which attempts to break the cryptographic algorithms by discovering the secret key of cryptographic algorithms.

- Chapter 2: **Power attacks on IoT tranducers**. The vulnerability of the mostly used cryptographic algorithm in Internet of Things, the advanced encryption standard (AES), is proved experimentally by measuring the chip power consumption.

- Chapter 3: **Optimization of power attacks**. A novel optimization technique for power analysis-based attacks is proposed by relying on *fractional experimental design*.
- Chapter 4: **Countermeasures against power attacks**. A method for the Vulnerability Assessment of Internet of Things (IoT) transducers cybersecurity at varying the power countermeasures is proposed.
- Chapter 5: **Timing attack on IoT transducers**. An experimental test of robustness to timing attack is reported for the asymmetric cryptographic algorithm most used in Internet of Things framework, i.e. the Elliptic Curve Digital Signature Algorithm (ECDSA).

Chapter 1

Elements of cryptography

In this chapter, an overview on the main goals of the cryptography is presented. The various paradigms used to secure a communication channel are briefly explored. Then, the focus is moved on side-channel attacks which attempts to break the cryptographic algorithms by discovering the secret key of cryptographic algorithms. The recovery of the secret key poses severe security problems. For smart transducers sharing the same secret key, the key recovery from one device enables an attacker to eavesdrop or to insert altered messages into the communication of other connected transducers. These considerations aim to introduce the context of the present work and to give some theoretical basis for the next chapters.

1.1 Fundamentals

Cryptography is the practice and study of techniques for securing communications between two parties over an insecure communication channel [34]. *Cryptoanalysis* studies techniques for breaking the cryptographic systems. The *Cryptology* is the term used to refer to both cryptography and cryptanalysis in a general sense as the study of techniques for data communication and data storage in a secure way.

Suppose a sender wants to send a message to a receiver. It has become customary to call these two parties Alice and Bob [35]. Also suppose that Alice chooses an insecure communication channel, such as a computer network or a telephone line. A malicious third party, commonly known as Eve, could intercept and read the message. Moreover, the adversary might be able to modify the message during the transmission. The main goal of the cryptography is to provide methods, techniques, algorithms, and protocols to prevent the eavesdropping or the alteration of the exchanged messages, and to ensure the *confidentiality* of the information, i.e. only the recipient of the message has access to the messages.

The *encryption* and *decryption* methods are largely employed to achieve the goal of confidentiality. Encryption methods transform the message, known as *plaintext*, into an alternative and unintelligible form, known as *ciphertext*, while decryption methods return back the plaintext from the ciphertext. Therefore, before sending the message to Bob, Alice encrypts the plaintext, obtaining the ciphertext, and sends the encrypted message to Bob. Bob obtains the plaintext by decrypting the ciphertext. The adversary Eve may still intercept the message, but

she is not able to obtain the original information from the ciphertext. The procedure of encryption and decryption over a communication channel is shown in Fig. 1.1.

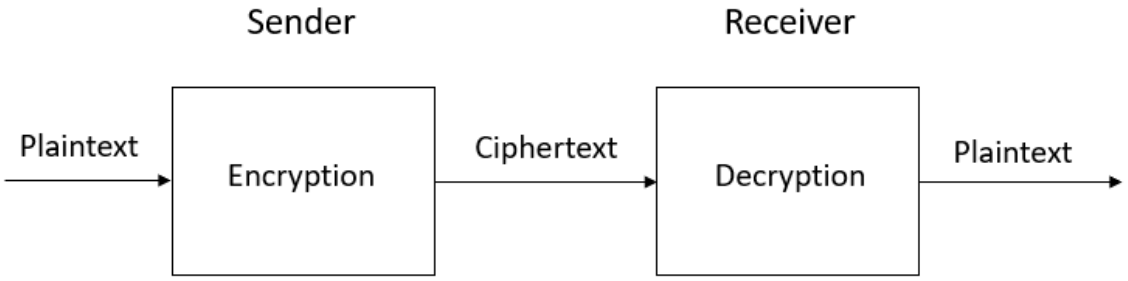


FIGURE 1.1: The process of encryption and decryption.

The encryption and decryption methods use a secret parameter, known as *key*, for transforming the input data. Only those who have access to the secret key can decrypt the message. Therefore, the key is known only to the two parties of the communication. In this way, only authorized parties can decipher a plaintext from the ciphertext and access to the original information. The reason to keep the key secret is based on the ***Kerckhoffs' principle***, which states that the entire security of an algorithm should be based only on the confidentiality of the key, not on the confidentiality of the actual algorithm. This principle can be interpreted more generally: in any security system, a secret is a potential source of breakage and should therefore be kept to a minimum and should be easily replaceable. Since the algorithm is public and known by everyone, the security of the system is relied on the secrecy of the key.

The use of encryption for secure communication has ancient origins. The first example of encryption is the Caesar cipher, used by Julius Caesar to sent messages to his general. In Caesar cipher all the letters in the plaintext are shifted by three (key) to produce the ciphertext. Prior to the early 20th century, cryptography was mainly concerned with linguistic and lexicographic patterns. Indeed, the encryption was based on *transposition cipher* and *substitution cipher*. The former reorganizes the letters in a message, while the latter replace one group of letters with another one. Since the 20th century, the cryptography has made extensive use of mathematics, including aspects of information theory, computational complexity, statistics, combinatorics, abstract algebra, number theory, and finite mathematics. Moreover, the encryption combines a variety of techniques, such as substitution and transposition, to reduce the statistical dependence between the plaintext and the ciphertext. Extensive open academic research into cryptography is relatively recent. It began in the mid-1970s, when the algorithm Lucifer developed by Horse Feistel at IBM was officially adopted, with the name Data Encryption Standard (DES) [36], as the first standard encryption algorithm. Since then, cryptography was widely employed in communications, computer networks, and computer security.

In modern era, the advent of computers and other communication devices introduced new problems for the communication security. This aspect addressed

the cryptography to new goals. Apart from the data confidentiality, the other main purposes of the cryptography are:

- *Data integrity*. The receiver of a message should be able to detect if the original message, sent by the sender, was modified during the transmission, either accidentally or deliberately.
- *Authentication*. The receiver of a message should be able to verify the identity of the sender from the message received.
- *Non-repudiation*. The sender should not be able to deny sending of a message.

A message written on a paper ensures the data integrity, authentication, and non-repudiation as the mean prevents manipulation and the handwritten personal signature allows to uniquely identify the sender. A message sent over a digital communication channel can be altered by changing some bytes. The cryptography provides algorithms and techniques to achieve the goals of data integrity, authentication, and non-repudiation over insecure communication channel.

The solution offered by cryptography for authentication and non-repudiation is the *digital signature*. The digital signature process is shown in Fig. 1.2. This technique is based on two algorithms: *sign* and *verify*. The sender uses the sign algorithm to produce a digital signature and sends the message with the signature to the receiver. The receiver implements the verification algorithm to assess the validity of the signature. The digital signature algorithms are based on a key-pair: one secret and one public. If Alice wants to sign a message m, she applies the algorithm of sign using her secret key sk and gets the signature $Sign(sk, m)$. Bob checks the validity by testing whether:

$$Verify(pk, s, m) = ok$$

where pk is the Alice's public key, s is the received signature, m the message. A valid digital signature gives a recipient very strong reason to believe that the message was created by a known sender (authenticity). Once the authenticity of the recipient is ascertained, the latter can not deny sending the message (non-repudiation).

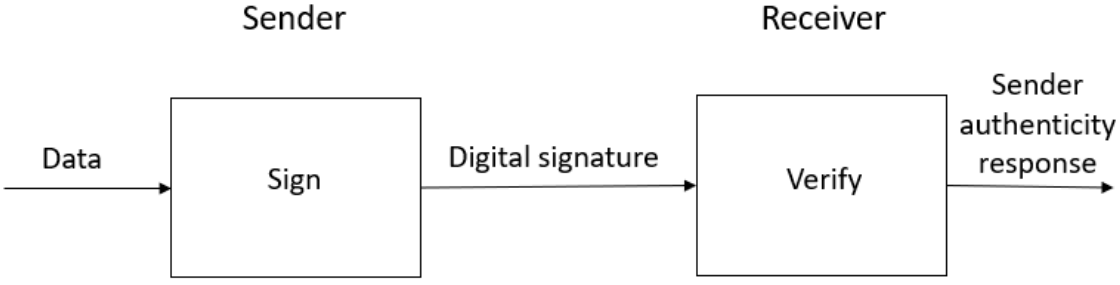


FIGURE 1.2: The process of digital signature.

The *message authentication code* (MAC) is used to provide data integrity. The technique is based on a process of generation and verification of a tag, called

MAC, as shown in Fig. 1.3 and of a secret key that is shared by sender and receiver. The MAC is generated from the message by means of a MAC generation algorithm, and it is sent to the receiver in addition to the message. The recipient may test the integrity of the message by applying the MAC generation algorithm on the message and checking the equality between the generated and received MAC m^*:

$$MAC(k, m) = m^*$$

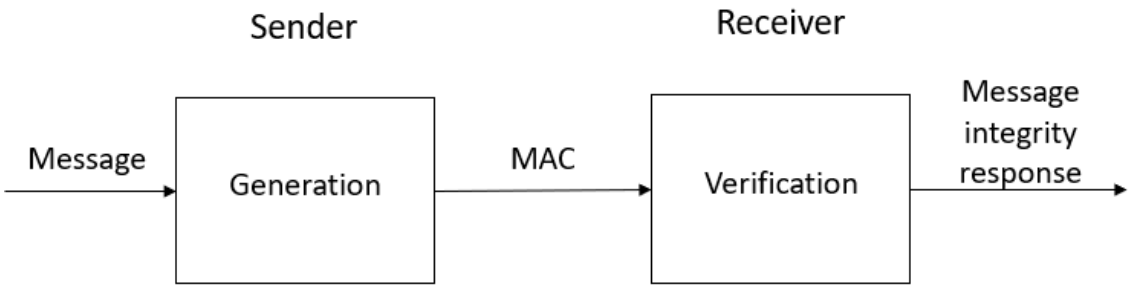


FIGURE 1.3: The process of message authentication code.

The techniques of encryption, decryption, digital signature, and message authentication are implemented by means of cryptographic algorithms. They are mathematical algorithms that use a secret parameter, known as *key*, for transforming the input data. The key parameter can be a single key or a key-pair, and it determines the classification of the cryptographic algorithms in two categories: *symmetric-key cryptography* and *asymmetric-key cryptography*.

1.1.1 Symmetric-key cryptography

The expression *symmetric-key* cryptography refers to cryptographic algorithms that use the same key k for encryption and decryption. The expression *secret-key cryptography* is often used to refer to this typology of cryptography for its property to keep the key secret between the parties of a communication.

The encryption and decryption algorithms are publicly known. Anyone knowing the key can decrypt the ciphertext and access the original information. Therefore, the key has to be kept secret by the two communicating parties. If Alice and Bob want to use the symmetric-key encryption scheme, they first have to exchange the key. Then, Alice encrypts plaintext using the encryption algorithm and the key. The ciphertext is sent to Bob. Bob uses the decryption algorithm and the key to recover the plaintext from the ciphertext. The scheme of symmetric-key encryption is shown in Fig. 1.4

The symmetric-key encryption algorithms are very fast in hardware and software. Therefore, they are very well-suited to the encryption of large amounts of data and in resource-constrained devices, as the Internet of Things devices. However, these algorithms suffer of a secure key distribution. Indeed, the key is exchanged over an insecure communication channel. Anyone intercepting the key in transit can later read or modify the information encrypted. The key exchange

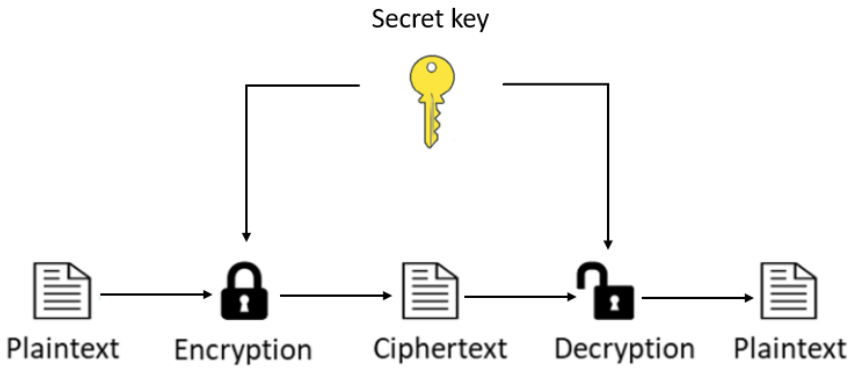


FIGURE 1.4: Block diagram of symmetric-key cryptography.

problem was solved with the discovery of the revolutionary concept of public-key cryptography. Nowadays, public-key cryptography is widely employed to exchange the key in a secure way.

The symmetric key algorithms are generally distinguished between *block ciphers* and *stream ciphers*. The encryption function of a block cipher processes plaintexts of fixed length. If the plaintext length exceeds the block length of a block cipher, various modes of operation are used, acting as a stream cipher, character by character. A stream cipher operates on streams of plaintext. This allows to encrypt plaintext strings of arbitrary length. Block ciphers are the most used. Examples of block ciphers include Data Encryption Standard (DES) [36] and Advanced Encryption Standard (AES) [25].

Advanced Encryption Standard

In 1997, the US National Institute of Standards and Technology (NIST) requested the international cryptographic community to submit suggestions for a successor to data encryption standard. Indeed, due to technological progress, private individuals became able to mount successful brute-force attacks on the DES algorithm, making it insecure. In 1998, among 15 candidates for the Advanced Encryption Standard (AES), the Rijndael algorithm, developed by the Belgian cryptologists Joan Daemen and Vincent Rijmen, was selected as the successor to the DES algorithm.

The Advanced Encryption Standard [25] is a symmetric-key cryptographic algorithm. It transforms a clear input message (*plaintext*) in an unintelligible output message (*ciphertext*) by means of repeatedly transformation rounds and a secret key. The AES is a block encryption algorithm with a block length of 128 bits. The input message is 128-bits fixed-length, but the key length can be of 128, 192, or 256 bits. The various AES types are labeled as AES-128, AES-192, and AES-256 respectively. The key length determines the number of transformation rounds of the plaintext: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Fig. 1.5 shows the flow diagram of the AES algorithm. For each transformation round, a *round key* obtained by the secret key is used. The production of all the round keys (*key expansion*) is implemented with a key scheduling

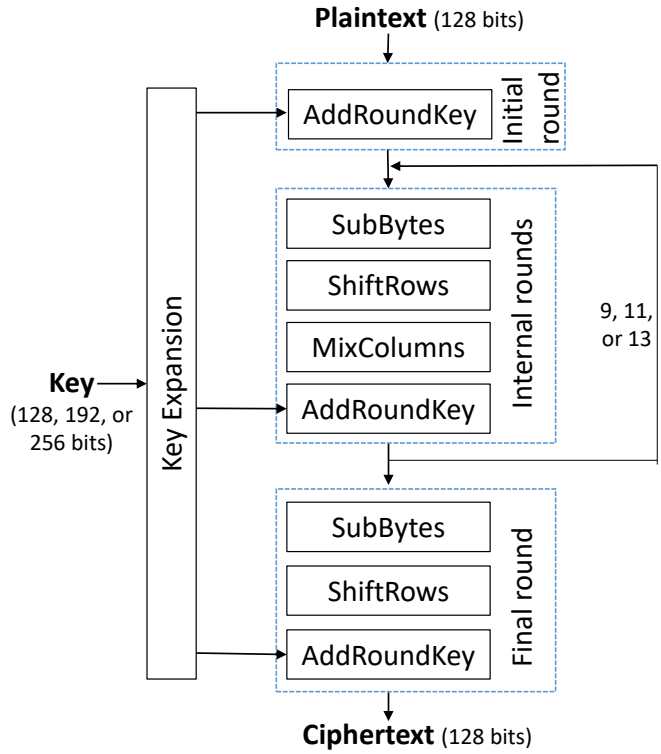


FIGURE 1.5: Flow diagram of Advanced Encryption Standard (AES) algorithm.

algorithm at the beginning. Then, the input message, represented as a byte array (*state*), is repeatedly transformed. An initial round performs an operation of *AddRoundKey* computing an exclusive-OR between the state and the round key. Then, 9, 11, or 13 internal rounds composed by *SubBytes*, *ShiftRows*, *MixColumns*, and *AddRoundKey* operations are computed. The *SubBytes* transforms each byte of the state array by a non-linear function (*SBox*), often pre-computed and stored in a lookup table. The *ShiftRows* operates on each row of the state, by shifting the bytes cyclically by a proper offset. In the *MixColumns*, the bytes of each column of the state are combined by an invertible linear transformation. The algorithm ends with a final round, where only *SubBytes*, *ShiftRows*, and *AddRoundKey* operations are computed.

1.1.2 Asymmetric-key cryptography

The *asymmetric-key cryptography* uses two different keys for encryption and decryption. The expression *public-key cryptography* is often used. The revolutionary idea of public-key cryptography was introduced by Diffie and Hellman in 1970 [37]. It is based on a pair of keys: *secret key* and *public key*. The secret key is known only to the owner and a public key is known to everyone. The public-key cryptography can be used for encryption and for digital signature.

Suppose that Bob has a key pair (pk, sk) and Alice wants to encrypt a message m for Bob. She computes the ciphertext by applying the encryption algorithm

with Bob's public key pk. Bob uses the decryption algorithm and his secret key sk to recover the plaintext from the ciphertext. The scheme of public-key encryption is shown in Fig. 1.6. Since only Bob knows the secret key, he is the only one who can decrypt the message.

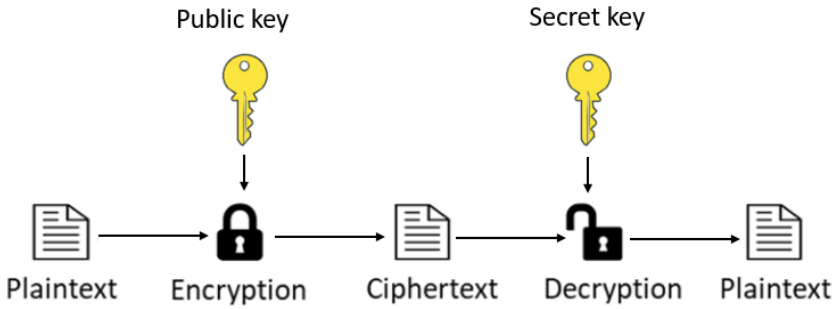


FIGURE 1.6: Block diagram of asymmetric-key cryptography.

Although the public-key cryptography can be used to encrypt a message, these schemes are not suitable for encrypting large amounts of data as they are less efficient and slow. The advantage of public-key cryptography concerns the distribution of key for symmetric-key encryption. Before sending the symmetric-key to Bob, Alice encrypts the key using the Bob's public key. Bob decrypts the key with his private key. After the key is exchanged securely, Alice and Bob can send messages to each other with symmetric-key encryption.

Another application of public-key cryptography is the digital signature, used to provide authentication and non-repudiation. In this case, the secret key is used for signing the message and the public key to verify it. If Alice wants to sign a message for Bob, she uses her secret key sk. Bob can verify the signature by using Alice's public key pk.

The most used cryptographic algorithms are Diffie–Hellman and RSA for key distribution, and Digital Signature Algorithm and RSA for digital signatures.

Elliptic Curve Digital Signature Algorithm

Elliptic Curve Digital Signature Algorithm (ECDSA) implement the Digital Signature Algorithm (DSA) using the elliptic-curve cryptography (ECC).

Elliptic-curve cryptography is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields [38]. The use of elliptic curves in cryptography was suggested independently by Neal Koblitz [39] and Victor S. Miller [40] in 1985. The major advantage of using ECC is the smallest key size, that makes elliptic curve based algorithms suitable for resources constrained devices. Moreover, the ECC offers high level of security based on Elliptic Curve Discrete Problem (ECDLP).

Elliptic Curve Digital Signature Algorithm (ECDSA) was first proposed in 1992 by Scott Vanstone as an elliptic curve version of the Digital Signature Algorithm (DSA). Then, it was accepted as a standard by ISO (ISO 14888-3), ANSI

(ANSI X9.62), IEEE (IEEE 1363-2000), and FIPS (FIPS 186-2). This algorithm computes a pair of numbers (r, s), called *signature*, for a given message m. The calculation of the value r is based on an *ephemeral key k*, called *nonce*, which changes in every signature generation. This means that signatures of the same message are different.

ECDSA consists of three procedures [38]: *key generation, signature generation, and signature verification*. Consider the ECDSA domain parameters described as $D = (q, FR, a, b, G, n, h)$, where q is the field size, FR is an indication of the representation used for the elements of F_q, a and b are two field elements in F_q which define the equation of the elliptic curve E, G is the generator point, n is the order of G, and h is the cofactor.

Key generation. A key pair, associated with a particular set of EC domain parameters $D = (q, FR, a, b, G, n, h)$, is computed as follow:

- Select a random or pseudorandom integer d in the interval [1, n−1].
- Compute $Q = dG$.
- Public key is Q; Private key is d.

Signature generation. A message m, with domain parameters $D = (q, FR, a, b, G, n, h)$ and associated key pair (d, Q), is signed as follow:

- Select a random or pseudorandom integer, called *nonce*, k, $1 \leq k \leq n - 1$
- Compute $kG = (x_1, y_1)$
- Compute $r = x_1$ mod n. If r = 0 then go to step 1
- Compute k^{-1} mod n
- Compute $e =$ SHA1(m), where SHA is a *secure hash algorithm* used to derive the message digest
- Compute $s = k^{-1}(e + dr)$ mod n. If s = 0 then go to step 1
- The signature for the message m is (r, s)

Signature verification. A signature (r, s), obtained on a message m, with domain parameters D = (q, FR, a, b, G, n, h) and associated public key Q, is verified as follow:

- Verify that r and s are integers in the interval [1, n-1]
- Compute SHA1(m)
- Compute $w = s^{-1}$ mod n
- Compute $u_1 = ew$ mod n and $u_2 = rw$ mod n
- Compute $(x_1, y_1) = u_1G + u_2Q$. If $(x_1, y_1) = O$ then the signature is invalid
- Accept the signature if and only if x_1 mod n = r

1.1.3 Security

The primary goal of a cryptographic system is to kept the messages exchanged over a communication channel secret. The encryption and decryption algorithms are public and known to all, so the security of a cryptographic system depends by the secrecy of the keys (***Kerckhoffs' principle***). Several attempts have been made during the years to break a cryptosystem, discovering the secret key of the cryptographic algorithms. Discovering the secret key means accessing to the exchanged messages. The process of recovering the secret key is known as *attack*, while the science of studying the techniques to break a cryptosystem is known as *Cryptoanalysis*.

The strength of cryptographic device against the attacks is typically analyzed to established their security level. The assessment procedure is conducted by testing laboratories that certified the device if it is considered safe. The security level is assessed on the basis of the effort, in terms of computational resources and time, required to break the cryptosystem. The following definitions of security are widely used [41]:

- Computational security. This means that an algorithm that tries to break the cryptosystem is computational infeasible, i.e. it would take a long time using currently available computational resources.
- Provable security. This means that breaking the cryptosystem is at least difficult as solving the given hard problem.
- Unconditional security. This means that the cryptosystem does not be broken because there is not enough information available to the adversary.

The security in cryptography depends also by the *attack model*, i.e. the information available to the attacker. Attack methods are traditionally classified in:

- Ciphertext-only attack. The attacker has access to several ciphertexts produced using the same secret key.
- Known plaintext attack. The attacker has access to several plaintexts and their corresponding ciphertexts.
- Chosen plaintext attack. The attacker is able to choose the plaintexts and obtain the related ciphertexts.
- Adaptively chosen plaintext attack. The attacker is able to choose parts of several plaintexts and obtain the related ciphertexts, where the choice of each plaintext depends on all previously obtained ciphertext.
- Chosen ciphertext and adaptively chosen ciphertext attacks. Similar to the chosen plaintext attacks, but with the roles of plaintext and ciphertext reversed.

A chosen plaintext or chosen ciphertext attack provides the adversary with more information than a known ciphertext attack. Therefore, they result stronger attack model than known ciphertext attack.

1.2 Side-channel attacks

In 1996 and in 1999, Paul Kocher succeed to reveal sensitive information of cryptographic algorithms by means of measurements of execution time [26] and power consumption [27]. He was the pioneer of a typology of measurement-based attacks, identified as side-channel attacks (SCAs). Side-channel attacks exploit weaknesses in the physical implementation of an algorithm in order to discover secret information of the cryptographic algorithm. Testing laboratories use side-channel attacks to test the protection of a cryptographic device. Researchers apply side-channel attacks to test new countermeasures or try to invent new attacks. Finally, there are attackers who actually use side-channel attacks to find the secret key within a cryptographic device in order to break it and benefit from it.

There are two categories of side-channel attacks, passive and active attacks. The former recovers the secret key by analyzing the observable phenomena from a cryptographic implementation, while in the latter the attacker modifies the execution of the cryptographic algorithm. Examples of passive side-channels involve power [27], electromagnetic radiation [42], timing [26] and sound. Active side-channel attacks are fault injections [43]. A scheme of the side-channel attack classification is shown in Fig. 1.7.

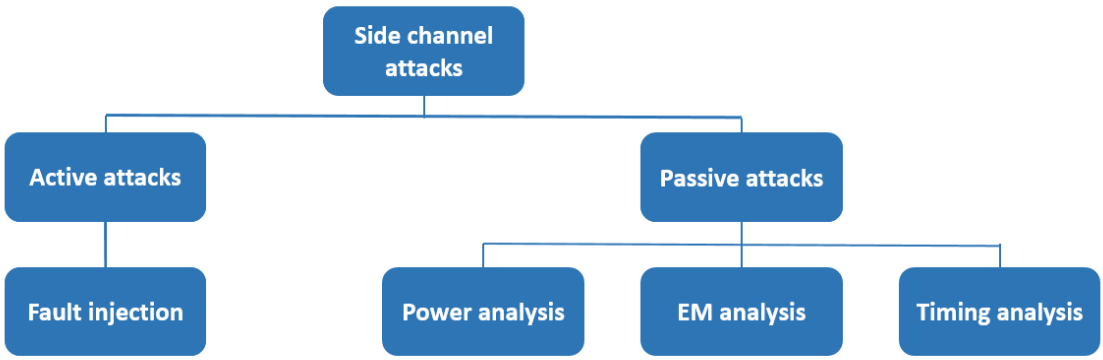


FIGURE 1.7: Classification of side-channel attacks.

The passive side-channel attacks consist of (i) measuring unintended computation effects (so-called leakages) from the cryptographic device, and (ii) applying statistical techniques to recover the secret key. Most prominent side-channel attacks include timing attacks, power analysis attacks, and electromagnetic emission attacks. Timing attacks exploit the variations in the execution times of a cryptographic algorithm that depend on different inputs. Power analysis attacks exploits the variations in the power consumption of a cryptographic device that depend on the processed data and the performed operation. Electromagnetic attacks are similar to the power analysis attack, but they exploit the variations in the electromagnetic emissions that depend on the processed data. In Fig. 1.8, a block diagram of a typical side-channel attack on an Internet of Things (IoT) device is reported. Through the main channel, the IoT transducer is fed by the data.

The cryptographic functions, for example the Advanced Encryption Standard, using the secret key produce the encrypted data. In the meantime, an unintended computation effect, such as execution time, power consumption, electromagnetic emission, light, or heat, is acquired through the side-channel and statistically analyzed to discover the secret key.

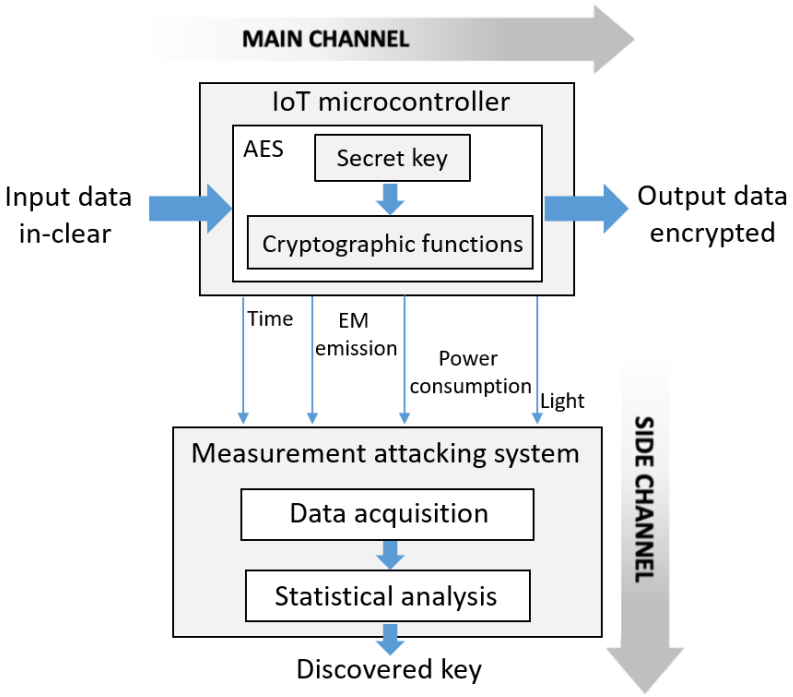


FIGURE 1.8: Block diagram of side-channel attacks.

Such attacks turn out to be very powerful as they are non-invasive, i.e. the attacker can only observe or control the accessible interfaces of the devices, and do not require expensive and invasive equipment [44]. Currently, the wide availability of commercial off-the-shelf (COTS) IoT devices pose severe security threats. Indeed, an attacker has physical access to these devices and easily manages to acquire leakages, as timing, power consumption, or electromagnetic emission. For transducers sharing the same secret key, the key recovery from one device enables an attacker to eavesdrop or to insert altered messages into the communication of other connected transducers. In this way, the confidentiality and the integrity of the whole system are compromised.

1.2.1 Power analysis attacks

The power analysis attack is a side-channel attack that exploits the variation in the power consumption of a cryptographic device to reveal the secret key. This type of attack benefits from the correlation between the level of power consumption and cryptographic operations of the device. Indeed, the instantaneous power consumption of a cryptographic device depends on the processed data and on the performed operations. Therefore, the data-dependency and the operation-dependency are exploited.

Originally, there were two types of power analysis attacks including simple power analysis (SPA) and differential power analysis (DPA). As research progressed in the area of side-channel attack, a third attack has been proposed in literature, which is formally referred to correlation power analysis (CPA). Finally, a more recent power attack was developed and identified as scatter attack. The above mentioned power attacks require the physical access to the device for monitoring the power consumption. In the SCA research area it is common to use the term power consumption although the actual quantity is a voltage. Moreover, the expression *power trace* is used for the value-discrete voltage waveform. The power analysis attacks differ in the method used to reveal the secret key. On the base of the adopted method, they can be classified in two main categories: *visual inspection-based* and *statistical analysis-based*. In the visual inspection-based, a single trace can be used to extract the secret key, while in statistical analysis-based, statistical operation have to be computed. The attacker uses a consumption model to predict the power consumption on the base of an hypothetical intermediate value. The predicted consumption value and measured one are analyzed in a statistical way to recover the secret key. The block diagram of statistical analysis-based attacks is reported in Fig. 1.9. The SPA belongs to the first category while the DPA, CPA, and scatter to the second one.

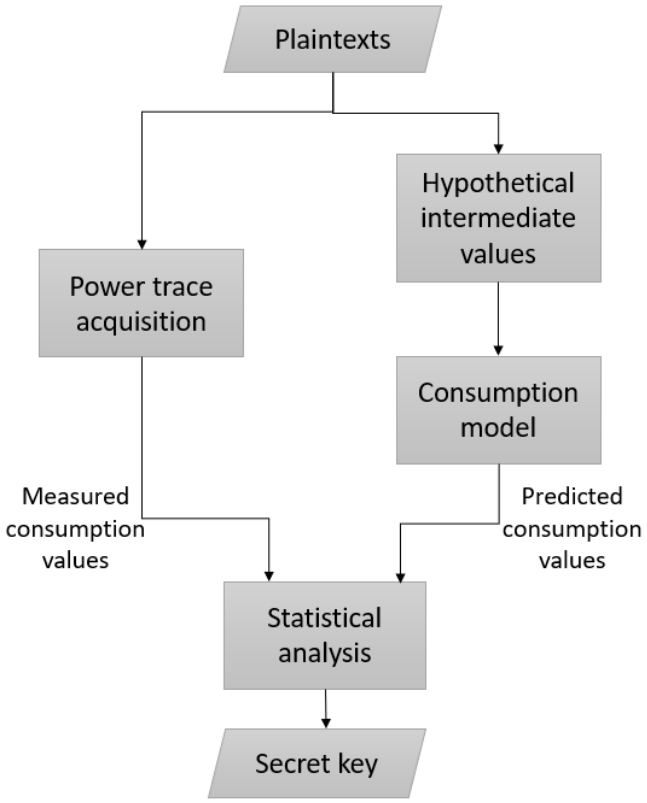


FIGURE 1.9: Block diagram of statistical-based power analysis attacks.

Power model

In power analysis attacks, the data values that are processed by the attacked device have to be mapped to power consumption values. The estimated power value allows to construct the statistical analysis of the power attacks. However, it is not important to know the absolute values of the power consumption but the variations related to the secret data. Two generic models are used for power analysis attacks: *Hamming-weight model* (HW) and *Hamming-distance model* (HD).

- Hamming-weight model: This power model assumes that the power consumption is proportional to the number of bits that are set in the processed data values. It is applied when the attacker does not know the consecutive data values.
- Hamming-distance model: the idea of this model is to describe the power consumption of the circuit in a time interval through the number of $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions that occur in the circuit. Formally, the Hamming-distance of two values v_0 and v_1 corresponds to the Hamming weight of $v_0 \oplus v_1$. The Hamming weight model corresponds to the number of bits that are set to one, hence $HW(v_0 \oplus v_1)$ corresponds to the number of bits that differ in v_0 and v_1. This power model is suitable to describe the power consumption of data bus.

Simple power analysis

Simple Power Analysis (SPA) is defined in [27] as "a technique that involves directly interpreting power consumption measurements collected during cryptographic operations". In other words, the key can be derived by analyzing the power trace directly. The goal of SPA is to reveal the secret key by a small number of power traces. In the most extreme cases, a single trace may be used to reveal the secret. The terms *single-shot SPA* attack and *multiple-shot SPA* attack are used to distinguish between two possible SPA implementations on the base of the number of power traces employed. In single-shot SPA attacks, only one power trace can be acquired, while in multiple-shot SPA attacks, a set of traces can be collected. The set of power traces can be obtained by providing the same plaintext. The advantage offered by such a technique is the reduction of the noise by computing the mean of traces.

The possibility to reveal the secret key directly from a power trace requires a detailed knowledge about the cryptographic algorithm implemented on the device under attack. This aspect makes the simple power analysis attack quite challenging in practice. Anyway, the visual inspection can provide useful information for a different attack. Indeed, it allows to identify the power consumption to be acquired referring to the operation of interest. The acquired traces can be exploited by more sophisticated technique attacks as the differential power analysis.

Differential power analysis

The Differential Power Analysis (DPA) attacks try to reveal the secret key of a cryptographic device taking advantage by a large number of power traces acquired while the device encrypts or decrypts different messages. The DPA attacks exploit the data dependency of the power consumption. The power consumption is analyzed at a fixed moment of time as a function of the processed data.

The differential power analysis implementation is based on two main phases: *power trace acquisition* and *statistical analysis*. In the acquisition phase, an attacker provides several plaintexts to the cryptographic device and acquires the related ciphertexts and power consumption. This phase returns a power trace vector, where each trace $\boldsymbol{t_i'} = (t_{i,1}, ..., t_{i,T})$ corresponds to an input data d_i in the input vector $\boldsymbol{d} = (d_1, ..., d_D)'$. The length T of the trace depends on the sampling rate and the acquisition time window. D and T are fixed at the beginning of the attack. The acquired data are the input of the next statistical analysis. The statistical analysis is conducted with a "*divide and conquer*" strategy, that is, a single byte of the key is discovered at a time. For each hypothesis on the key byte k, the *hypothetical intermediate value* $v_{i,k} = f(d_i, k)$ is calculated, with d_i an input data block for $i = 1, ...D$ and f a dependent-key function of the cryptographic algorithm. A *power model*, based on a single bit value in $v_{i,k}$, is used to predict the power consumption of hypothetical keys. Typically, the least significant bit $LSB(v_{i,k})$ is used. In particular, the power trace $\boldsymbol{t_i}$, corresponding to the data value d_i, is labelled as $\boldsymbol{t_{i,0}}$ if the target bit $LSB(v_{i,k})$ is equal to zero, otherwise as $\boldsymbol{t_{i,1}}$. In this way, the power traces are divided in two groups $G_{0,k}$ and $G_{1,k}$.

$$G_{0,k} = \{\boldsymbol{t_i} | LSB(v_{i,k}) = 0\} \qquad G_{1,k} = \{\boldsymbol{t_i} | LSB(v_{i,k}) = 1\} \tag{1.1}$$

A mean vector $\boldsymbol{A_{g,k}}$ is calculated for each group. Each element A of these vectors is computed as:

$$A_{g,k,p} = \frac{1}{N_g} \sum_{i=1}^{N_g} t_{i,g,p} \tag{1.2}$$

where the subscripts $g \in \{0, 1\}$ identifies the group, k refers to the key byte hypothesis, p is the position of the sample in the power trace vector t, and N_g specifies the number of power traces belonging to the group. The difference between the two mean vectors:

$$\boldsymbol{DPA_k} = \boldsymbol{A_{1,k}} - \boldsymbol{A_{0,k}} \tag{1.3}$$

corresponds to a pattern for the k hypothesis of the key byte. The basic operations of DPA statistical analysis for one key byte hypothesis are shown in Fig. 1.10. At the end, a number of patterns equal to the hypothesis of the key byte is obtained.

When the key byte hypothesis is correct, the hypothesis on $LSB(v_{i,k})$ coincides with the real value assumed by the cryptographic algorithm and the two sets in (1.1) are split in a right way. This means that $\boldsymbol{A_{1,k}}$ and $\boldsymbol{A_{0,k}}$ have very different values. Therefore, their difference, in (1.3), will be grater than zero. Contrarily, when a wrong hypothesis on the key byte is made, the traces in two subsets $G_{0,k}$

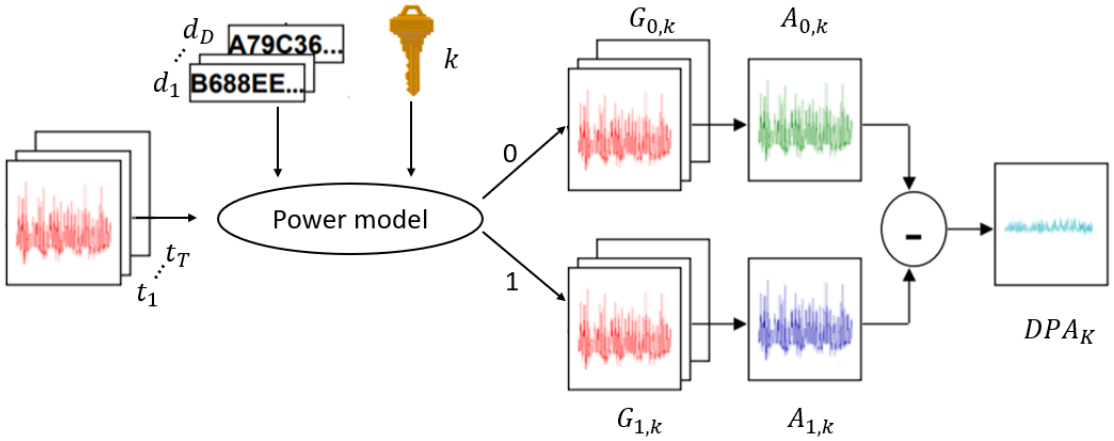


FIGURE 1.10: Basic operations of the differential power analysis statistical analysis for one key byte hypothesis k.

and $G_{1,k}$ are statistically uncorrelated and their means are approximately equal: $\boldsymbol{A_{1,k} \approx A_{0,k}}$. This means that the difference in (1.3) is approximately zero. Therefore, high values in $\boldsymbol{DPA_k}$ maximizes the likelihood that the k key byte hypothesis coincides with the secret key byte. The secret key is discovered by repeating this procedure for all the key bytes.

Correlation power analysis

The Correlation Power Analysis (CPA) is a statistical analysis-based power analysis attack, i.e. it consists of two main phases: trace acquisition and statistical analysis. The trace acquisition phase is conducted as in Section 1.2.1 and it returns a power trace vector, where each trace $\boldsymbol{t_i'} = (t_{i,1}, ..., t_{i,T})$ corresponds to an input data d_i in the input vector $\boldsymbol{d} = (d_1, ..., d_D)'$. Also in CPA, the statistical analysis phase is conducted with a divide and conquer strategy. For each key byte, a matrix of the *hypothetical intermediate values* is computed:

$$\boldsymbol{V} = \begin{pmatrix} v_{0,0} & v_{0,1} & \cdots & v_{0,K-1} \\ v_{1,0} & v_{1,1} & \cdots & v_{1,K-1} \\ \vdots & \vdots & \ddots & \vdots \\ v_{D-1,0} & v_{D-1,1} & \cdots & v_{D-1,K-1} \end{pmatrix}$$

being

$$v_{i,k} = f(d_i, k), \text{ with } i = 0, ..., D-1 \text{ and } k = 0, ..., K-1$$

where f is a dependent-key function of the cryptographic algorithm, d is an input data block, k refers to the key byte hypothesis, D corresponds to the number of input data blocks and K to the number of key byte hypothesis. The matrix $\boldsymbol{V}$ is mapped to a matrix $\boldsymbol{HP}$ of *hypothetical power consumption values* by means of a

power model:

$$\boldsymbol{HP} = \begin{pmatrix} h_{0,0} & h_{0,1} & \dots & h_{0,K-1} \\ h_{1,0} & h_{1,1} & \dots & h_{1,K-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{D-1,0} & h_{D-1,1} & \dots & h_{D-1,K-1} \end{pmatrix}$$

Typically, the *Hamming weight model* is used, consisting in evaluating the number of bits equal to one in $v_{i,k}$. The power traces acquired can be written as rows in a matrix $\boldsymbol{RP}$, whose dimensions are $D \times T$, with T the length of the traces.

$$\boldsymbol{RP} = \begin{pmatrix} t_{0,0} & t_{0,1} & \dots & t_{0,T-1} \\ t_{1,0} & t_{1,1} & \dots & t_{1,T-1} \\ \vdots & \vdots & \ddots & \vdots \\ t_{D-1,0} & t_{D-1,1} & \dots & t_{D-1,T-1} \end{pmatrix}$$

being $t_{i,j}$ the j-th sample of the i-th power trace. Correlation coefficients are used to determine the linear relationship between the columns of $\boldsymbol{HP}$ and $\boldsymbol{RP}$ matrix and are collected in the matrix $\boldsymbol{R}$:

$$\boldsymbol{R} = \begin{pmatrix} r_{0,0} & r_{0,1} & \dots & r_{0,T-1} \\ r_{1,0} & r_{1,1} & \dots & r_{1,T-1} \\ \vdots & \vdots & \ddots & \vdots \\ r_{K-1,0} & r_{K-1,1} & \dots & r_{K-1,T-1} \end{pmatrix}$$

being:

$$r_{k,j} = \frac{\sum_{i=0}^{D-1}(h_{i,k} - \bar{h}_k) \cdot (t_{i,j} - \bar{t}_j)}{\sqrt{\sum_{i=0}^{D-1}(h_{i,k} - \bar{h}_k)^2 \cdot \sum_{i=0}^{D-1}(t_{i,j} - \bar{t}_j)^2}}$$

where $\bar{h}_k$ is the mean value of the k-column of $\boldsymbol{HP}$ and $\bar{t}_j$ is the mean value of the j-column of $\boldsymbol{RP}$. Each row $\boldsymbol{r_k}$ of $\boldsymbol{R}$ is a pattern to one key hypothesis, allowing to understand the byte hypothesis with the highest likelihood to be the actual secret byte. The basic operations of the correlation power analysis are shown in Fig. 1.11.

When the key byte hypothesis is correct, the hypothesis on $v_{i,k}$ coincides with the real value assumed by the cryptographic algorithm and the k-th columns of $\boldsymbol{HP}$ and the j-th of $\boldsymbol{RP}$ are correlated. This means that the k-th row of $\boldsymbol{R}$ will have values approximately equal to one. Contrarily, when a wrong hypothesis on the key byte is made, the k-th columns of $\boldsymbol{HP}$ and the j-th of $\boldsymbol{RP}$ are uncorrelated and the k-th row of $\boldsymbol{R}$ will have values approximately equal to zero. Therefore, high values of correlation coefficient in a row maximizes the likelihood that the key byte hypothesis coincides with the secret key byte. The secret key is discovered by repeating this procedure for all the key bytes.

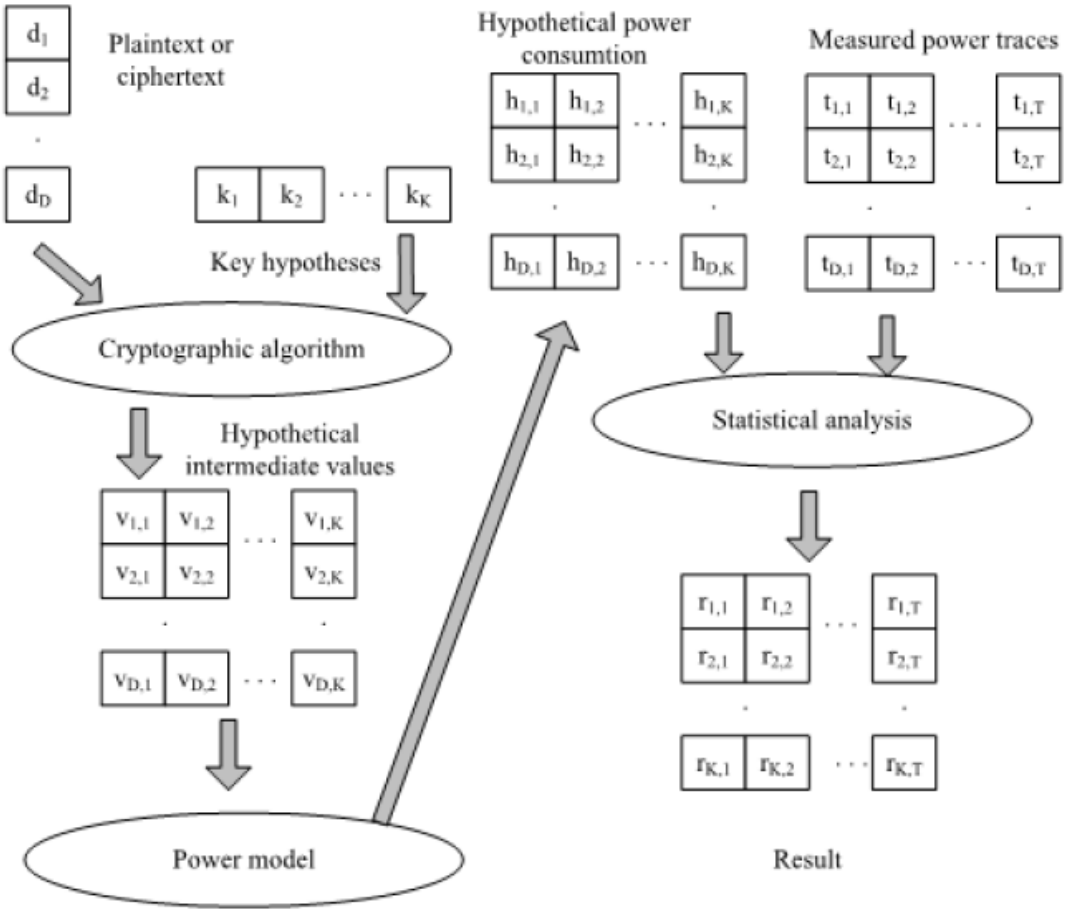


FIGURE 1.11: Basic operations of the correlation power analysis.

Scatter attack

The scatter attack is a recently developed statistical power analysis attack [45]. As the other statistical power analysis attacks, the scatter implementation consists of a trace acquisition phase and a statistical analysis one conducted with a "divide and conquer strategy". The former is carried out as in Section 1.2.1, while the latter represents the measurements points of the power traces in an innovative way. It consists of a distribution representing the number of times each value occurred. A representation of the histograms obtained from the power traces is shown in Fig. 1.12. The created histograms are added to the accumulator $Acc_{(k)(h)}$ for each key byte hypothesis k and the power model value h. The power model is applied to the *hypothetical intermediate value* $v_{i,k} = f(d_i, k)$, with d_i an input data block for $i = 1, ...D$. Once the accumulation step is performed, the corresponding values shall be normalised with the total number of point within the accumulator. Denoting X the random variable related to the measurement, and Y the random variable related to the estimation, the probability density function is computed as $pdf(x) = P(X = x|Y)$.

To discriminate the probability density function related to the correct key from all others for a given power estimation h, different distinguishers may be used.

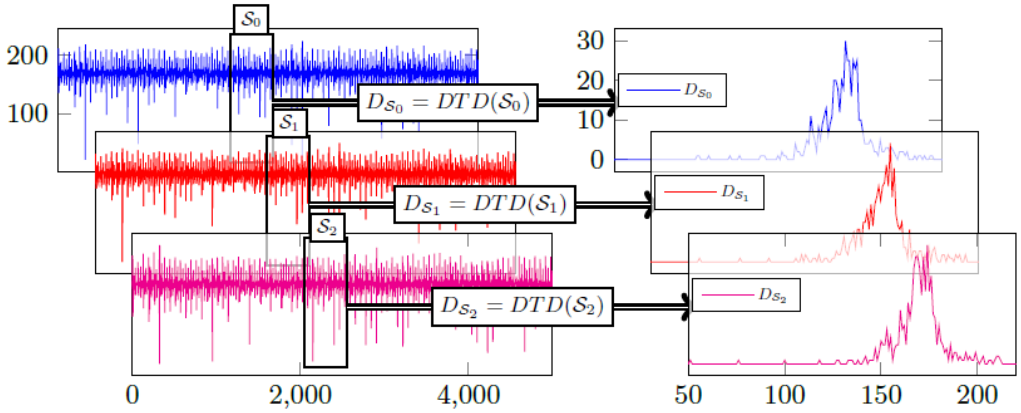


FIGURE 1.12: Illustration of transformation from power traces to histograms (from [45]).

The most used is the Pearson's chi-squared (χ^2) statistical test. This test expresses how much a distribution differs from a general distribution. The general formula is given as follows:

$$\chi^2 = \sum_{u=1}^{U} \frac{(B[u] - E[u])^2}{E[u]} \tag{1.4}$$

being $B[u]$ the observed frequency of u, and $E[u]$ the expected frequency of u. The application to the scatter establishes how much an accumulator differs from the expected frequency that can be expresses as the average distribution:

$$\chi^2_{(k)(h)} = \sum_{u=1}^{U} \frac{(pdf_{(k,h)}[u] - \frac{1}{K}\sum_{k'=1}^{K} pdf_{(k',h)}[u])^2}{\frac{1}{K}\sum_{k'=1}^{K} pdf_{(k',h)}[u]} \tag{1.5}$$

where K is the number of all possible hypothesis of the key and U the set containing all the possible ordinate values of points in the set of power traces. With this expression, the factors can be combined over different values h in order to provide a simple discriminant related to k:

$$scatter_{\chi^2}(k) = \prod_{h \in H} \chi^2_{(k)(h)} \tag{1.6}$$

where H is the set of all possible values h.

When the key byte hypothesis is correct, the hypothesis on $v_{i,k}$ coincides with the real value assumed by the cryptographic algorithm and the related $scatter_{\chi^2}(k)$ is characterized by highest value among others. Therefore, high values in the discriminant maximizes the likelihood that the key byte hypothesis coincides with the secret key byte.The secret key is discovered by repeating this procedure for all the key bytes.

1.2.2 Power countermeasures

The security of cryptographic algorithms with respect to power analysis attacks is improved by software and hardware countermeasures. Power analysis attacks success in discovering the secret key because the power consumption of cryptographic devices depends on intermediate values of the executed cryptographic algorithms. Therefore, the goal of power countermeasures is to make the power consumption of cryptographic device independent of the processed data. The countermeasure include hiding and masking techniques. In the following, more information about the countermeasures are presented.

Hiding

The goal of hiding countermeasures is to make the power consumption of cryptographic device independent of (i) the intermediate values and (ii) the operations that are performed. The proposals developed to reach this goal can be divided in two groups. The first group of proposals introduces variations of power consumption in time domain. The variations consist of performing the operations of the executed cryptographic algorithms at different moments of time during each execution. The second group of proposals affects the power consumption in amplitude domain, by changing the power consumption characteristics.

The reason for the randomization of cryptographic operations is the power trace alignment property for the power analysis attack. The DPA attacks achieve the secret key with low effort when the power traces are aligned, i.e. the power consumption of each operation is located at the same position in each power trace. If this propriety is not satisfied, DPA attacks require significantly more power traces. Therefore, the proposal of introducing variations in time domain by performing the operations of the executed cryptographic algorithms at different moments of time during each execution makes more difficult to attack a device. The most used techniques for the implementation of randomization in time domain are insertion of dummy operations and shuffling of operations. The former inserts randomly dummy operations before, during, and after the execution of the cryptographic algorithm, while the latter randomly change the sequence of operations in a cryptographic algorithm.

The idea behind the variation in amplitude domain is to make the power consumption equal or random during all clock cycle. In other words, the goal is to set the signal to noise ratio (SNR) to zero. This goal is not achievable in practice but the developed solutions allow to lower its value. These solutions act by increasing the noise or reducing the signal. The noise increasing is obtained by performing several operations in parallel. The signal reduction is obtained by filtering the power consumption of a cryptographic device.

Masking

The goal of a power countermeasure is to break the link between power consumption and processed data of the cryptographic algorithm. Masking achieves the goal by randomizing the processed intermediate values. This approach offers

the advantage of acting at algorithmic level without changing the power consumption of a cryptographic device. Indeed, the power consumption of the cryptographic device is data-dependent but independent of the intermediate values.

Masking implements the data randomization by concealing each intermediate value v through a value m, called *mask*, according to the relationship: $v_m = v * m$. The mask m is generated by the device and can be fixed or can change in each execution of the algorithm, resulting unknown for an attacker. The operation $*$ is defined according to the peculiar cryptographic algorithm, and it identifies two kinds of masking: *Boolean* and *arithmetic*. In Boolean masking, the exclusive-or operation is used to combine intermediate values and masks: $v_m = v \oplus m$, while in arithmetic masking, the modular addition: $v_m = v + m \pmod n$, or the modular multiplication: $v_m = v \times m \pmod n$ is used, where the modulus n is defined according to the cryptographic algorithm. In case of encryption, the masks are directly applied to the plaintext or to the key at the beginning of the algorithm. At the end of the encryption, the mask has to be removed to obtain the ciphertext [46].

1.2.3 Timing attack

Timing attack is the type of side-channel attack that involves the time taken to complete critical operations to discover secret information about the cryptographic algorithm.

Implementations of cryptographic algorithms often perform computations in non-constant time. Reasons include performance optimizations to bypass unnecessary operations, branching and conditional statements, RAM cache hits, processor instructions (such as multiplication and division) that run in non-fixed time, and a wide variety of other causes. If such operations involve secret parameters, these timing variations can leak some information and, provided enough knowledge of the implementation is at hand, a careful statistical analysis could even lead to the total recovery of these secret parameters. For example, the execution time for the *square-and-multiply algorithm* used in modular exponentiation depends linearly on the number of '1' bits in the key. While the number of '1' bits alone is not nearly enough information to make finding the key easy, repeated executions with the same key and different inputs can be used to perform statistical correlation analysis of timing information to recover the key completely, even by a passive attacker.

Observed timing measurements often include noise (from such sources as network latency, or disk drive access differences from access to access, and the error correction techniques used to recover from transmission errors). Nevertheless, timing attacks are practical against a number of encryption algorithms, including RSA, ElGamal, and the Digital Signature Algorithm.

Chapter 2

Power attacks on IoT transducers

In this chapter, the vulnerability of the mostly used cryptographic algorithm in Internet of Things, the advanced encryption standard (AES), is proved experimentally. The AES is implemented in an IoT microcontroller and its vulnerability is demonstrated experimentally by measuring the chip power consumption and applying a recently developed side-channel attack, the scatter attack.

A vulnerability test method, adopted to reveal the secret key of the cryptographic algorithm, is presented by relying on a classical statistical method proposed in literature. Then, an experimental case study is reported with respect to a successful attack to an 8-bit IoT microcontroller the AES-128 algorithm. Furthermore, an enhanced version of the attack was also realized by increasing the signal-to-noise ratio (SNR) of the power consumption traces and the results are compared with that obtained without signal pre-processing in terms of discovered number of byte.

2.1 Problem statement

The wide availability of off-the-shelf IoT devices poses severe security threats. Their location in unsupervised environment makes the IoT devices particularly prone to the side-channel attacks. Indeed, an attacker has physical access to these devices and easily manages to acquire leakages, such as timing, power consumption, or electromagnetic emission. This type of attack, instead of attacking the mathematical properties of the algorithm, takes the advantages of physical phenomena that occur when cryptographic algorithm are implemented in hardware. The side-channel attacks result very powerful against the mostly used symmetric solutions in IoT framework, the Advanced Encryption Standard (AES) [33].

In the following, the vulnerability of the advanced encryption standard (AES) cryptographic algorithm implemented in an IoT microcontroller is proved experimentally by measuring the chip power consumption and applying a recently developed side-channel attack, the *scatter* [45]. Then, an enhanced version of the attack is realized by digital pre-processing. The aim of the pre-processing is to improve the signal-to-noise ratio (SNR) resulting in a better outcome of the attack. Indeed, the attack is able to discover the secret key with a lower number of power traces. The AES is implemented on an 8-bit processor of an IoT microcontroller without countermeasures, common in on-field working. The implemented scatter attack is capable of finding the 16-bytes secret key of AES-128 algorithm. The

enhanced version of the attack reduces the number of measured power traces to reveal the secret key.

2.2 Vulnerability test method

The vulnerability of the Advanced Encryption Standard is tested by a power attack, namely the scatter attack, aimed to discover the secret key of an IoT microcontroller (*IoT device under test*), secured by the cryptographic algorithm AES with a secret key of 16 bytes or, that is the same, of 128 bits (AES-128). A testing system emulates the malicious measurement system of the scatter attack and it is wired to the IoT microcontroller under test for monitoring the power consumption (*power trace acquisition phase*) during the encryption. The measured data of the power consumption are processed by a method of signature analysis (*statisical analysis phase*) in order to reveal the secret key of the AES-128 algorithm. The scatter attack is able to reveal the key by analysing several power traces, whose variations in amplitude are closely related to the value of the key. Therefore, the trace acquisition phase is preceded by an *exploratory phase* aimed at triggering the next acquisitions on the power consumption related to the key value-dependent function. A block diagram of the vulnerability test method is shown in Fig. 2.1.

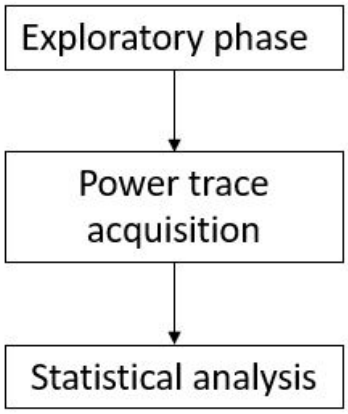


FIGURE 2.1: Block diagram of the vulnerability test method.

2.2.1 Exploratory phase

The exploratory phase aims to identify the power consumption related to the key value-dependent function in order to provide a trigger for the acquisition phase. For the advanced encryption standard, this function is represented by the AddRoundKey and SubBytes of the first round. Indeed, the *key expansion* of the AES algorithm produces 10 round keys, the first of those generated coincides with the secret key. Therefore, the first round employs the secret key in clear. Moreover, among the operations computed in the first round, the AES SBox output has a statistical influence on the power consumption [47].

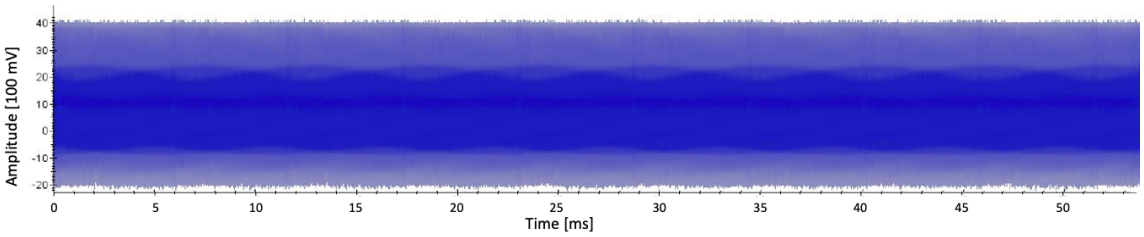


FIGURE 2.2: Power consumption trace acquired during the AES-128 encryption.

The power consumption related to the SubBytes is identified by a coarse-grained acquisition of the AES algorithm and a nested auto-correlation on the acquired power consumption. Below the details are reported. In Fig. 2.2, a coarse-grained power acquisition on the AES is illustrated. The figure highlights no distinguishable pattern. Then, a nested auto-correlation function is performed. The function divides the selected trace part in N fragments of equal length. Successively, a mutual correlation is computed on a matrix of fragments. The first row in the matrix contains the correlation coefficients between the first fragment and all other fragments. The last row in the matrix contains the correlation coefficients between the last fragment and all other fragments. Subsequently, the matrix is visualized in a graph representing a good correlation by a light pixel, and a bad correlation by a dark pixel. Obviously, the diagonal is always a white line, as this represents the correlation of fragments with themselves. An example of the results, considering the first two rounds as the input of the nested auto-correlation, is gray-scale displayed in Fig. 2.3. The nested auto-correlation allows to recognize specific algorithmic structures and where the sensitive operations are executed. In fact, the power consumption related to each operation of the AES can be identified by a visual comparison between the auto-correlation result and the power trace. In this way, the trigger and the duration of the next acquisition are determined.

2.2.2 Power trace acquisition

The acquisition phase is characterized by repeated acquisitions during several AES executions on D random data blocks. On the acquired power traces, an over-sampling ratio and a decimation operation are employed. The former decreases the noise floor, while the latter reduces the samples to be stored in memory. The information about the measurement system and the measurement procedure are extensively discussed in dedicated sections.

The acquisition phase outputs a power trace vector, where each trace $\boldsymbol{t_i'} = (t_{i,1}, ..., t_{i,T})$ corresponds to an input data d_i in the input vector $\boldsymbol{d} = (d_1, ..., d_D)'$. The length T of the trace depends on the sampling rate and the acquisition time window. D and T are fixed at the beginning of the attack. The acquired data are the input of the next statistical analysis.

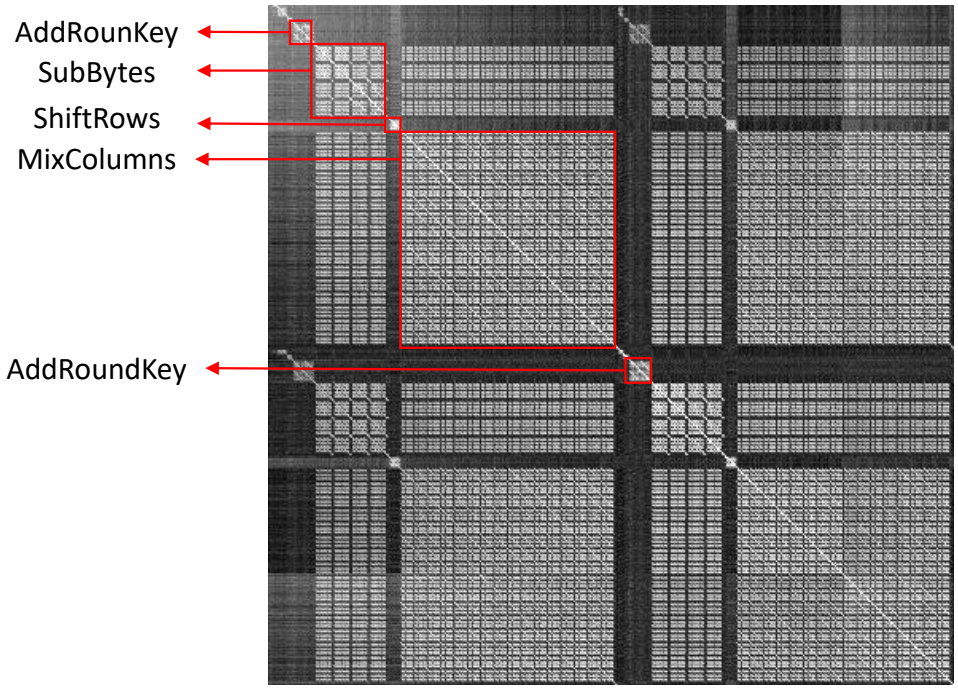


FIGURE 2.3: Auto-correlation results (light pixel: good correlation, dark pixel: bad correlation, and red outlines: operations of the algorithm for the first round.

2.2.3 Statistical analysis

The power traces are analyzed statistically by means of a "*divide-and-conquer*" strategy, that is a single byte of the key is discovered at a time. The flow diagram of the scatter attack statistical analysis is reported in Fig. 2.4. For each byte i_{th} of the key, with $i = [0, 15]$, the following steps are applied:

- *Histogram creation*: from each power trace, obtained by a sampling operation with an 8 bit analog to digital converter (ADC) and, thus, characterized by values in the interval [0, 255], an occurrence histogram is built by counting the number of times each value occurs. This step can be executed only once for all the key byte hypothesis.

- *Accumulation*: for each key byte hypothesis $k \in K = [0, 255]$, the estimated power consumption $h \in H = [0, 8]$ is calculated by applying the Hamming weight power model on the hypothetical intermediate values $v_{i,k}$, i.e. the outputs of SBox in the first AES round. These values are calculated considering all the key byte hypothesis and for each hypothesis the i_{th} byte of all the data blocks are considered. The v values for one key byte hypothesis can be computed as:

$$v_{i,k} = SBox(d_i, k) \quad where \quad i = 0, ..., D$$

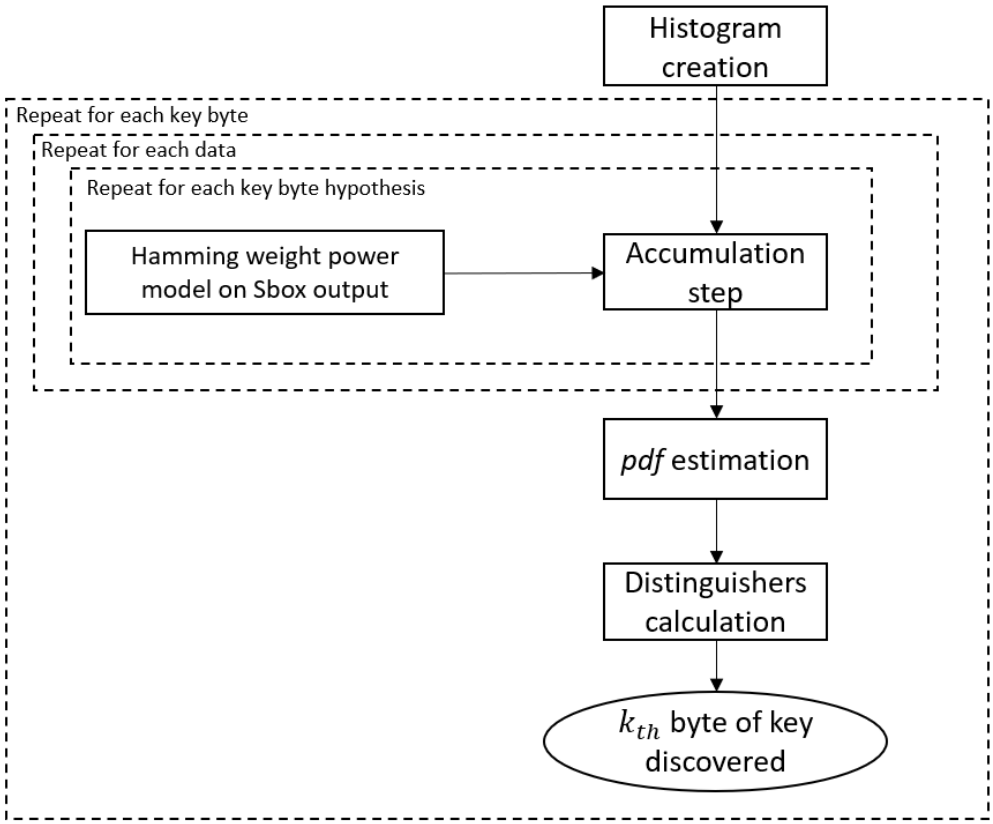


FIGURE 2.4: Flow diagram of the statistical analysis of a scatter attack.

then, the distribution is added into the accumulator $Acc_{(k,h)}$.

- *Pdf estimation*: once the accumulation step is performed, the corresponding values shall be normalised with the total number of point within the accumulator. Denoting X the random variable related to the measurement, and Y the random variable related to the estimation, the probability density function is computed as $pdf(x) = P(X = x|Y)$.
- *Distinguishers calculation*: the Pearson's chi-squared (χ^2) statistical test distinguisher is applied to each $pdf_{(k,h)}$ for a given power estimation h. Then, these factors are combined over the different values h, in order to provide a simple discriminant related to each key byte hypothesis k as:

$$scatter_{\chi^2}(k) = \prod_{h \in H} \chi^2_{(k)(h)} \tag{2.1}$$

When the key byte hypothesis is correct, the hypothesis on $v_{i,k}$ coincides with the real value assumed by the cryptographic algorithm and the related $scatter_{\chi^2}(k)$ is characterized by highest value among others. Therefore, high values in the discriminant maximizes the likelihood that the key byte hypothesis coincides with the secret key byte. The secret key is discovered by repeating the described procedure for all the key bytes.

2.3 Measurement system

In this section, the (i) instrumentation and the (ii) measurement procedure adopted in the trace acquisition are illustrated.

2.3.1 Instrumentation

The device under test (DUT) is the ATMega-163, a low-power CMOS 8 bit microcontroller based on the AVR architecture [48], embedded on a smart card. The microcontroller presents 16 kB in-system flash, 512 Bytes EEPROM, 1024 Bytes Internal SRAM, and 8 MHz maximum clock, and implements the advanced encryption standard with a key of 128 bits. This software implementation does not include the side-channel countermeasures.

The acquisition phase is implemented with the oscilloscope Teledyne Lecroy HDO9304, characterized by 3 GHz bandwidth, 40 GSa/s sample rate, and 8-bit analog-to-digital converter (ADC). A further hardware component, useful for the communication between the DUT and the oscilloscope, is employed. This component consisted of Power Tracer by Riscure [49], a low-noise card reader for side-channel power measurements with precise triggering capabilities. The Power Tracer supplies the integrated circuit on the smart card and communicates with it via ISO/IEC 7816-3 protocol. Moreover, it contains a low noise amplifier (26 $pA/\sqrt{Hz}$ @ 1 MHz) with top end low-noise and high bandwidth analogue components that are electrically isolated from digital circuitry. Thanks to this circuitry, the Power Tracer provides the power consumption in output with an excellent signal-to-noise ratio. The capacitors inside the Power Tracer are precharged to power the smart card during each single measurement to avoid any external noise in the circuit. Typically, power consumption of contact smart cards are measured via a resistor inserted between the ground pin of a smart card and the ground of a card reader. Power Tracer measures power consumption without measurement resistance in the power chain, allowing stable card voltage, maximum signal bandwidth, high sensitivity, and low insertion error [50].

The measurement setup, used in the power consumption acquisition, is shown in Fig. 2.5, and the related block diagram is reported in Fig. 2.6.

2.3.2 Measurement procedure

A personal computer (PC), by means of a market-leading and professional tool, namely Inspector by Riscure, sends the initial configuration parameters to the digital oscilloscope by means of a LAN interface. Moreover, the software communicates with the Power Tracer by means of a USB connection in order to enable the smart card reader for the communication with the smart card. In particular, the PC provides the smart card reader with the plaintexts to be sent to the smart card in the APDU (Application Protocol Data Unit) format. The APDU is the standard protocol, defined by ISO/IEC 7816-4, allowing the communication between a smart card reader and a smart card. The APDU contains a mandatory 4-byte header (CLA, INS, P1, P2) identifying some information of the command

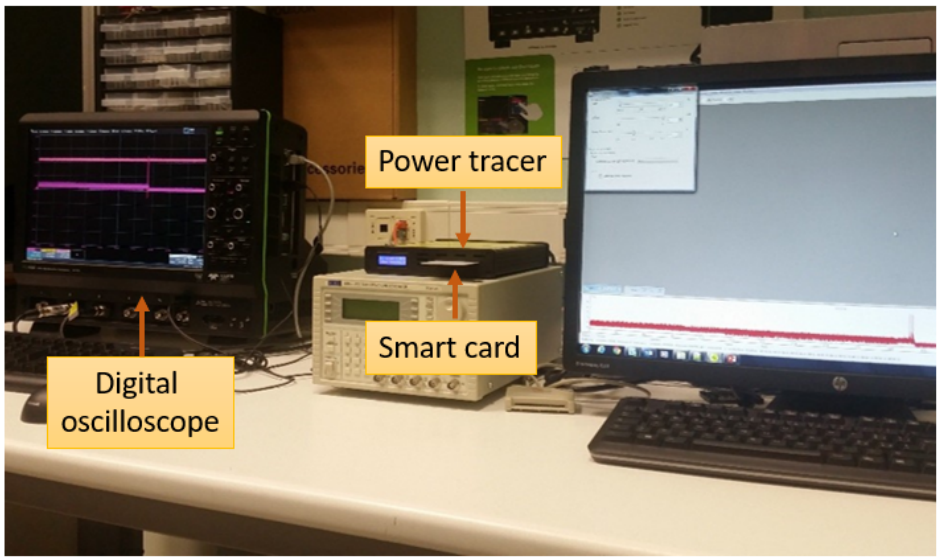


FIGURE 2.5: Measurement setup for the power traces acquisition.

(i.e. the application class, the command, and the parameters used by the command respectively) and the 16 bytes to encrypt. The smart card encrypts the message using the AES-128 algorithm and returns the encrypted message to the smart card reader. The smart card reader returns the ciphertext to the PC. The power tracer is also used to send a trigger signal to the oscilloscope by means of serial I/O line to synchronize the acquisition on the encryption. The oscilloscope acquires the power traces from the output signal of the power tracer, i.e. power consumption with an excellent signal-to-noise ratio. The sample data are sent to the PC by means of a LAN interface. Before the oscilloscope acquires the power consumption, a BNC coaxial low pass filter (Mini Circuits BLP-50+, DC to 48 MHz, 50 Ω) cuts the signal frequencies over 48 MHz as they represent no significant information.

In the communication between smart card and smart card reader, the Inspector tool on the PC appears as the user interface of the card reader allowing to prepare the commands that will be physically sent by the reader to the smart card. The Java code used for sending the encryption commands to the smart card reader is reported in Fig. 2.7. In the APDU, i.e. the cmd variable, $A0$ is the CLA identifying the cipher instruction class, 04 is the INS representing the encryption command, $03\ 00$ are the instruction parameters $P1$ - $P2$, 10 encodes in hexadecimal the 16 bytes to encrypt, $C739D7EAFAE4EDA3C739D7EAFAE4EDA3$ is the plaintext, 10 encodes in hexadecimal the 16 bytes expected as response. In each of the acquisitions, the plaintext is different as the bytes are changed in position by means of the $randomize$ function. The plaintext ($inputData$) and the ciphertext (outputData) are saved in files by means of specific saving functions ($addDatIn$ and $addDatOut$).

The Inspector tool allows to configure the number of acquisitions and the oscilloscope acquisition parameters. 50,000 power traces, each one of 1.5 10^6 samples, were acquired at a sampling rate of 500 MSa/s. Each trace was obtained

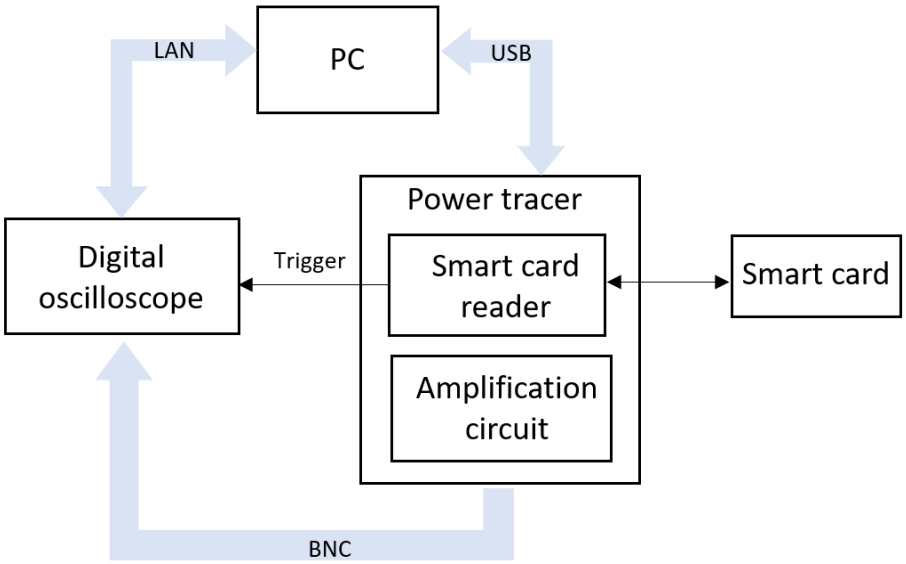


FIGURE 2.6: Block diagram of the measurement setup.

from the encryption of a random value of the *plaintext*. The time required for the acquisition phase was 8 hours.

```
// Sending crypto command
phase("Crypto command");
byte[] cmd = hex("A0 04 03 00 10 C7 39 D7 EA FA E4 ED A3 C7 39 D7 EA FA E4 ED A3 10");
cmd = randomize(cmd, 5, getInputLength());
byte[] inputData = sub(cmd, 5, getInputLength());
addDataIn(inputData);
byte[] response = command(cmd);

// Checking response
if (response.length == getOutputLength() + STATUS_WORD_LENGTH) {
        // Response correctly formatted
        byte[] outputData = sub(response, 0, getOutputLength());
        addDataOut(outputData);
 }
```

FIGURE 2.7: Java code to send the encryption commands to the smart card reader and receive the response.

2.4 Scatter attack on the measured power traces

On the acquired power traces, a decimation operation was applied in order to reduce the samples to be stored in memory, and to decrease the computation time

for the statistical phase. The decimation was obtained with a resampling operation by decreasing the frequency of the samples. For the resample frequency determination, the "fast Fourier transform" (FFT) of a typical power trace was conducted to identify the signal frequency band. The FFT results in Fig. 2.8 show the major harmonics contributions below 500 kHz with the exception of the bin at 4 MHz corresponding to the device under test clock frequency. Therefore, the resampling frequency was fixed to 1 MSa/s. The decimation produces each power traces of 2 kByte in size.

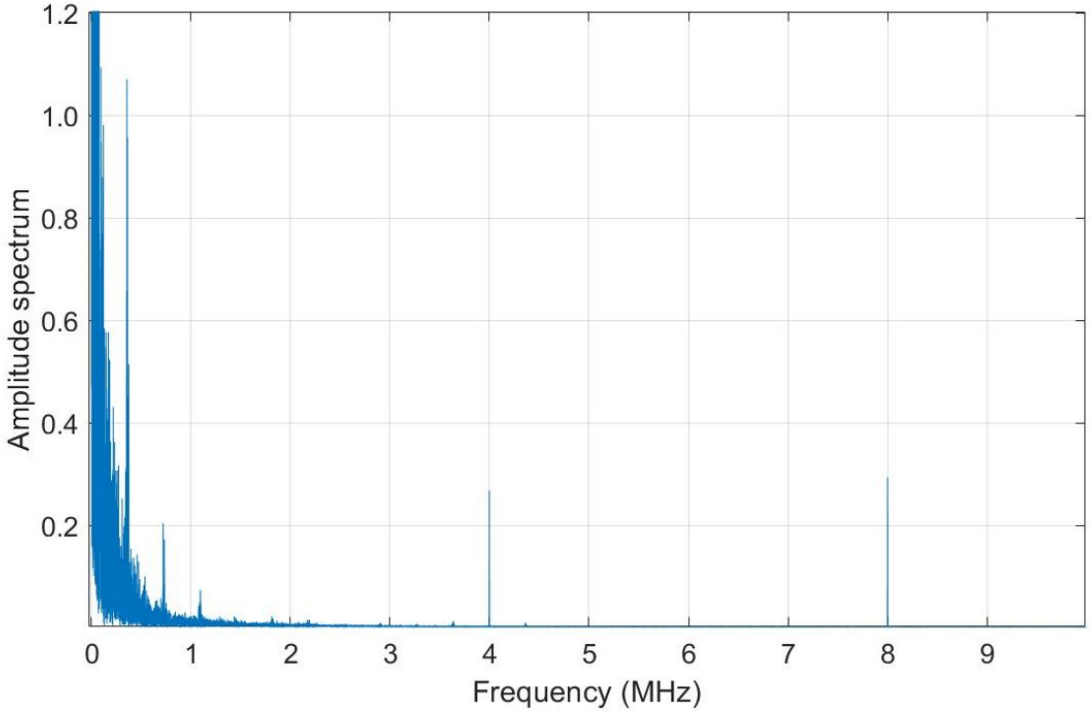


FIGURE 2.8: Amplitude spectrum of a power trace.

The statistical analysis was implemented with the numerical computing environment MATLAB. After loading the plaintexts and the power traces the histogram is created. Then, the subsequent operations of accumulation, pdf estimation, and distinguishers calculation are repeated for each byte of the key, implementing a ”divide and conquer” strategy. Fig. 2.9 shows the estimated discriminants for each hypothesis of the first key byte, normalized to the maximum value. The figure highlights a peak at the hypothesis 172, i.e.’AC’ in hexadecimal, and a almost zero value for the other hypothesis. Since the discriminant related to the correct key byte has the highest value among others, the ’AC’ value is identified as the first key byte. The same operation is repeated for each key byte until the entire key is disclosed. The comparison between the discovered key and the actual key results in 15 bytes correctly disclosed. The result highlights that more power traces should be acquired to discover correctly the 16 bytes key of the AES-128 by means of the presented attack. The attack phase, implemented on a PC with a processor quad-core at 2.8 GHz and 16 GB RAM memory at 2400 MHz, lasted 2 hours on the 50,000 power traces.

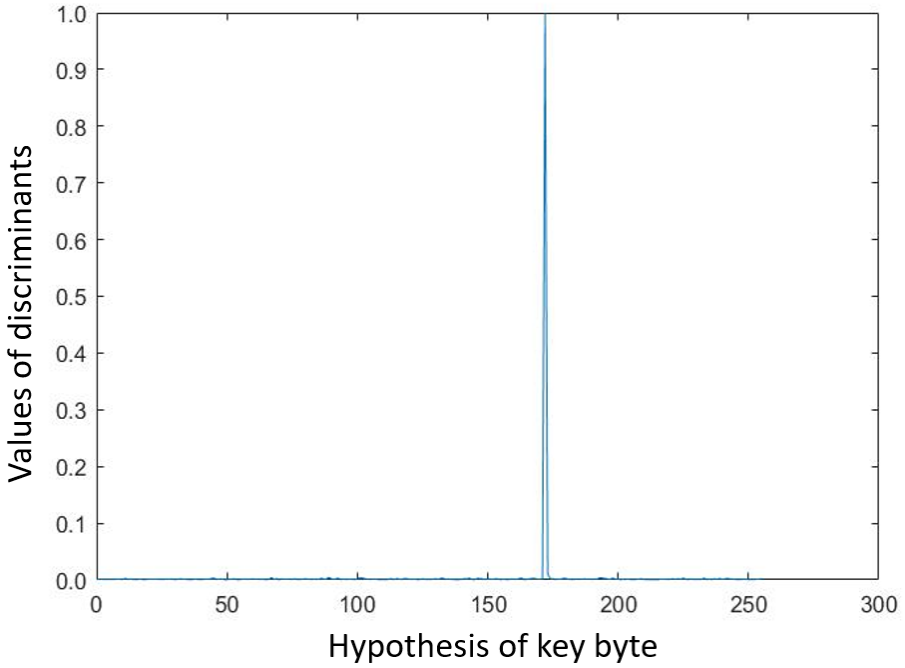


FIGURE 2.9: Normalized discriminants for the hypothesis on the first key byte.

2.5 Enhanced scatter attack

The success rate of the power attacks is significantly affected by the signal-to-noise ratio (SNR) of the power traces [29]. Techniques for noise reduction are particularly important when measuring power consumption because power analysis attacks are very sensitive to the magnitude of these signals to recover the value of the secret key. Therefore, it is important to eliminate noise effectively and improve the SNR of power traces to extract the secret key with minor effort. Indeed, a good level of signal to noise ratio involves in reducing the number of power traces needed to reveal correctly the secret key, and, consequently, in decreasing the time to perform a successful attack.

A filtering operation was employed in order to enhance the test attack effectiveness. The power traces are filtered by a low pass digital filter, which makes each sample a weighted average of the previous and the current sample. The filter behaviour can be express by the equation 2.2, where k is the filter weight.

$$y(n) = \frac{(k \cdot x(n-1) + x(n))}{k+1} \tag{2.2}$$

The transfer function of the filter, obtained as the ratio of the Z-transformation between the output and the input of the filter, can be expressed as in eq. 2.3.

$$H(z) = \frac{z}{(1+k) \cdot z - k} \tag{2.3}$$

The block diagram of the filter is shown in Fig. 2.10.

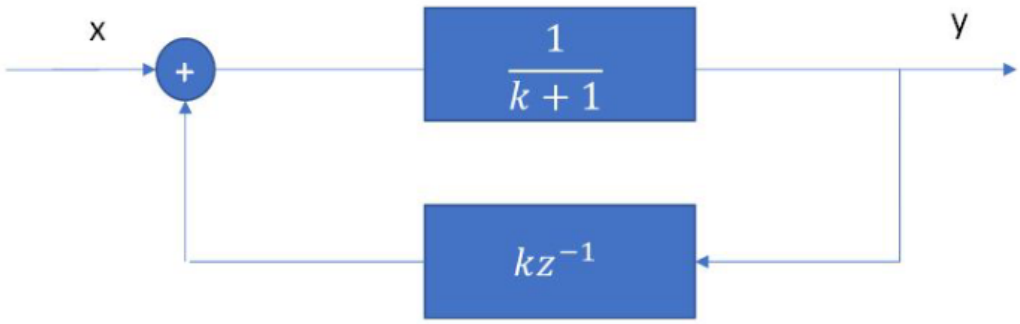


FIGURE 2.10: Block diagram of low-pass filter.

The improvement of the signal to noise ratio is also obtained by the decimation operation applied after an operation of oversampling. In fact, the decimation contributes in improving the signal to noise ratio by reducing the noise floor.

The Fig. 2.11 shows the filtered power consumption trace obtained by applying the digital filtering and the decimation operation on the power trace of Fig. 2.11. The filtering operation allows to improve the signal quality. In fact, by a visual inspection, it is possible identify ten almost clearly separated rounds in the filtered power trace. The first nine rounds have the same duration, while the last one is shorter because the MixColumns operation is missing. The Fig. 2.12, a zoomed-in view of the trace segment nearby the first round, highlights patterns with different frequency allowing to associate the operation performed in each power segment. The signal frequency during the AddRoundKey appear higher than the other operations due to the mathematical computations present in its. The time analysis of the power trace allows to appreciate the wide length of the MixColumns spending more than 50 % of the round duration. The filtered power traces are employed to conduct the statistical analysis of the scatter in order to evaluate the effectiveness of the enhanced test procedure.

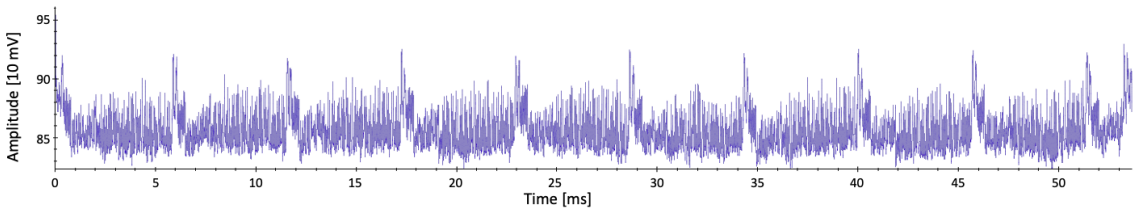


FIGURE 2.11: Filtered version of the power consumption trace of Fig. 2.2.

2.5.1 Optimal filter configuration

For the assessment of the best configuration for the filter, different values for the filter weight were considered. An example of an attack with an acquisition of 50,000 power traces related to the operation *SubBytes* is considered. Fig.

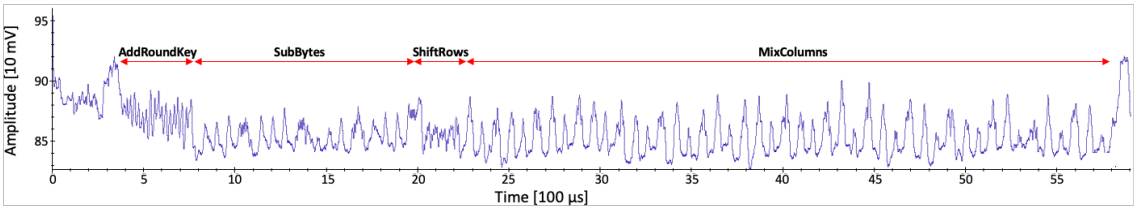


FIGURE 2.12: Filtered power consumption trace related to the first round of AES-128 encryption algorithm.

2.13 shows the number of bytes disclosed by the attack as a function of the filter weight. A linear regression was chosen to model the data owing to its best goodness-of-fitting. Indeed, the ANOVA Fisher-test turns out to be positive at a significance level of 5%. The correctness of this model was proved also by analyzing the behaviour of the attack performance at varying the filter weight. As an example, for a weight equal to 250, the attack allows to discover 14 bytes of the key, such as predicted by the model. The experiment proved that the number of disclosed bytes by keeping the number of power tracks fixed increases according to the filter weight. Indeed, with 50,000 power traces, a filter weight equal to 400 allows to discover the 16 bytes key while a weight of 300 returns only 15 bytes. Therefore, a filter weight equal to 400 was considered as the best configuration.

The reason behind this behaviour is that the lowpass filter acts on cutting off the frequency with little information producing a signal less noisy that allows the power analysis attack to better identify information related to the secret key. In fact, the power analysis attack is able to better correlate the magnitude of the signals with processed data and so determine the value of the secret key.

2.6 Effectiveness of the enhanced test procedure

The effectiveness of the enhanced scatter attack was proved experimentally by reducing the sample size of power traces needed to correctly discover the key. In Fig. 2.14, the number of bytes disclosed correctly are reported as a function of the number of power traces, where the filter weight is fixed to 400. A third-order polynomial interpolation was chosen to fit the data because the related ANOVA Fisher-test returns a significance level at 85%. The correctness of this model was proved also by analyzing the result of the attack at varying number of the power traces. In particular, 14 bytes were disclosed with 15,000 power traces. Therefore, with the uncertainty of a byte, the model respects the actual behaviour of the experiment. The plot highlights that, with a filter weight of 400, a number of 30,000 power traces is sufficient to find the encryption key exactly.

Finally, the Fig. 2.15, gives an overview of all the experiment results. In particular, different colours are used to indicate the disclosed bytes from the experiments, where each experiment is conducted with a fixed value for the number of traces and for the weight of the filter. The plot highlights that a minor number of traces is sufficient to have a successful attack by increasing the weight of the filter.

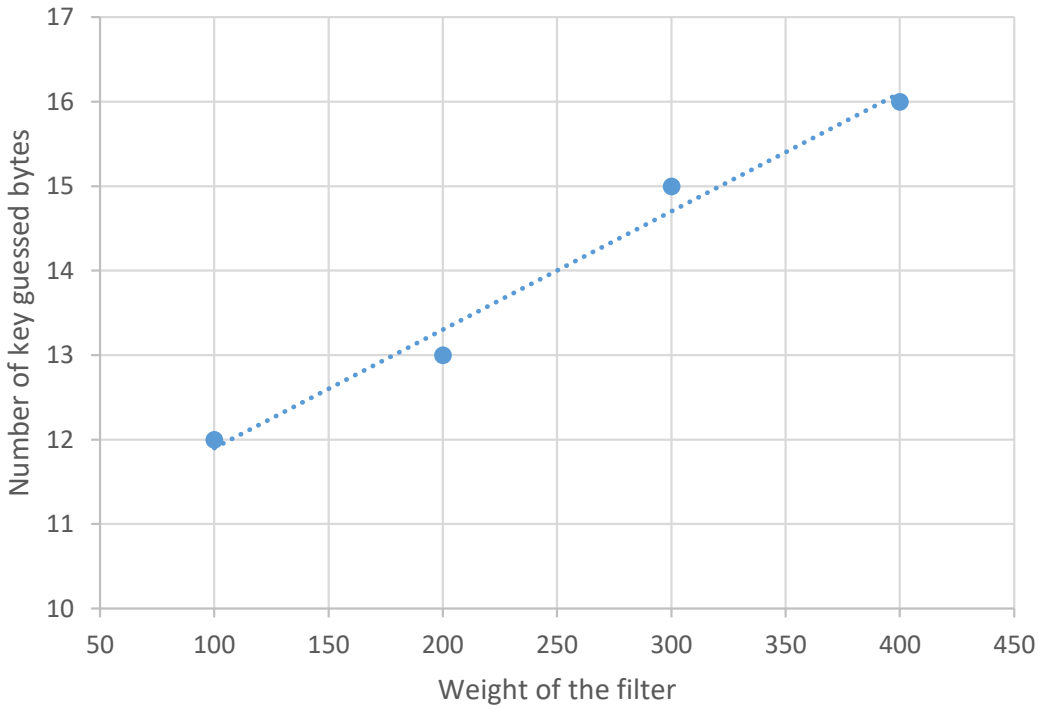


FIGURE 2.13: Number of bytes disclosed by the scatter attack at varying the filter weight (dotted line:linear regression).

The conducted analysis highlights that the proposed filter is able to enhance the effectiveness of the power attack. In fact, the use of a filter allows to discover the entire key unlike the lack of pre-processing with the same number of traces. Moreover, the filter is able to reduce the number of traces needed for a successful attack. Decreasing the number of power traces also reduces the time needed to find the secret key.

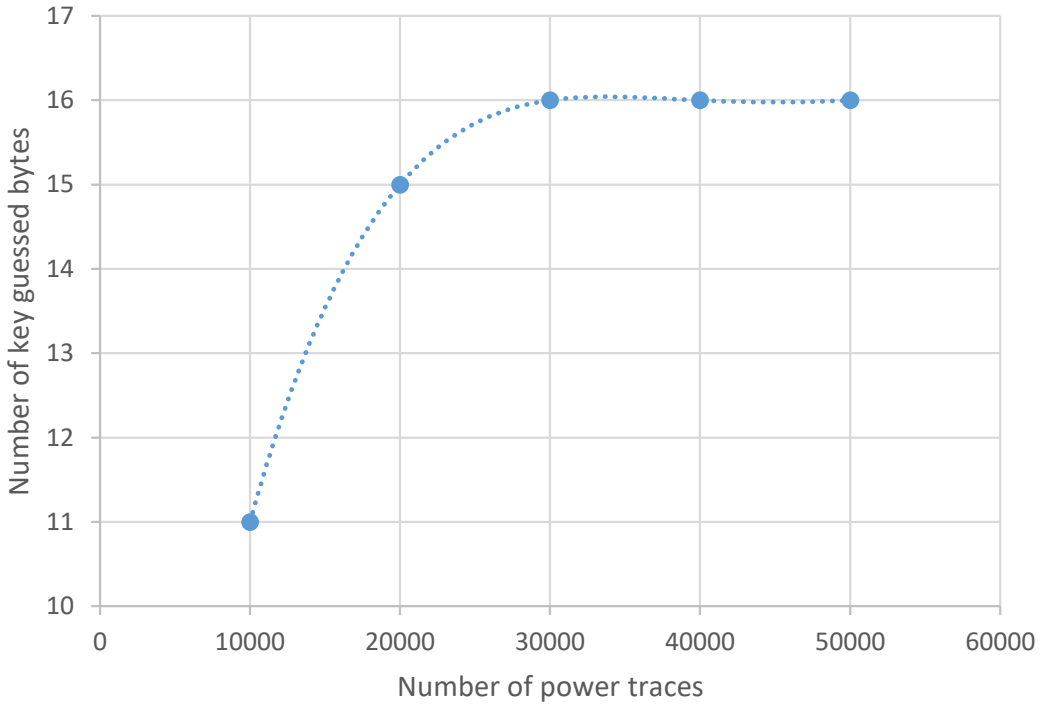


FIGURE 2.14: Number of bytes disclosed by the scatter attack as a function the number of power traces for a weight of the filter equal to 400 (dotted line: 3^{rd}-order polynomial interpolation).

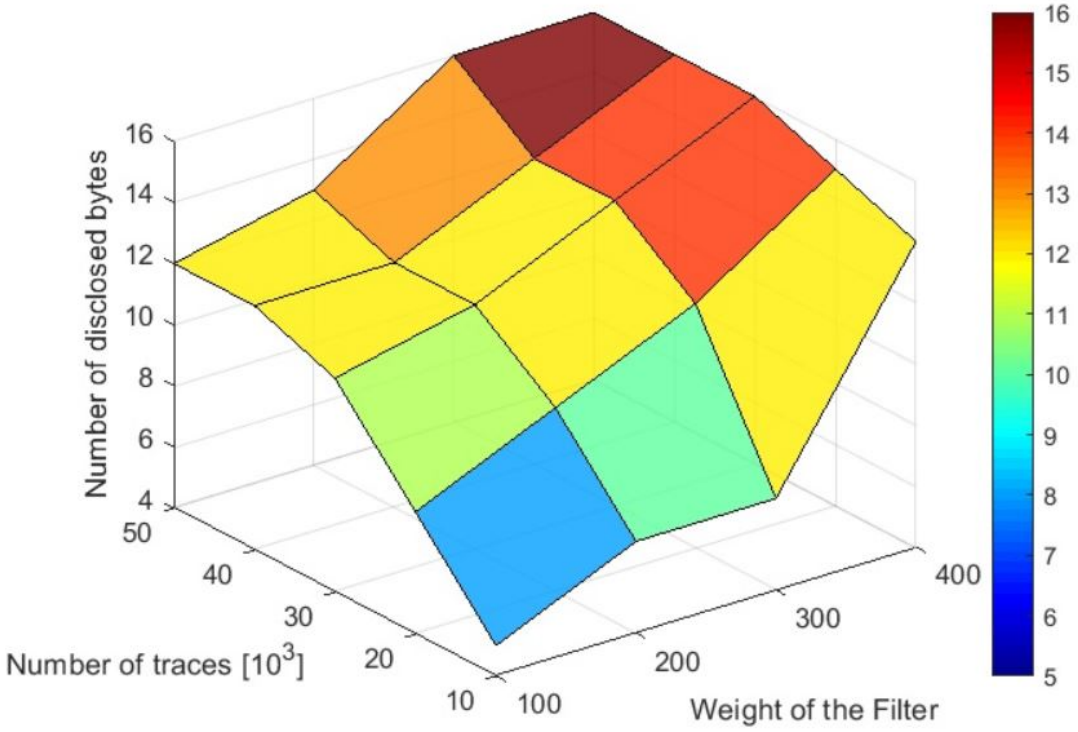


FIGURE 2.15: Plot summarizing the number of disclosed bytes in each experiment obtained with a fixed number of power traces and weight of the filter.

Chapter 3

Optimization of power attacks

In this chapter, a novel optimization technique for power analysis-based attacks is proposed by relying on *fractional experimental design*, which is already successfully used to optimize manufacturing processes. In this new context, such a technique allows to identify the parameter values maximizing the number of correctly discovered key bytes with the minimum cost of the attack in terms of time and resources. The proposed method contributes also to the reduction of repeatability and reproducibility uncertainty and to a more detailed description of the attack procedure. In the chapter, after presenting the optimization method, the application to a case study is reported. This consists of the most powerful power analysis attack, the correlation power attack (CPA).

3.1 Problem statement

The design of an IoT transducer requires a deep hardware security threats analysis in order to ensure confidentiality and integrity of exchanged data over the network. There are different frameworks which describe how a cryptographic device should be protected. An example of such a framework is the *Common Criteria for Information Technology Security Evaluation* (Common Criteria or CC), an international standard (ISO/IEC 15408) for computer security certification. These frameworks are adopted by developers in order to realize a product compliant with the security standards and by security laboratories in order to assess the robustness of devices.

The mainly tasks of the laboratories are (i) evaluate that all the security features are taken into account and are well designed and implemented by the developers during all the design and development phases, and (ii) the security features of the product are tested by the developer. Moreover, the security laboratories implement physical attacks to practically verify the security of the device. If the device actually meets the claims about the security attributes and the verdict of the attack is "fail", then the product is considered safe. In this case, the security laboratory produces a certification that attests a good level of security. Obviously, it is impossible to ensure that a product is fully safe. The security evaluations are based on the idea of allowing competent people to test during a sufficient time the product to verify that in these conditions no vulnerability can be identified.

The activity of physical evaluation is called *Vulnerability Assessment*. Typical physical attacks are side channel attacks, perturbation attacks, and protocol

attacks. These attacks are not exhaustively described by the Common Criteria evaluation. Indeed, these tests are related to a non explicit "state of the art" [51]. As concerns the side-channel attacks, the poorly described phases for conducting an attack (i.e. acquisition, pre-processing, and statistical analysis) contribute to different possible attack implementations and to weakening the generalizability of the results, namely the tests are poorly reproducible. In the scientific literature, different versions of the SCAs are proposed. Regarding the most effective power attack, Correlation Power Analysis (CPA), significant variations involve the phase sequence. For example, in [52] and [53], the pre-processing is omitted while, in [54] and [55], different typologies of filtering are employed. Furthermore, also the lack of a procedure for choosing the parameter values for a fixed type of attack gives rise to possible different implementations. For example, in [56, 57, 58], and [52], the sample rate varies by an order of magnitude.

The lack of an optimized configuration for the attacking parameters contributes also in a higher effort for disclosing the secret key. Indeed, this can result in more power traces needed to discover the secret key and, as a consequence, in more time to conduct an attack. Typically, the improvement of power analysis attacks, aimed at revealing the secret key with the minimum number of power traces, is achieved by pre-processing techniques of power traces. The explored pre-processing techniques are Kalman filter [59], wavelet-based de-noising method [29], and Principal Component Analysis (PCA) [60].

In the following, an optimization of power attack parameters allowing to identify the parameter values maximizing the number of key bytes correctly discovered with the minimum cost of the attack in terms of time and resources is presented. The approach is based on a statistical experimental design. The proposed method contributes also to the reduction of repeatability and reproducibility uncertainty and to a more detailed description of the attack procedure.

3.2 Optimization method

The optimization method proposed in this chapter is based on a a statistical experimental design. In particular, the *fractional experimental design* is considered as it presents the advantage of the lower number of experiments compared to other experimental designs favouring the reduction in experimental times and cost, without penalizing the statistical significance of the results.

The proposed method consists of five main phases: (i) *Definition of problem*, (ii) *Experiment planning*, (iii) *Experiment execution*, (iv) *Result Analysis*, and (v) *Verification*. A block diagram of the proposed method is shown in Fig. 3.1. Firstly, the parameters that majorly influence the attack, the related values, and the objective function to optimize are defined. Then, the experiments are planned on the base of the orthogonal array and are performed. The Analysis of Variance (ANOVA) is applied to investigate the significant level of attack parameters on the outcome of the attack. In this way, the optimum parameter values maximizing the number of key bytes correctly discovered with the minimum cost of the attack in terms of time and resources are identified.

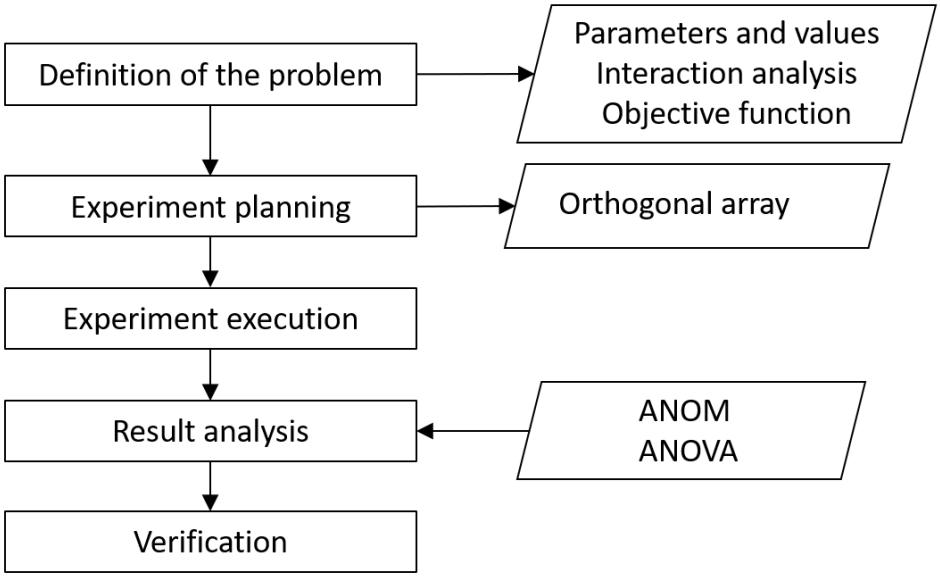


FIGURE 3.1: Block diagram of the optimization method.

3.2.1 Definition of the problem

The definition of the problem consists of (i) identifying the influence parameters on the power attack, (ii) choosing the related values, (iii) assessing the interactions between parameters, and (iv) identifying the objective function to be optimized.

Parameters and values

The block diagram of the power measurements attack is shown in Fig. 3.2. The system (i) acquires the power leaked from an IoT transducer performing cryptographic operations based on the use of the secret key, (ii) pre-processes the power traces, and (iii) applies statistical analysis to discover the secret key. Each phase of the attack requires specific parameters to be configured: the *sampling rate* for the acquisition phase, the *number of power traces* for the statistical analysis, and specific parameters depending on the technique implemented for the pre-processing. Typically, fast bidirectional filter, resampling, Kalman filter [59], wavelet-based de-noising method [29], and Principal Component Analysis (PCA) [60], are employed as pre-processing techniques. On the base of the pre-processing technique to implement, all the configuration parameters are determined. The related values are chosen according to the device under test characteristics.

Parameter interaction analysis

The parameter interaction analysis is carried out to confirm the choice of a linear model for the objective function. The assessment is conducted by means of a graphical representation of the objective function for all the possible combinations of the parameter couples. For two-parameters experiments (A and B) with two values (A_{low}, A_{high}, B_{low}, and B_{high}), two lines result from the analysis: one

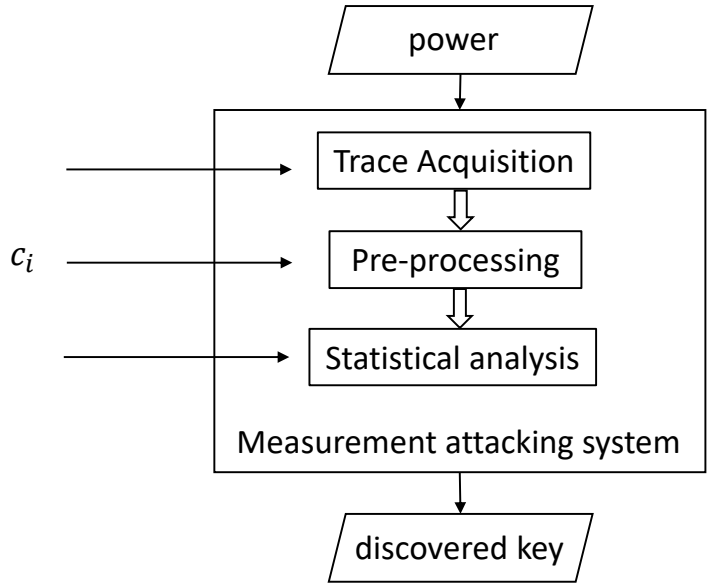


FIGURE 3.2: Block diagram of measurements attack system (c_i: attack configuration parameters).

joins the experimental response in the combinations A_{low}-B_{low} and A_{low}-B_{high}, and the other one in the configuration A_{high}-B_{low} and A_{high}-B_{high}. Lines approximately parallel indicate that the two parameters do not interact significantly and a linear model can be used for the objective function. The graphical representation of the objective function can be extended for more-parameters experiments with more-values. The result involves in more lines, whose slopes establish the presence of interactions among parameters.

Objective function

The model of the objective function depends on the statistical independence of the parameters. In case of statistical independence, an additive model considering only their main effects can be employed. The main effect of a parameter is defined as the variation produced on the experimental response by a change in the parameter value. When the covariance between the parameters is not negligible, their interactions have to be considered in the model.

The goal of a power attack is to fully reveal the encryption key. The fundamental parameter of an attack effectiveness is the number of key byte correctly discovered. Therefore, the objective function can be defined as the maximization of this quality index. An additive model can be used to represent the dependence of the objective function (η) respect to the experimental parameters assumed to be independent:

$$\eta = \mu - \sum_{i=1}^{n} \delta_{iq} + u \tag{3.1}$$

where μ is the *overall mean* of all the η_j outputs of experiments, δ_{iq} is the effect of the parameter i_{th} set at a certain value q_{th} under analysis, n is the number of design parameters, and u is the model uncertainty. The parameter independence can be verified after the experiments by assessing the model prediction accuracy. If the model results inappropriate to describe the process, further additive terms of covariance between factors can be introduced into the eq. (3.1). The goal of the optimization method is to discover the parameter levels maximizing the objective function.

3.2.2 Experiment planning and result analysis

The experiment plan is defined according to an orthogonal array of Taguchi [61], a type of fractional factorial design where all levels of all factors are considered equally. The plan is represented by a matrix, where the rows correspond to an experiment in a given configuration of the attack parameters, and the columns correspond to the value that the parameters have to assume in the experiments.

The experiments allows to assess the effects of each parameter value on the objective function of the eq. (3.1). The estimators d_{iq} of the effects δ_{iq} are computed with the *analysis of the mean* (ANOM), while the uncertainty of model is assessed with the *analysis of the variance* (ANOVA) [62, 63]. The ANOM expression for the calculation of the estimators d_{iq} is:

$$d_{iq} = m_{iq} - \mu \tag{3.2}$$

where μ is the mean of the N results of the experiments:

$$\mu = \frac{1}{N} \sum_{j=1}^{N} \eta_j \tag{3.3}$$

and m_{iq} are the averages of the n_{iq} results of the experiments where the i_{th} design parameters occurs at the q_{th} level:

$$m_{iq} = \frac{1}{n_{iq}} \sum_{j=1}^{n_{iq}} \eta_j \tag{3.4}$$

For the uncertainty of the model, the statistical significance of each design parameters on the objective function is evaluated through the ANOVA. This evaluation is conducted by means of a Fisher test on the variance ratio:

$$F_i = \frac{\sigma_i^2}{\sigma_\epsilon^2} \tag{3.5}$$

where σ_i^2 is the variance of the i_{th} parameters:

$$\sigma_i^2 = \frac{\sum_{q=1}^{n_w} n_{iq} \cdot d_{iq}^2}{\gamma_i} \tag{3.6}$$

with $\gamma_i = n_w - 1$ the degrees of freedom of the i_{th} parameters, and σ_ϵ^2 is the error variance:

$$\sigma_\epsilon^2 = \frac{\sum_{q=1}^{N} \epsilon_j^2}{\gamma_\epsilon} \tag{3.7}$$

γ_ϵ is the degree of freedom of the error:

$$\gamma_\epsilon = \gamma_t - \sum_{i=1}^{n} \gamma_i \tag{3.8}$$

with γ_t the degrees of freedom of the total sum of the deviation squares. The sum of squares of the error is derived from the *theorem of the deviance decomposition* as:

$$\sum_{j=1}^{N} \epsilon_j^2 = \sum_{j=1}^{N} \eta_j^2 - N \cdot \mu^2 - \sum_{i=1}^{n} \sum_{q=1}^{n_w} (n_{iq} \cdot d_{iq}^2) \tag{3.9}$$

where n_w is the level number of each design parameter, n_{iq} is the experiment number in the design matrix where the i_{th} parameter occurs at the q_{th} level.

3.2.3 Optimum prediction and result verification

The optimum attack configuration corresponds to the d_{iq} combination maximizing the objective function (3.1):

$$\eta_{opt} = \mu + \sum_{i=1}^{n_k} d_{iopt} \tag{3.10}$$

where n_k is the number of significant parameters.

The quality of the prediction is verified by comparing the uncertainty prediction u_p with the radius of the $z\sigma_p$ confidence interval. In particular, the following condition has to be satisfied:

$$u_p \leq |z\sigma_p| \tag{3.11}$$

The uncertainty prediction is computed as:

$$u_p = |\eta_{opt} - \eta_{opt}^*| \tag{3.12}$$

with η_{opt}^* the result of the experiment in the optimum configuration, and the variance σ_p^2 is computed as:

$$\sigma_p^2 = \sigma_u^2 \left(\frac{1}{n_r} + \frac{1}{n_0} \right) \tag{3.13}$$

with n_r the number of repetitions of the verification experiment and n_0 the size of the equivalent sample [62] computed from:

$$\frac{1}{n_0} = \frac{1}{N} + \sum_{i=1}^{n_k} \left(\frac{1}{n_{iopt}} - \frac{1}{N} \right) \tag{3.14}$$

where n_{iopt} is the number of experiments with the i-th parameters at the optimum level.

3.3 Case study

The proposed optimization method was applied to a case study consisting of the correlation power analysis (CPA) attack addressed against an 8-bit IoT microcontroller. The attack was performed by the Riscure Inspector [49], the de-facto standard tool used by embedded developers and security certification laboratories, and the IoT microcontroller was protected by AES-128 algorithm.

3.3.1 Power analysis parameter identification

The attack target was the ATMega-163, a low power CMOS 8 bit microcontroller based on the AVR architecture embedded on a smart card. The microcontroller presents 16 kBytes in-system flash, 512 Bytes EEPROM, 1024 Bytes Internal SRAM, and 8 MHz maximum clock, and implements the advanced encryption standard with a key of 128 bits. This software implementation does not include the side-channel countermeasures. Therefore, for the pre-proccesing phase of the power attack, a fast bidirectional filtering and a resampling were implemented through the Riscure Inspector tool. Once the pre-processing technique are established, the configuration parameters for the case-study are determined: the *sampling rate* for the acquisition phase, the *filter weight* of the fast bidirectional filter and the *resampling rate* for the pre-processing, and the *number of power traces* for the statistical analysis.

Within the definition of the problem, the value of sampling rate was fixed at 250 MSa/s, well above the Nyquist rate, since the frequency clock of the device under test was 4 MHz, and the power signal bandwidth was below 500 kHz, while resampling rate, number of traces, and filter weight were considered as influence parameters. As concerns the values of the parameters, typical values for the number of power traces employed in the CPA are of the order of hundreds [56, 57, 58]. Therefore, multiples of one hundred were investigated. For the resample frequency determination, the amplitude spectrum of a typical power trace was determined by a Fast Fourier Transform (FFT) in order to identify the signal frequency band. The FFT results in Fig. 3.3 show the major harmonics contributions below 500 kHz. Finally, no indication is reported in literature about the filter weight, so values of different orders of magnitude are investigated. The parameter values chosen for the present case-study are reported in Table 3.1.

TABLE 3.1: Experimental parameters and their values.

Parameters	Unit	Value 1	Value 2	Value 3
Filter weight		50	500	5000
Resampling frequency	kSa/s	100	500	1000
Number of power traces		100	250	400

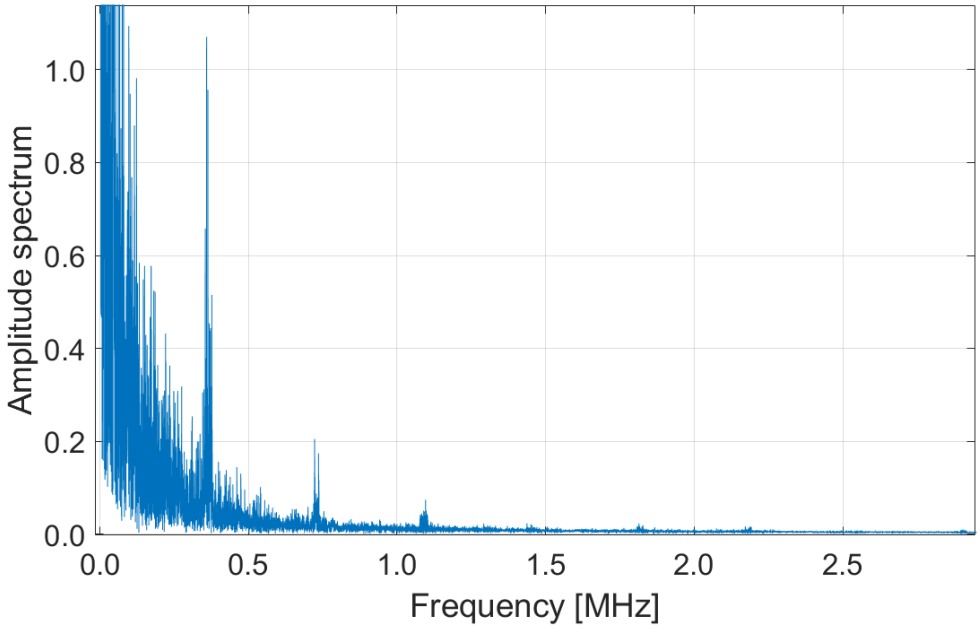


FIGURE 3.3: Amplitude spectrum of a power trace.

The interaction analysis was implemented by the statistical software JMP [64]. The first-order interaction plots, obtained by considering all the possible couple of parameters, are reported in Fig. 3.4: each cell of the matrix represents the interaction plots of the two parameters on the corresponding row and column, respectively. The result is parameterized with respect to the row parameter values. The similar slopes of the trends reveal the negligibility of the first-order interactions, allowing the use of a linear objective function.

3.3.2 Experiment planning and execution

The number of parameters and their values lead to the choice of a L9 orthogonal array [65]. The L9 design consists of a matrix, with up to four columns, corresponding to the number of influence parameters, and nine rows, corresponding to the experiments to be conducted, indicating the values each parameter has to be assumed. The L9 design plans the combinations in order to cover the entire parameter space, favouring the reduction of the experiments number. This design allows to model a problem until 4 parameters and 3 values, so it is suitable for the present case-study with 3 parameters and 3 values. The absence of a fourth parameter enhances the accuracy of the model identification. In fact, the excess of experimental information can be exploited for the uncertainty assessment.

The experiment executions are carried out according to the experimental plan (Table. 3.2), by setting the parameters to the value indicated in each row of the matrix and reporting the related disclosed key bytes.

The acquisition phase was implemented with the oscilloscope Teledyne Lecroy HDO9304 and the Power Tracer by Riscure. The oscilloscope Teledyne Lecroy HDO9304 is characterized by 3 GHz bandwidth, 40 GSa/s sample rate, and 8-bit

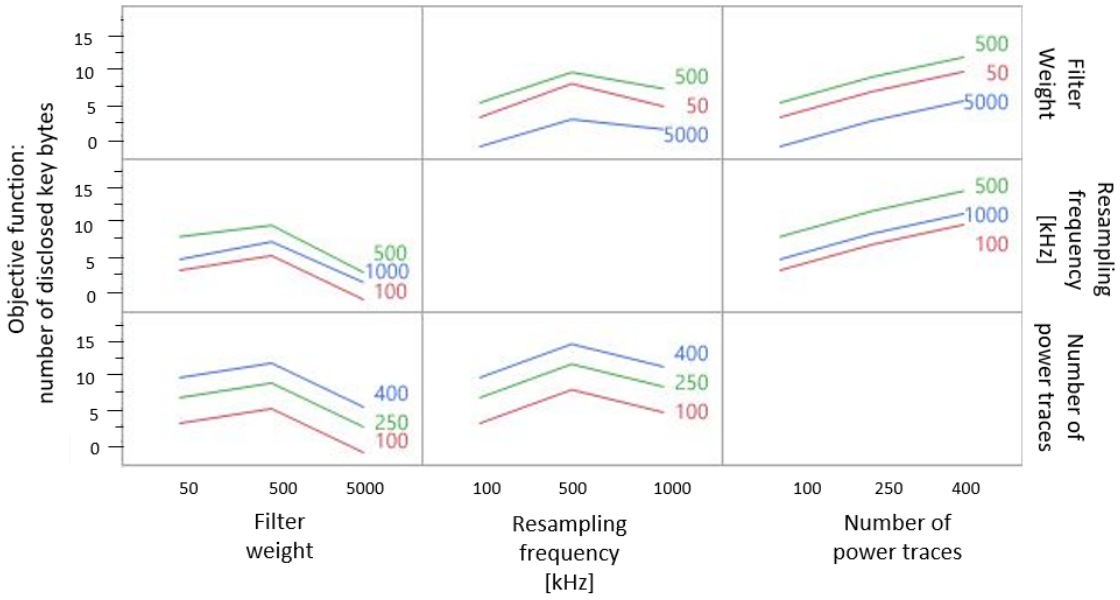


FIGURE 3.4: First-order interaction plots: number of disclosed key bytes in function of the column parameter values and parameterized with respect to the row parameter values. First-order interactions are negligible as the trends have similar slopes.

analog-to-digital converter (ADC). Power Tracer is a low-noise card reader for side-channel power measurements with precise triggering capabilities. This device supplies the integrated circuit on the smart card and communicates with it via ISO/IEC 7816-3 protocol. Moreover, the Power Tracer contains a low noise amplifier (26 $pA/\sqrt{Hz}$ @ 1 MHz) with top end low-noise and high bandwidth analogue components, electrically isolated from digital circuitry, that provides the power consumption in output with an excellent signal-to-noise ratio. The details on the measurement setup are reported in Section 2.3, while the measurement procedure is described in Section 2.3.2.

The CPA statistical analysis phase was realized with the Inspector Tool. This phase consists of calculating the correlation between the predicted and real power consumption, where the predicted power is calculated for each key hypothesis on the base of a power model. High correlation values allow to identify the actual secret key. More detailed information on the statistical analysis for conducting a CPA attack are reported in Section 1.2.1. The Table 3.3 reports the mean and the standard deviation of the experimental results for the present case-study for each row of the L9 plan. The means were computed on 10 repetitions conducted for each L9 matrix row.

TABLE 3.2: Orthogonal array of experiments (Taguchi L9), grey: empty column.

Run	1 Filter weight	2 Resampling freq. [kHz]	3 Number of power traces	4
1	50	100	100	1
2	50	500	250	2
3	50	1000	400	3
4	500	100	250	3
5	500	500	400	1
6	500	1000	100	2
7	5000	100	400	2
8	5000	500	100	3
9	5000	1000	250	1

TABLE 3.3: Experimental results of the L_9 orthogonal array. Means and 1-σ repeatability are calculated on 10 repetitions.

Exp. N	Parameters			Experimental results η_j	
	Filter weight	Resampling freq. [kSa/s]	Number of power traces	Mean of disclosed bytes	1-σ
1	50	100	100	2.7	1.4
2	50	500	250	11.4	2.0
3	50	1000	400	11.0	1.2
4	500	100	250	8.6	2.3
5	500	500	400	16.0	0.0
6	500	1000	100	6.9	1.0
7	5000	100	400	5.1	1.3
8	5000	500	100	2.4	1.1
9	5000	1000	250	4.7	0.8

3.3.3 Result analysis

The effects of each parameter value on the objective function are evaluated by means of the analysis of mean (ANOM). Firstly, the mean μ of the 9 results of the experiments, eq. (3.3), and the averages m_{iq} on the 3 results of the experiments where the i_{th} design parameters occurs at the q_{th} level, eq. (3.4), are calculated. The μ value is 7.6 while the m_{iq} values are reported in Table. 3.4. Then, the estimators d_{iq} are computed with the eq. (3.2) and reported in Table. 3.5. In this way, the objective function values are computed by means of the eq. (3.1) for each row of the L9 matrix. In particular, for the mean calculation on a row, only the d_{iq} of the q_{th} levels present on that row are considered. The obtained values are reported in Table. 3.6.

The ANOM results allow to evaluate the parameter value effects on the objective function, computed as the mean of the values of the objective function η

TABLE 3.4: m_{iq} values
(i: design parameter with 1 = filter weight, 2 = resampling frequency, 3 = number of power traces, q: levels of each design parameter).

		i		
		1	2	3
q	1	8.4	5.5	4.0
	2	10.5	9.9	8.2
	3	4.1	7.5	10.7

TABLE 3.5: d_{iq} values
(i: design parameter with 1 = filter weight, 2 = resampling frequency, 3 = number of power traces, q: levels of each design parameter).

		i		
		1	2	3
q	1	0.7	-2.2	-3.6
	2	2.9	2.3	0.6
	3	-3.6	-0.1	3.1

when the parameter i_{th} assumes the level q_{th}. For example, the effect of the value 50 for the filter weight is computed as the mean of the η on the row 1,2,3 of the L9 plan, i.e. when the filter weight is equal to 50 ($(2.5 + 11.2 + 11.3)/3 = 8.4$). The effects for all the parameter values are reported in Table 3.7 and plotted in Fig. 3.5, where the estimated error (for a confidence level of 99.97 %) is assessed according to the ANOVA procedure. The values of the parameters maximizing the number of byte discovered by the power attack can be easily identified. The best configuration is 500 for the filter weight, 500 kSa/s for the resampling frequency, and 400 for the number of power traces.

The Analysis of Mean (ANOVA), as reported in Section 3.2.2, is conducted to assess the statistical significance of the parameters. The analysis was based on variance ratio (F_i) between the i_{th} parameter variance and the error variance. For the variance calculation, the degrees of freedom and the sum of squares are computed. All the results for the F_i value calculation are reported in Table 3.8. Moreover, the F_i of the model, obtained as the ratio between the model variance and the error variance, and the percentage contributions are reported. The ANOVA confirmed the statistical significance of all parameters. All the parameter F-values are higher by an order of magnitude with respect to F-value (19.0) calculated at a confidence level of 95 % of the Fisher test. The highest 158.2 F-value (corresponding to 42.1 % contribution) reveals the number of traces as the most significant factor. Results of the statistical significance analysis are shown in the Pareto chart (Fig. 3.6). The histogram bars report the F-value for each attack parameter, while the line represents the related percentage contribution. The bars are presented in descending order: this highlights the number of power traces as the most important parameter among those considered.

TABLE 3.6: Disclosed bytes computed with the objective function model of eq. (3.1).

Run	Filter weight	Resampling freq. [kHz]	Number of power traces	η
1	50	100	100	2.5
2	50	500	250	11.2
3	50	1000	400	11.3
4	500	100	250	8.9
5	500	500	400	15.8
6	500	1000	100	6.7
7	5000	100	400	4.9
8	5000	500	100	2.7
9	5000	1000	250	4.5

TABLE 3.7: Parameter value effects on the objective function.

Parameters	Values		
	1	2	3
Weight of the filter	8.4	10.5	4.1
Resampling frequency	5.5	9.9	7.5
Number of power traces	4.0	8.2	10.7

3.3.4 Optimum attack configuration and verification

The optimum attack configuration corresponds to the d_{iq} combination maximizing the objective function. As highlighted in Figure 3.5, the best attack configuration is 500 for the filter weight, 500 kSa/s for the resampling frequency, and 400 for the number of power traces. The quality of the prediction is verified by comparing the uncertainty prediction u_p with the radius of the $z\sigma_p$ confidence interval, as explained in Section 3.2.2, considering a value equal to 2 for z.

All the key bytes (16) correctly disclosed is the predicted result in the optimum attack configuration. For the experimental verification, 10 repetitions were implemented. The output of each repetition was 16 bytes correctly revealed. Therefore, the uncertainty prediction ϵ_p was equal to 0 and the verification reported a positive result.

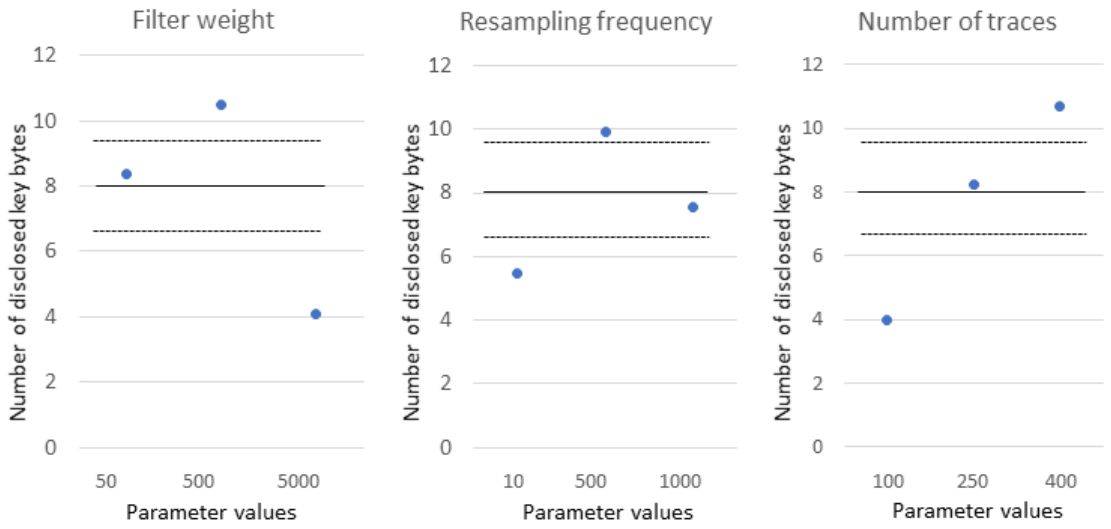


FIGURE 3.5: The parameter effects on the objective function. (The continuous line: effect mean, dashed line: uncertainty range, 2σ: confidence interval.

TABLE 3.8: ANOVA result.

Parameters	Degree of freedom (DF)	Sum of square (SS)	Variance (SS/f)	F_i	Contribution of factor [%]
Model	6	163.3	27.7	124.0	99.7
Filter weight	2	64.4	32.2	147.9	39.3
Resampling frequency	2	30.0	15.0	68.8	18.3
Number of traces	2	68.9	34.4	158.2	42.1
Error	2	0.4	0.2		0.3
Total	8	163.7			100

F = 19 (95% confidence level).

3.4 Metrological improvement

The optimization method also contributes to increase the repeatability and the reproducibility of the attack. In order to prove the metrological improvements resulted from the optimization method, the 1-$sigma$ repeatability and 1-$sigma$ reproducibility of the population are computed, and the results are compared with the corresponding quantities obtained in the optimum attacking configuration.

As a consequence of the experimental design orthogonality, the nine experiments reported in Table 3.3 are a representative population sample of all possible experiments. Therefore, the mean of the 1-$sigma$ repeatability of the 9 experiments is a good estimator of the 1-$sigma$ repeatability of the population. The 1-$sigma$ repeatability of the population is 1.2 in absolute value and 16 in percentage, normalized to the mean of the disclosed key bytes obtained from the model.

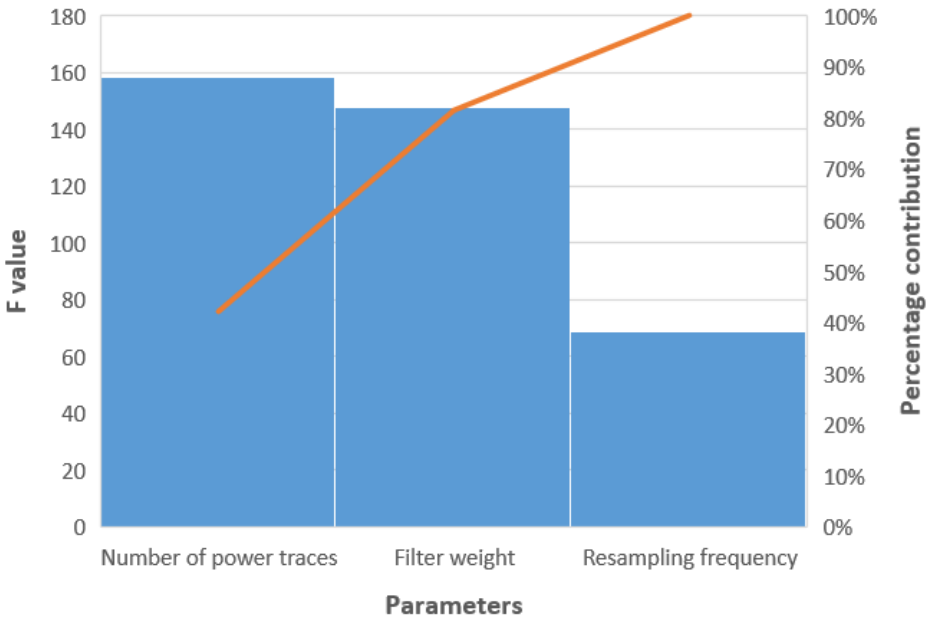


FIGURE 3.6: Pareto chart of the parameters: the histogram bars represent the F-values (left axis) and the orange line the cumulative percentage contributions (right axis).

The optimum parameter configuration, resulted from the proposed optimization method, exhibits a null 1-$sigma$ repeatability. Therefore, adopting the optimum configuration, an improvement in the repeatability can be achieved for the Vulnerability Assessment.

Regards the reproducibility, once the model has been identified and verified statistically, the 1-$sigma$ reproducibility of the population can be estimated by the standard deviation of the disclosed byte means forecast by the model. In the Table 3.9 the number of disclosed byte forecast by the model are reported for all the possible combinations of the attacking parameters. The 1-$sigma$ reproducibility of the population is computed as the standard deviation of the 27 values obtained from the model. The 1-$sigma$ reproducibility of the population results 4.3 in absolute value and 56 in percentage, normalized to the mean of the disclosed key bytes obtained from the model. The 1-$sigma$ reproducibility of the optimum configuration, computed considering only the contribution of 3 parameters, results in a null value.

The Table 3.10 summarizes the 1-$sigma$ repeatability and reproducibility percentage values of the population and of the optimum configuration. The results highlights an improvement in the repeatability. Moreover, an enhancement in 1-$sigma$ reproducibility are also expected by eliminating the contribution of uncertainty due to the variability of the considered parameters.

TABLE 3.9: Number of disclosed byte forecast by the model for all the possible combinations of the attacking parameters.

Run	Filter weight	Resampling freq. [kHz]	Number of power traces	Disclosed bytes
1	50	100	100	2.54
2	50	100	250	6.78
3	50	100	400	9.24
4	50	500	100	7.01
5	50	500	250	11.24
6	50	500	400	13.71
7	50	1000	100	4.61
8	50	1000	250	8.84
9	50	1000	400	11.31
10	500	100	100	4.68
11	500	100	250	8.91
12	500	100	400	11.38
13	500	500	100	9.14
14	500	500	250	13.38
15	500	500	400	15.84
16	500	1000	100	6.74
17	500	1000	250	10.98
18	500	1000	400	13.44
19	5000	100	100	-1.76
20	5000	100	250	2.48
21	5000	100	400	4.94
22	5000	500	100	2.71
23	5000	500	250	6.94
24	5000	500	400	9.41
25	5000	1000	100	0.31
26	5000	1000	250	4.54
27	5000	1000	400	7.01

TABLE 3.10: Comparison of percentage 1-σ reprobucibility and 1-σ repeatability between variable and optimum parameter configuration.

	% 1-σ reprobucibility due to parameter variation	% 1-σ repeatability
Variable parameter configuration	56	16
Optimum parameter configuration	0	0

Chapter 4

Countermeasures against power attacks

In this chapter, a method for the Vulnerability Assessment of Internet of Things (IoT) transducers cybersecurity at varying the power countermeasures is proposed. The method provides a metric to express the effectiveness of a countermeasure in straightening the IoT transducers security. The countermeasures under test reinforce the most popular cryptographic algorithm, Advanced Encryption Standard (AES), running on a low-cost, low-energy, and low-performance smart transducer. The security of such countermeasures is assessed against the attacks standardized in smart card industry: differential power analysis (DPA) and correlation power analysis (CPA). In the chapter, after presenting the method for the Vulnerability Assessment, the application to a case study is reported.

4.1 Problem statement

In an IoT architecture, threats are not just network-based. Indeed, the presence in the environment of IoT transducers, collecting vital health parameters [3], environmental parameters [66], and many others [67], opens the doors to a wide range of attacks [68], producing a serious problem for the IoT security. The side-channel attacks, for instance, require physical access to the device in order to monitor chip leakages, as timing, power consumption, or electromagnetic emission. This typology of attack are very powerful as can reveal the secret key of cryptographic devices with low cost equipment compromising the confidentiality and the integrity of the system as a whole. For the above reasons, the design of an IoT transducer requires a deep security threats analysis. Moreover, proper security measures have to be considered. Contributing to ensuring adequate protection for IoT transducers, today means safeguarding integrity, confidentiality, and availability of data, in many strategic sectors: healthcare [19, 69], smart house [15], environmental monitoring [70], and industry 4.0 [18, 71].

Hiding and masking are two widely used methods to secure device against power attacks [72]. The goal of a power countermeasure is to break the link between power consumption and processed data of the cryptographic algorithm. The hiding countermeasures introduces variations of power consumption in time domain or amplitude domain, while masking implements a data randomization by concealing each intermediate value. The result of these operations prevents

any attack from revealing the secret by exploiting the leakage of information in the power consumption. Possible implementations of such countermeasures include software and hardware solutions. In particular, the countermeasures can affect three levels of abstraction (Fig. 4.1): algorithm level, gate level, and transistor level [73]. Algorithmic-level countermeasures are applied to the source code

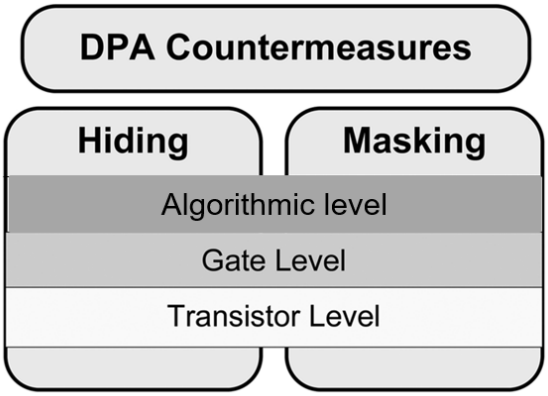


FIGURE 4.1: DPA countermeasures at different abstraction levels.

of the cryptographic algorithms. The second type of DPA countermeasures takes place at the gate level of the digital circuitry. Logic styles that provide constant power consumption regardless of the processed data are built on standard logic gates such as AND, OR and XOR. This type of countermeasure is known as a hiding technique, and it utilises the concept of dual-rail with pre-charge logic (DPL). On the other hand, masking countermeasures can be applied at this level by performing the logical operations of the original inputs masked by some random data. Countermeasures at gate level can be easily accommodated into standard-cell design flow within FPGA or ASIC. The drawback of gate-level countermeasures is the area and performance overhead that is introduced by the extra redundant logic. The third type of DPA countermeasures is based on building customised cells from transistor networks. These transistor-level countermeasures are based on either masking or hiding methods. The design target of such countermeasures is to build new logic gates that perform the standard logical function based on a different transistor layout. The new transistor layout should provide either constant power consumption regardless of the processed data or masking at transistor level. The drawback of transistor-level countermeasures is the complexity of incorporating new designs on the standard cell-based design flow. On the upside, such designs are lighter and provide better performance in terms of speed and power when compared to gate-level countermeasures.

Security countermeasures must be well designed and implemented to counteract the side-channel attacks. Therefore, it is necessary to evaluate their resilience in the presence of such kinds attacks [74]. In [75], an automatic verification tool of software power countermeasure implementations is presented, while in [76] testing strategies based on power analysis attacks, including black box and clear box methods, are reported. In healthcare literature, only the first steps

have been taken regarding vulnerability evaluation respect to the side-channel attacks. In [77], a security-oriented design framework for medical IoT transducer is proposed to assist designers in identifying the components directly or indirectly affected by hardware attacks. In [78], a platform performing Correlation Power Analysis (CPA) attacks is presented to evaluate the robustness of cryptographic algorithms in MCU-based systems.

In this paper, a method based on power measurement for the Vulnerability Assessment of IoT transducers at varying the countermeasures is proposed. Moreover, the method provides a metric to express the effectiveness of a countermeasure in straightening the IoT transducer security.

4.2 Strength assessment method

The proposed method is aimed to assess the security performance among different power countermeasures designed to reinforce a software implementation of AES-128. In Fig. 4.2, the principal phases of the method are illustrated. Firstly,

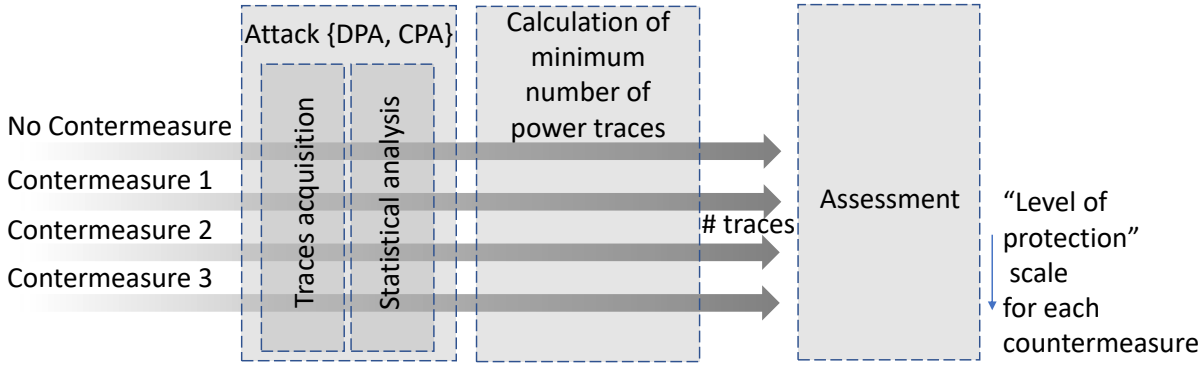


FIGURE 4.2: Block diagram of test procedure.

power attacks are addressed against the device implementing the AES-128 protected by a countermeasure. The chosen attacks are standardized in smart-card industry: differential power analysis (DPA) and correlation power analysis (CPA). These consist of (i) a trace acquisition phase and (ii) a statistical analysis one. Each attack is conducted at varying the security measures. From each attack, the number of traces needed to discover the secret key is calculated. Indeed, this parameter is used for assessing the countermeasures effectiveness. The more the countermeasure is effective, the more the number of traces increases. The method provides also a metric, *strength factor* (SF), that quantifies the level of protection for each countermeasure. This parameter is computed in the assessment phase.

4.2.1 Power attacks

The strength assessment method begins with the power attacks, DPA and CPA. The *trace acquisition* phase is extensively described in Section 2.2. As concern

the *statistical analysis*, the generalized method for DPA and CPA, valid to any cryptographic algorithm, is reported in Section 1.2.1 and 1.2.1, respectively. For clarity of reading, in this Section the flow diagrams of the DPA and CPA statistical analysis applied to the AES are reported (Fig. 4.3 and Fig. 4.4 respectively).

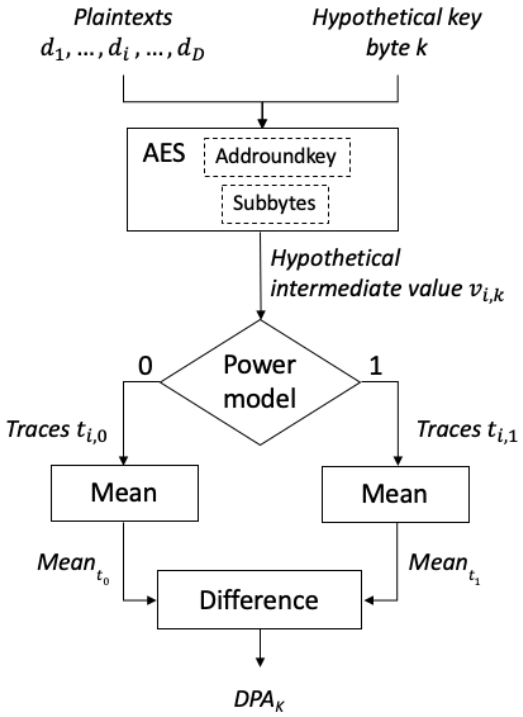


FIGURE 4.3: Flow diagram of Differential Power Analysis (DPA) applied to the Advanced Encryption Standard (AES).

For the AES, the hypothetical intermediate value corresponds to the output of the SubBytes operation of the first AES round. Indeed, this operation exhibits a strong correlation with the key value.

In addition to these phases, the attack implements also a pre-processing phase. It follows the trace acquisition, improving the quality of the power traces. This phase includes: (i) filtering, (ii) resampling, and (iii) elastic alignment.

The filtering operation, described in Section 2.5, consists of a low pass digital filter which makes each sample a weighted average of the previous and the current sample. It allows to improve the signal quality decreasing the noise level. The resampling implements a sampling with a lower frequency contributing to reducing the samples to be stored in memory. The elastic alignment synchronizes the power traces by compressing and stretching them. This technique is powerful in presence of random delay insertion countermeasure. In fact, the traces synchronization involves higher mean values and correlation coefficients improving the sensitivity of DPA and CPA respectively. The Fig.4.5 shows the overlapping of two power traces without (a) and with (b) the alignment technique. The random behaviour in the power traces of Fig. 4.5(a) highlights the presence of a random delay insertion countermeasure, while the Fig. 4.5(b) exhibits as the alignment technique reduces the countermeasure effect by synchronizing the traces. From

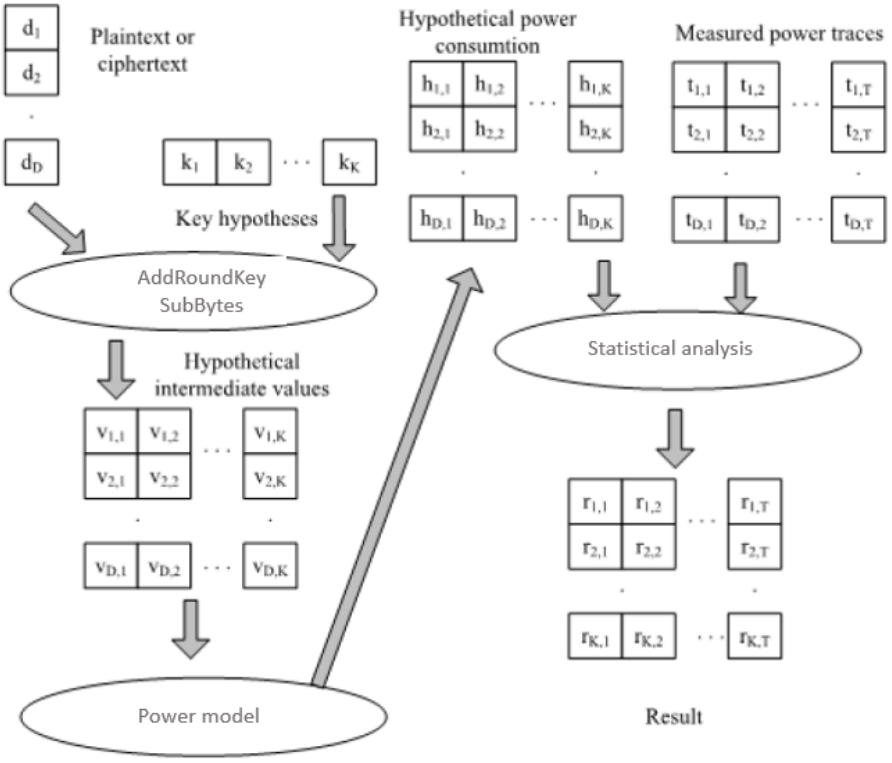


FIGURE 4.4: Flow diagram of Correlation Power Analysis (CPA) applied to the Advanced Encryption Standard (AES).

the example in Fig. 4.5, it is evident how the synchronization of power traces contributes to a higher correlation between the traces: there is major correspondence between the power traces and the executed operation in the algorithm.

4.2.2 Calculation of minimum number of power traces

Each of the eight combinations between attacks and countermeasures is repeated N times. Each repetition $r \in$ [1,N] returns the minimum number of power traces $min_{A,C,R}$, where A = $\{ 0, 1 \}$ indicates the attack (0 = DPA and 1 = CPA), C refers to the security measure, and R is a particular repetition. A different batch of power traces is considered in each repetition.

The first repetition determines the minimum number of power traces $min_{A,C,1}$ with a *successive-approximation* method in a range with extremes determined in a preliminary experimental campaign. The successive N-1 repetitions implement a *grid search* method with a variable step initialized to the minimum number of power traces $min_{A,C,1}$ found in the first repetition. The step is an increment of a certain number of power traces until the secret key is not fully recovered, and a decrement of an order of magnitude lower until the minimum number of power traces $min_{A,C,R}$ is identified. At the end of the repetitions, the minimum number of power traces min_{A^*,C^*}, for each countermeasure at varying the attack, is computed as the mean of $min_{A^*,C^*,R}$.

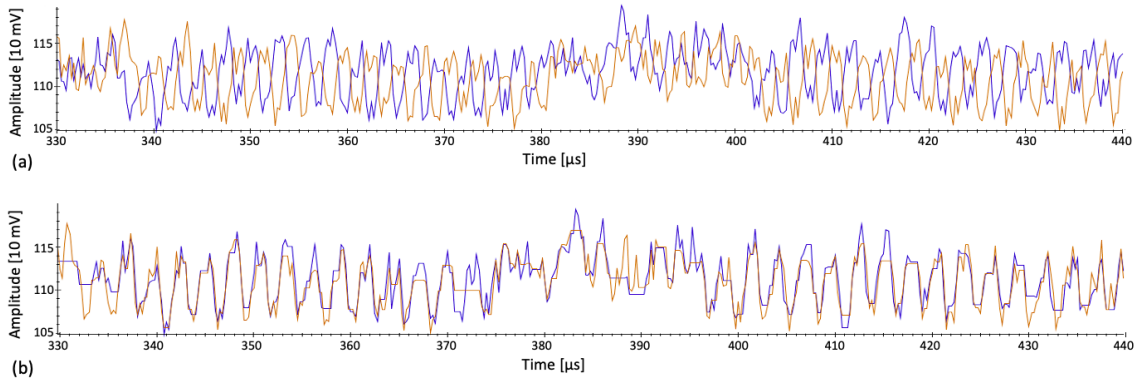


FIGURE 4.5: Overlapping of two power traces without (a) and with (b) the alignment technique.

4.2.3 Assessment

The assessment phase has the goal to compute a synthetic parameter in order to quantify the security level introduced by each countermeasure. This parameter, identified as *strength factor* (SF), is assessed as:

$$SF_{C*} = \frac{\sum_{i=1}^{N_A} min_{A,C^*}}{\sum_{i=1}^{N_A} min_{A,1}} \tag{4.1}$$

where N_A is the number of implemented power attacks, min_{A,C^*} is the minimum number of power traces for a fixed countermeasure C^* at varying of the power attack, and $min_{A,1}$ is the minimum number of power traces for no-countermeasure at varying of the power attack.

4.3 Case study

In the following, the application of the proposed method to a case study is reported. In particular, the security level of three types of power countermeasures that reinforce the most popular cryptographic algorithm, Advanced Encryption Standard (AES), running on a low-cost, low-energy, and low-performance smart transducer was evaluated.

4.3.1 Device under test

The countermeasures under analysis are: i) random delay insertion, ii) random SBox, and iii) boolean masking. Moreover, also the configuration with no countermeasures was evaluated. All the considered countermeasures are applied at algorithmic level on the source code of the Advanced Encryption Standard algorithm with a 128-bits key length.

Random delay insertion and random SBox are two implementations of hiding. The former introduces a misalignment by means of randomly occurring delays, the latter randomizes the execution of SBox in AES. The randomization of

Sbox operations are possible as the operations are independents each others. The boolean masking consists in applying a mask to the plaintext value at the beginning of the AES. When the mask is applied, the AES round have to keep tracks of the mask, and, at the end, the mask has to be removed from the output. The architeture of masked AES is shown in Fig. 4.6.

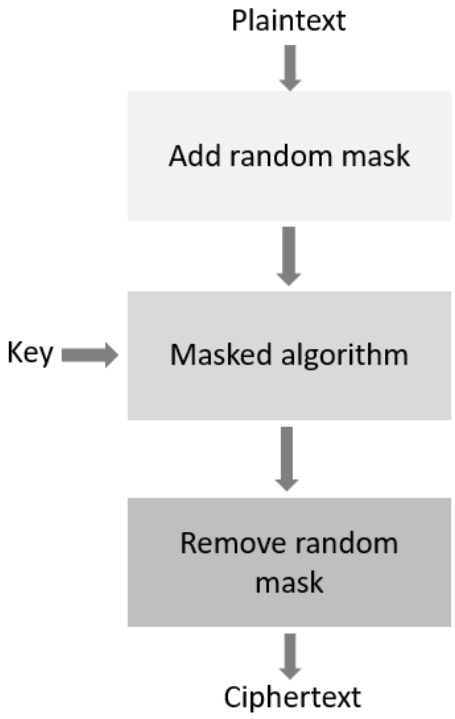


FIGURE 4.6: Masked Advanced Encryption Standard.

The above countermeasures are implemented in a software version of the AES-128, running on the ATMega-163, a low-power CMOS 8-bit microcontroller based on the AVR architecture [48], with 16K Bytes In-system Flash, 512 Bytes EEPROM, 1024 Bytes Internal SRAM, and 8 MHz maximum clock.

4.3.2 Execution of power attacks

The power attacks were conducted against the microcontroller implementing the AES-128 in absence of countermeasures and for each countermeasure under analysis. For the acquisition phase, 50 MSa/s was considered as sampling frequency. For the pre-processing phase, 400 as filter weight and 1 MSa/s as resampling frequency were chosen. Moreover, 500 power traces were acquired when no countermeasures were inserted while 30,000 is the number chosen for the configurations in presence of countermeasures.

The statistical analysis of the attacks was realized with the Inspector Tool by Riscure. The tool receives in input a number of power traces where the key is used and it computes the mean differences for the DPA and the correlation coefficients for the CPA for each hypothesis of the key k. For each key byte hypothesis, the mean differences are identified as $\boldsymbol{DPA_k}$ while the correlation coefficients

correspond to the rows $\boldsymbol{r_k}$ of the correlation matrix $\boldsymbol{R}$. Finally, it returns the key hypothesis most likely to be the actual one. In general, very high values in a difference of mean related to the actual key byte, $\boldsymbol{DPA^*}$, maximize the likelihood that the related key byte hypothesis coincides with the secret key byte. In the same way, high values of correlation coefficients in the row of the **R** matrix related to the actual key byte, $\boldsymbol{r}^*$, maximize the likelihood that the related key byte hypothesis coincides with the secret key byte. Therefore, the tool chooses the key byte hypothesis most likely to be the actual one by considering the highest values in a difference of mean for the DPA and highest values of correlation coefficients for the CPA.

To better understand the criterion for choosing the key byte hypothesis most likely to be the actual one, the DPA and CPA statistical analysis was implemented in MATLAB in order to plot the difference of means $\boldsymbol{DPA_k}$ and the correlation coefficients $\boldsymbol{r_k}$ for all the key byte hypothesis k. The Fig. 4.7 shows the overlapped plots of $\boldsymbol{DPA_k}$, where $k \in [0, 255]$ is the hypothesis of the first key byte, after 100 power traces with (a) no countermeasures and (b) masking. The $\boldsymbol{DPA}^*$, corresponding to the correct key byte hypothesis, is plotted in green, whereas the others in blue. In absence of countermeasures, Fig. 4.7(a), a spike is evident respect to the others values in the same pattern and respect to the patterns of others key hypothesis, allowing to correctly identify the key byte. Conversely, when masking countermeasure is adopted, Fig. 4.7(b), no value of $\boldsymbol{DPA}^*$ is the maximum among the values in $\boldsymbol{DPA_k}\ \forall\ k$, preventing the DPA attack to success.
Analogously, the Fig. 4.8 shows the overlapped plots of correlation coefficients $\boldsymbol{r_k}$, where $k \in [0, 255]$ is the hypothesis of the first key byte, after 100 power with (a) no countermeasures and (b) masking. When no countermeasures are inserted, Fig. 4.8(a), the CPA attack is able to correctly identify the key byte by considering the maximum value in $\boldsymbol{r}^*$ respect to the values in $\boldsymbol{r_k}\ \forall\ k$. Conversely, when masking is inserted, 4.8(b), no values in $\boldsymbol{r}^*$ is the maximum among $\boldsymbol{r_k}$, preventing the CPA attack to success in discovering the correct key byte.

4.3.3 Minimum number of power traces

For the calculation of the minimum number of power traces needed to discover the secret key, five attack repetitions were implemented for each of the eight combinations between attacks and countermeasures. The exploratory ranges for the first repetition were [50, 500] for no-countermeasures, [$min_{A,1,1}$, 1000] for random delay, [$min_{A,1,1}$, 20000] for random SBox, and [$min_{A,1,1}$, 30000] for masking, where $min_{A,1,1}$ is the minimum number of power traces needed to success under a specific attack A during the first repetition $R = 1$ without countermeasures $C = 1$. Each of the successive four repetitions searched for the minimum number of power traces starting at $min_{A,C,1}$ found in the first repetition of a specific combination of attack and countermeasure with a step in increment of 10 until the secret key was not fully recovered, and decreasing by 2 until the minimum number of power traces $min_{A,C,R}$ was identified. The experimental results, consisting of the minimum number of power traces needed to discover the secret key in a particular configuration of attack and countermeasure, are obtained as the mean

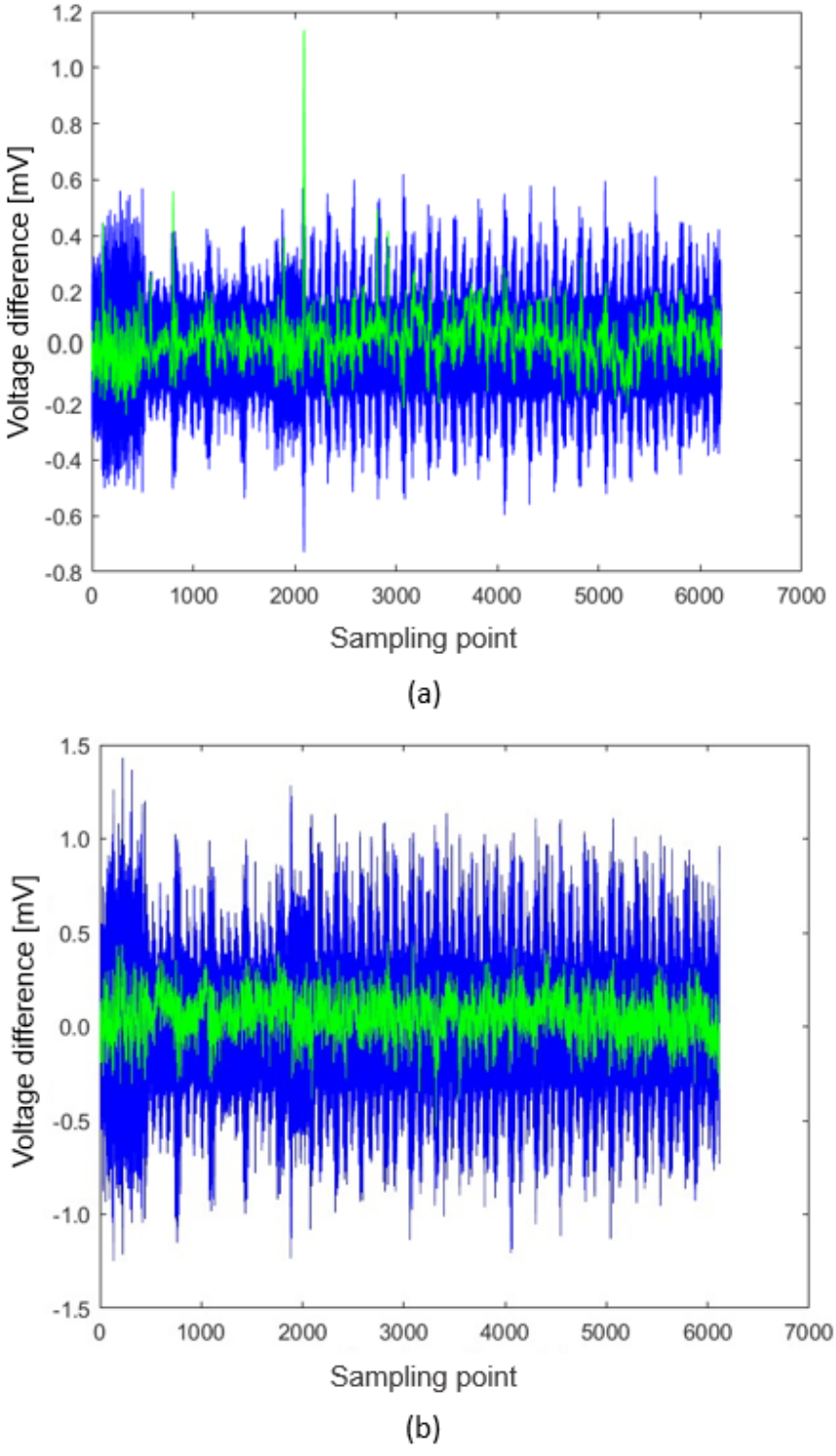


FIGURE 4.7: The overlapped plots of $\boldsymbol{DPA_k}$ ($k \in [0, 255]$ is the hypothesis of the first key byte) after 100 power traces with (a) no countermeasures and (b) masking. The $\boldsymbol{DPA}^*$, corresponding to the correct key byte hypothesis, is plotted in green, whereas the others in blue. In case of masking no value of $\boldsymbol{DPA}^*$ is the maximum with respect to the values in $\boldsymbol{DPA_k}$.

of the minimum number of power traces found out in the five repetitions. These values are summarized in Table 4.1 at varying the attacks and countermeasures. In particular, the Table reports the means and the 1-σ repeatability of power traces in each configuration of attack and countermeasure. The secret key is found with the lower number of power traces in absence of countermeasures. The random delay increases the number of power traces respect to the absence of countermeasures of 20 power traces for the DPA and 57 power traces for the CPA, while the random SBox introduces an increase of the order of thousands. In case of masking, 30,000 power traces resulted not sufficient to discover the secret key, therefore a higher number should be explored. In general, the results demonstrated how power countermeasures enhance the security of the IoT microcontroller. Indeed, each countermeasure increases the number of power traces needed to discover the secret key. The most effective countermeasure resulted the masking, as exhibits the highest number of power traces and, consequently, the greatest effort for attacking.

TABLE 4.1: Mean number and 1-σ repeatability of power traces for discovering the secret key at varying the attacks and countermeasures.

	DPA		CPA	
	Mean	1-σ	Mean	1-σ
None	$1.1 \cdot 10^2$	$1 \cdot 10$	$7.3 \cdot 10$	$0.9 \cdot 10$
Random delay	$1.3 \cdot 10^2$	$3 \cdot 10$	$1.3 \cdot 10^2$	$2 \cdot 10$
Random SBox	$29.0 \cdot 10^3$	$8 \cdot 10^2$	$10.1 \cdot 10^3$	$7 \cdot 10^2$
Masking	30000^a		30000^b	

(a): the number of power traces allowing to discover a mean of 1 byte on 5 repetitions with a 1-σ repeatability of 1

(b): the number of power traces allowing to discover a mean of 1 byte on 5 repetitions with a 1-σ repeatability of 1

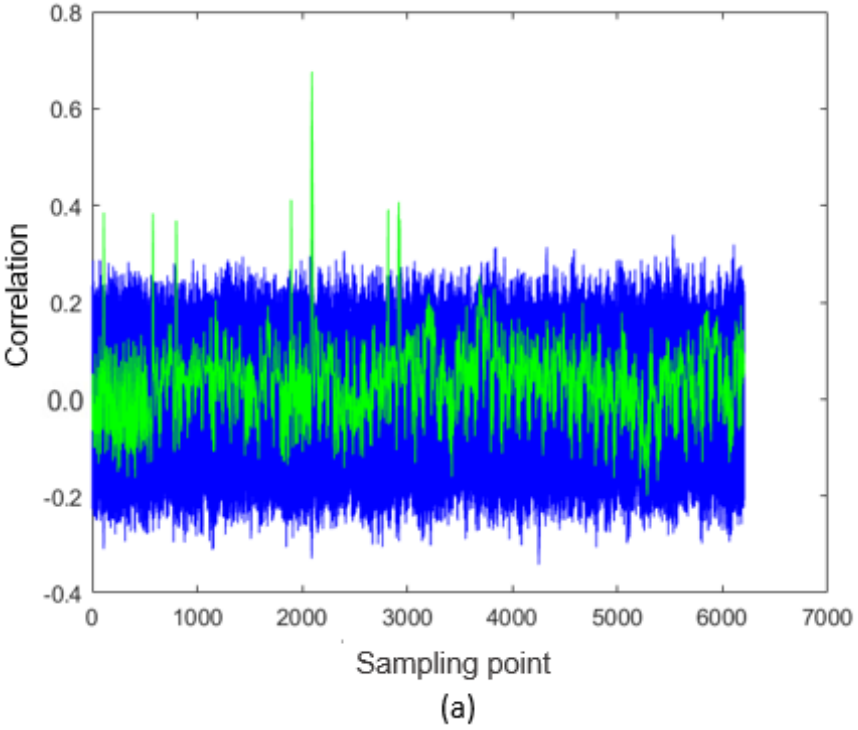
4.3.4 Countermeasure assessment

The minimum numbers of power traces needed to discover the secret key obtained by each combination of attack and countermeasure are employed to compute the strength factors (SFs) according to the eq. 4.1. The SF values for each countermeasure are reported in Table 4.2. Random delay strengthens the AES of a 1.3 factor with respect to no countermeasure condition; the strengthening factor for random SBox is 208, while more than 318 for masking. In case of masking, the non availability of a minimum number of power traces does not allow to determine a strength factor value but only a low limit.

The comparison of strength factors highlights masking as the most effective over other power countermeasures as it increases the number of power traces needed to success in the attack to a greater extent than the other countermeasure.

TABLE 4.2: Strength factor for each countermeasure.

Countermeasures	Strength Factor (SF)
None	1
Random delay	1.3
Random SBox	208
Masking	»318



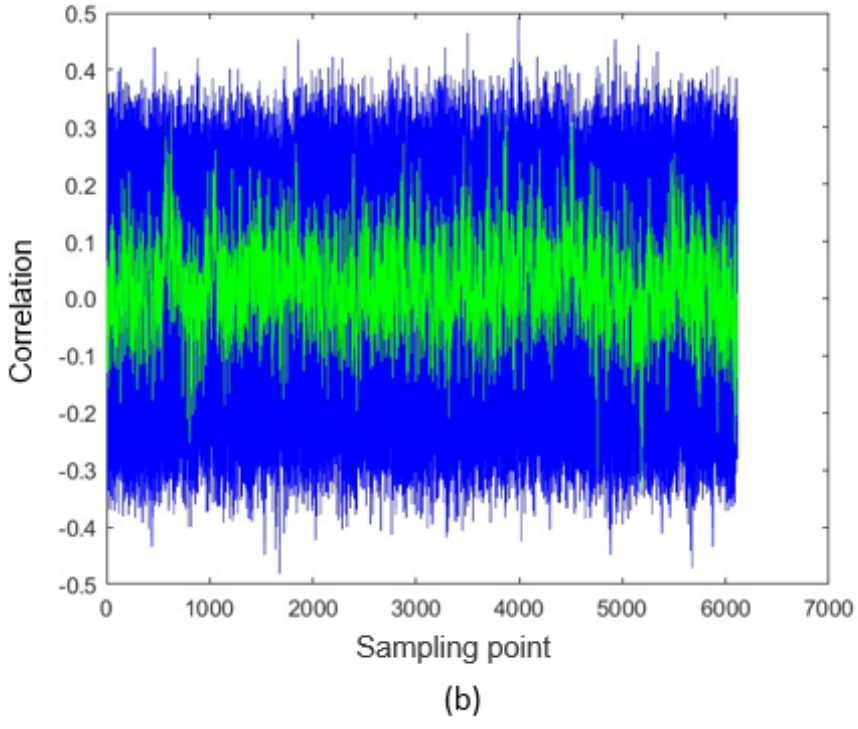


FIGURE 4.8: The overlapped plots of correlation coefficients $\boldsymbol{r_k}$ ($k \in [0, 255]$ is the hypothesis of the first key byte) after 100 power traces with (a) no countermeasures and (b) masking. The $\boldsymbol{r}^*$, corresponding to the correct key byte hypothesis, is plotted in green, whereas the others in blue. In case of masking no value of the vector $\boldsymbol{r}^*$ is the maximum with respect to all the values contained in the other k_{th} vectors $\boldsymbol{r}$.

Chapter 5

Timing attack on IoT transducers

In this chapter, another typology of side-channel attack, the timing attack, is explored. Unlike the power analysis attack, this attack exploits the execution times to reveal the secret key of a cryptographic algorithm. In particular, an experimental test of robustness to timing attack is reported for the asymmetric cryptographic algorithm most used in Internet of Things framework, i.e. the Elliptic Curve Digital Signature Algorithm (ECDSA). The ECDSA function of the *MbedTLS*, the widely used cryptographic library, is implemented in an IoT microcontroller and its robustness is demonstrated experimentally by measuring the execution times of the cryptographic algorithm and applying a classical method proposed in literature, based on the use of a mathematical object known as *lattice*. The robustness test method applied on the MbedTLS ECDSA function does not have success in revealing the secret key of the cryptographic algorithm, demonstrating the robustness of the ECDSA implementation under test.

5.1 Problem statement

In an Internet of Things framework, the resource constraints of smart transducers, such as limited amount of energy, short communication range, low bandwidth, limited processing, and reduced memorization capacity [79], have favoured, for a long time, the symmetric cryptography as solution to provide security. Indeed, it boasts speed of computation and low energy cost. Contrarily, asymmetric cryptography was not considered suitable due to its long duration. Elliptic curve cryptography (ECC) was the first approach to public key cryptography used in an IoT framework. In fact, the small keys, employed by these algorithms, result in less time, storage, bandwidth, and power consumption for the computation of their operations.

One application of the asymmetric key cryptography is the digital signature, able to (i) authenticate the message (the message comes from the sender), (ii) provide integrity (the message has not been modified), and (iii) guarantee non-repudiation (the sender cannot lie afterward about not having signed anything). The widely used elliptic curve-based protocol is the Elliptic Curve Digital Signature Algorithm (ECDSA). Many researches proved that this algorithm is suitable in constrained-source environment [80, 81]. In fact, the ECDSA exhibits small cost of computation compared with other public key cryptography algorithms, such as Rivest Shamir Adleman (RSA), traditional Digital Signature Algorithm

(DSA), and ElGamal [82]. As an example, ECDSA with a 256-bit key offers the same level of security for the RSA algorithm with a 3072-bit key. For this reason, many embedded cryptographic libraries (MbedTLS, GnuTLS, and wolfSSL) offer asymmetric cryptographic functions to provide secure communication for IoT, smart grid, connected home, automobiles, and more [16, 83].

Many researches try to break the security offered by such algorithm. In particular, the ECDSA was demonstrated vulnerable to side-channel attacks. An electromagnetic attack against the ECDSA signature implemented on mobile devices using a function of the OpenSSL cryptographic library is reported in [84]. In [85], the same library used for X86 processor architecture was proved vulnerable by a FLUSH+RELOAD cache side-channel attack. Also a fault attack [86] and a timing attack [87] were successfully addressed against ECDSA signature. In particular, the side channel attacks recover some information on the ephemeral key and a more sophisticated method, namely lattice attack, exploits the recovered information to reveal the secret key.

Because a straightforward implementation of asymmetric cryptographic algorithms could not be enough to guarantee security, cryptographic libraries have to implement several countermeasures. As concern the timing attack, it exploits cryptosystem implementations that do not run in constant time: their execution times measured by the attacker are somehow state-dependent and hence key-dependent. Therefore, the related countermeasure consists of making the implementations that are dependent by the key constant or random.

In this chapter, an experimental test of robustness to timing attack is reported for the ECDSA signature function of the widely used cryptographic library in IoT framework, namely MbedTLS. To this aim, a timing-lattice attack is addressed against an ARM microcontroller implementing the ECDSA digital signature of the library under test. Times are exploited to derive some information on the *ephemeral key* used in the ECDSA signature algorithm and the more sophisticated method, namely lattice attack, is implemented to reveal the secret key from the recovered information.

5.2 Timing-lattice attack method

The robustness is tested by a timing-lattice attack aimed to discover the secret key of an IoT microcontroller implementing the ECDSA digital signature. The timing attack exploits execution times of the signature generation to obtain information on the ephemeral key. Indeed, the ECDSA algorithm, in addition to the secret key, makes use of an ephemeral key (or nonce) that changes randomly in subsequent runs of the algorithm. Known parts of the ephemeral keys for multiple signatures are, then, employed by a lattice attack to reveal the secret key.

The timing-lattice attack method can be distinguished in three phases: (i) collection, (ii) filtering, and (iii) lattice-attack. A block flow is illustrated in the Figure 5.1. During the collection phase, enough pairs of signatures are collected from repeatedly runs of the ECDSA signature generation. In particular, the time intervals required to sign the messages, the messages themselves, and the signatures are acquired and are saved in a file. The execution times are exploited in

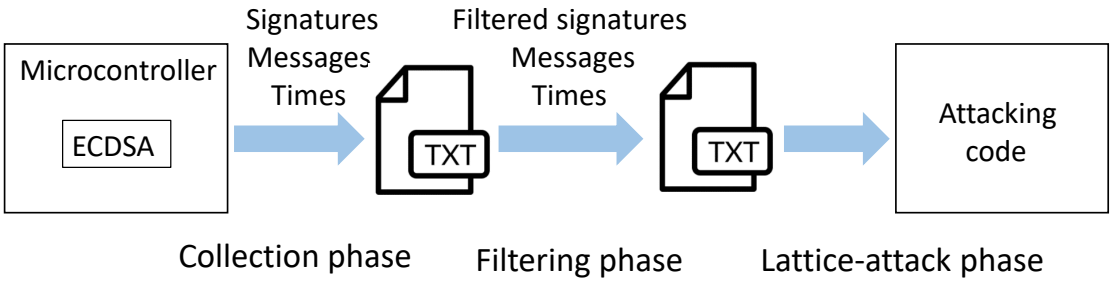


FIGURE 5.1: Diagram flow of timing-lattice attack.

the filtering phase to implement a timing attack. Based on a time dependency, this phase selects a small set of signatures from those collected, characterized by a nonce k with the most significant bits equal to zero. Indeed, when the most significant bits in the nonce are equal to zero, the signature calculation should be faster. Therefore, selecting only the fastest signatures, there is a good probability that the nonces used for their generation have a certain number of most significant bits equal to zero. The selected signatures are used to mount a lattice attack, as described in [88], with the goal of recovering the secret key used to generate the ECDSA signatures.

5.2.1 Lattice attack phase

The lattice attack was implemented following the method proposed by D. Wong in [88]. In the following, the main passages are reported. Consider Z_v as the set of signatures generated by using particularly "small" nonces k_i, that is nonces with a certain number of most significant bits equal to zero. Each signature $(r_i, s_i) \in Z_v$ can be expressed mathematically as:

$$r_i = k_i G \text{ (mod n)}$$
$$s_i = k_i^{-1}(H(m_i) + dr_i) \text{ (mod n)}$$

where m_i is the message, H is the hash function, $H(m_i)$ is the digest of the message, G is the generator point, n is the order of the generator, and d the secret key. Since the $H(m_i)$ is known, because the messages are known and the hash function is published in a standard, the second equation can be written as:

$$H(m_i) = s_i k_i - dr_i \text{ (mod n)}$$

As the attack goal is to find the nonces and, subsequently, the secret key, the latter has to be removed from the equations. Therefore, one of the equation has to be used to remove d from the others. The first equation is chosen. In particular, it can be written as:

$$H(m_0) = s_0 k_0 - dr_0 \text{ (mod n)}$$
$$\forall \text{ i, } H(m_0) r_0^{-1} r_i = s_0 k_0 r_0^{-1} r_i - dr_i \text{ (mod n)}$$

Since all the r_i are known, the second equation can be computed and used to remove the private key d from all the other equations:

$$\forall \text{ i} \neq 0, H(m_i) - H(m_0)r_0^{-1}r_i = s_i k_i - s_0 k_0 r_0^{-1} r_i \text{ (mod n)}$$
$$(H(m_i) - H(m_0)r_0^{-1}r_i)s_i^{-1} = k_i - k_0 s_0 r_0^{-1} r_i s_i^{-1} \text{ (mod n)}$$

In this way, $v-1$ equations with only two unknowns, the nonces k_i and the nonce of the first equation k_0, are obtained. These nonces should have around the same size, which is relatively small. The $v-1$ equations can be written as:

$$k_i + A_i k_0 + B_i = 0 \text{ (mod n)}$$

On a set of modular equations the lattice can result useful. In fact, D.Wong in his work [88], shapes the equations to obtain the main lattice problems, i.e. CVP (*Closest Vector Problem*) or SVP (*Shortest Vector Problem*), in order to find the solutions in the set of equations. In particular, he transforms the system into a matrix form:

$$\begin{pmatrix} k_0 \\ k_1 \\ \vdots \\ k_{v-1} \end{pmatrix} = \begin{pmatrix} -1 & & & \\ A_1 & n & & \\ \vdots & & \ddots & \\ A_{v-1} & & & n \end{pmatrix} \begin{pmatrix} -k_0 \\ z_1 \\ \vdots \\ z_{v-1} \end{pmatrix} - \begin{pmatrix} 0 \\ B_1 \\ \vdots \\ B_{v-1} \end{pmatrix}$$

In this way, it is evident the instance of the *Closest Vector Problem*. In fact, finding a vector from the lattice, spanned by the columns of the above v x v matrix, that is closed to non-lattice vector $(0, B_1, \ldots, B_{v-1})$, it is possible calculate the coefficients vector $(-k_0, z_1, \ldots, z_{v-1})$ and so compute the nonces vector $(k_0, k_1, \ldots, k_{v-1})$. If a nonce is known, the secret key is revealed. In fact from the s equation of ECDSA:

$$d = skr^{-1} - H(m)r^{-1} \text{ (mod n)}$$

To solve the CVP problem, D. Wong employs the *Embedding Strategy*. This technique reduces the CVP problem to the SVP problem and directly applies the *lattice basis reduction* algorithm, known as LLL. The Embedding Strategy succeeds in the objective by adding the non-lattice vector B in the basis. In this way, it is part of the lattice and it is possible to apply the reduction algorithm. The lattice is spanned by the columns of the following matrix:

$$\begin{pmatrix} -1 & & & & B_0 \\ A_1 & n & & & B_1 \\ \vdots & & \ddots & & \vdots \\ A_{v-1} & & & n & B_{v-1} \\ 0 & 0 & \ldots & 0 & \frac{n}{2^{l-1}} \end{pmatrix}$$

with l equal to the number of most significant bits known to be zero in the nonces. The reduction algorithm applied to this matrix could provide $B-k$ or $k-B$ (B is the non-lattice vector and k is the lattice vector) which is in the lattice and should be very small. From the reduction algorithm result, it is possible to determine k knowing B.

5.3 Case study

The proposed experimental test of robustness to timing-lattice attack is reported for the ECDSA signature function of the widely used cryptographic library in IoT framework, namely MbedTLS, implemented in an IoT microcontroller.

5.3.1 Device under test

The device under test (DUT) is the ARM Cortex-M4 STM32L496ZG microcontroller, embedded in the NUCLEO-L496ZG board. The microcontroller presents high-speed memories (up to 1 Mbyte of Flash memory, 320 Kbyte of SRAM), a core operating at a frequency of up to 80 MHz, and implements the Elliptic Curve Digital Signature (ECDSA) to sign messages using MbedTLS library, the most used in IoT framework.

MbedTLS is a C library that implements cryptographic primitives, X.509 certificate manipulation and the SSL/TLS and DTLS protocols. Its small code footprint makes it suitable for embedded systems. Indeed, with the minimum complete TLS stack, it requires under 60 kB of program space and under 64 kB of RAM.

5.3.2 Collection phase

For the collection phase, a routine that invokes the MbedTLS function several times to generate the ECDSA digital signature of a message, and accurately measure the time required by each signing operation was created. The Mbed Studio tool was used to implement and deploy the source code on the microcontroller. On the base of the executable code, the microcontroller implements repeatedly the digital signatures and, by each signature generation, it measures the time required for the signing operation. Such measure is performed by the microcontroller timer with a resolution of micro-seconds. Moreover, the microcontroller communicates with the PC by means of a USB protocol allowing to save the execution times, the signatures, and the messages in a text file. The Figure 5.2 shows the NUCLEO-L496ZG board that communicates with the PC by means a USB connection.

A flowchart of the code for the collection phase is reported in Figure 5.4. Before calling the library function $ecdsa\_write\_signature$ to implement the ECDSA signatures generation, the curve and the key pair were set. The curve is the NIST P-256, while the ECDSA key pair was generated by $mbedtls\_ecdsa\_genkey$ function, using a Deterministic Random Byte Generator (DRGB). The values of private and public keys are generated only once and used for all the signatures. As the time needed to generate a signature does not depend on the value of the message, the same message was used in the signing function. This choice avoids to generate a new random message for each signature, resulting in simplicity and speed of the experiment. Then, the digest of the message is computed. The SHA-256 was chosen. After a preliminary phase of configuration, a routine invoking the MbedTLS $ecdsa\_write\_signature$ several times in order to generate the

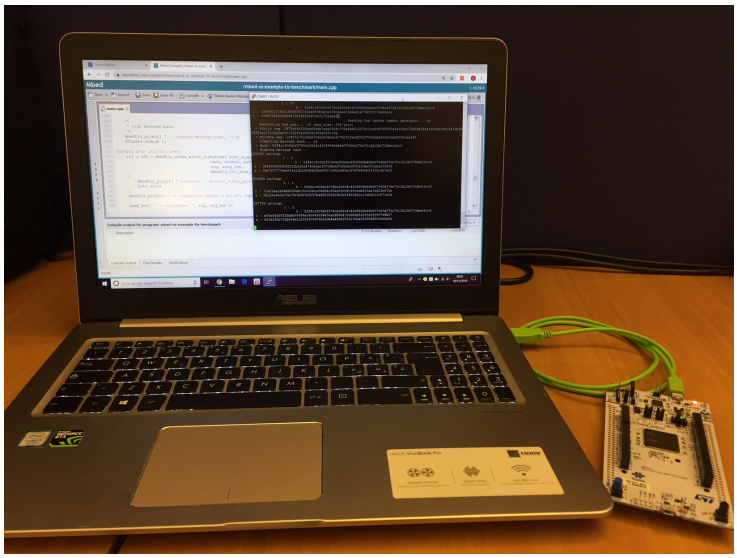


FIGURE 5.2: NUCLEO-L496ZG board including the ARM Cortex-M4 STM32L496ZG microcontroller while communicates with a PC by means of USB protocol.

ECDSA signatures of a message, and accurately measure the time required by each signing operation is used. A number of 10,000 repetitions was chosen. Definitely, the implemented routine takes as input the message to be signed and the private key, and returns the signature, the time required to compute the ECDSA signature and the hash of message, known as digest, that was pre-calculated before the signing routine. The microcontroller communicates the output values to PC by means of a USB protocol, that are saved in a text file. The saved information will be analyzed in the filtering phase.

An extract of the text file, containing the ECDSA key pair, the digest of the message, the digital signatures, and the execution times is illustrated in Figure 5.3. The information about the private key is reported in the text file but it is not used for the implementation of the timing-lattice attack.

```
# Private
# SEC_DS =      0xc504247e2dbed71944808357d65f39d5768ab99c35bc22e0aa4eb5cbf42281f1
# PUB_DS_Qx = 0xd32703defe97d57f1bd5a7b63e0b151dfed35e39a35ee416a587d34d413e5cc2
# PUB_DS_Qy = 0x0673842a2c00910b8d578d86a269d6462386c4e289642abacbb9e34c85559c96
#
t : 299767
h : 02f4ccf09d1f5740e28864c430f866b46b697646d376e7813d12b873dbe16cc6
r : 6ef810bd8d5c6c3185674b9d8240bcde3a39898a3989b3ac5f584e3c6be16599
s : 8689a16af3c17e22be61393d26d44b2910a0818885b986be447aed21dc52ff24
t : 298241
h : 02f4ccf09d1f5740e28864c430f866b46b697646d376e7813d12b873dbe16cc6
r : 13c1b9ad185d21f0c2097847456fb80d7c0f1a130ca8be25a01174084f4438ad
s : 8d77a597fb0a02313e983e9ef2f3f11eddc370fae89ea2648de4b6768e018f89
```

FIGURE 5.3: Extract of text file generated by the collection phase. SEC_DS, PUB_DS: ECDSA key pair, (r,s): digital signatures, h: digest of the message, and t: signing times in μs.

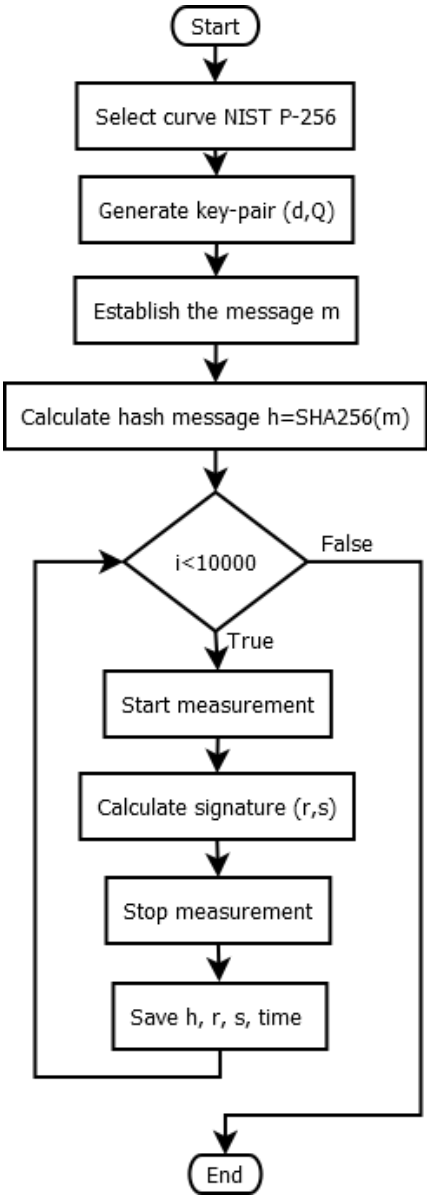


FIGURE 5.4: Flowchart of the code for the collection phase.

5.3.3 Filtering phase

For the generation of the ECDSA signature on a NIST P-256, an ephemeral key k of 256 bit is used. This value is randomly generated in the interval $[1, n-1]$ with n the order of the generator. When the most significant bits in k are equal to zero, the signature calculation should be faster. Therefore, selecting only the fastest signatures we expect that they were generated using a nonce with a certain number of most significant bits equal to zero. Based on this behaviour, in the filtering phase, only the fastest signatures are picked up from the text file obtained by the collection phase. In particular, the signatures between two timing bounds are selected.

The Figure 5.5 shows an occurrences histogram of the signing times. The fastest signatures was obtained in 272933 μs, while the slowest in 311465 μs. The

distribution of the signatures has a mean value of 288852 μs and a standard deviation of 7 μs. For the filtering phase, the signatures whose execution times belong

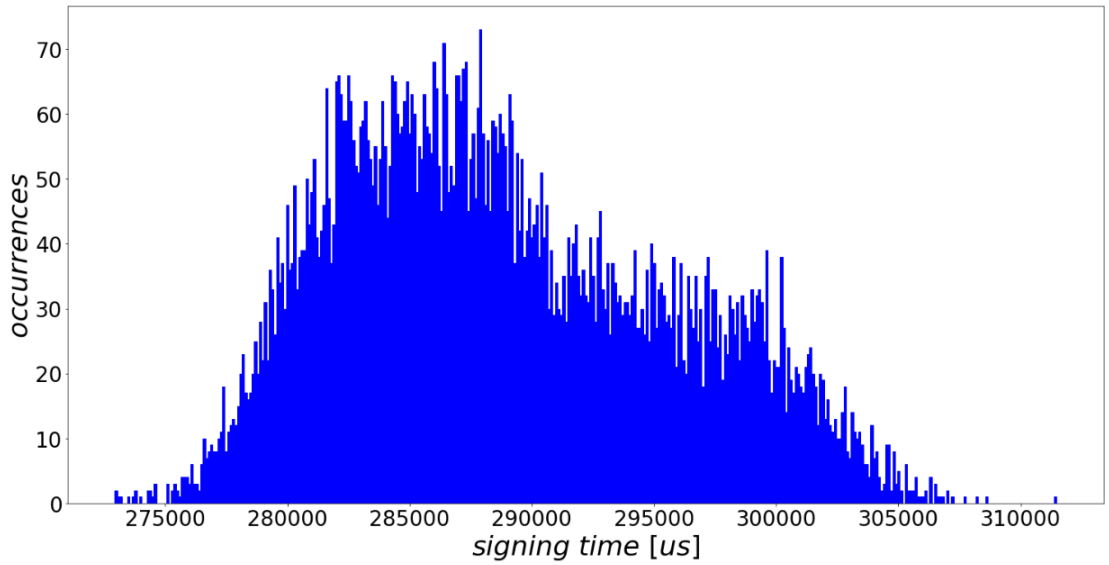


FIGURE 5.5: Occurrences histogram of the signing times.

to a certain timing interval were selected. The lower limit of the interval was set at the minimum signing time while the upper limit is iteratively increased from the minimum time value to the mean value of the time distribution. As an example, for the time interval $[272933, 288852]$ μs, a number of 103 signature was selected.

For each time interval considered, the filtered signatures were saved in a text file and the lattice attack, as described in Section 5.2.1, was applied on them, hypothesizing that the filtered signatures were generated with a nonce with 8 most significant bits equal to zero. SageMath, an Open-Source Mathematical Software System, was used to filter the digital signature and to implement the lattice attack.

5.3.4 Experimental results

The timing-lattice attack was applied on the filtered signatures by each time interval considered. From each experiment, the nonce values are determined and used to recover the secret or public key. In fact, from the equation of the s component of the digital signature:

$$s_i = k_i^{-1}(H(m_i) + dr_i) \text{ (mod n)}$$

it is possible to explicit the d secret value:

$$d = s_i k_i r_i^{-1} - H(m_i) r_i^{-1} \text{ (mod n)}$$

As in this equation all the parameters are known (the component r and s of the digital signature, the message, and the digest), the value of the secret key can be computed.

For verifying that the key disclosed is the actual value of the private key of the microcontroller, the secret key d produced by each lattice attack is multiplied with the generator point G and the result is compared with the public key. In fact, the public key is computed as:

$$Q = dG$$

In all the timing intervals considered, the lattice was not able to find the secret key.

The obtained result proves the robustness of the ECDSA of the MbedTLS library to the timing-lattice attack on 10,000 ECDSA signatures.

5.3.5 Discussion

Further analysis were conducted in order to understand the reason for the failure of the timing-lattice attack against the ECDSA function implemented in the library MbedTLS. In particular, the relationship between the bitsize of the nonces and the signing time was investigated.

From the s equation of ECDSA, it is possible to derive the mathematical expression for the nonce k:

$$k = H(m)s^{-1} + drs^{-1} \text{ (mod n)}$$

In this equation, the signatures and the digest of the message are known. Since the key pair was generated at the beginning, also the value of the private key is known. It is better to clarify that the value of the private key is known but it was not used in the timing-lattice attack. It is employed only to investigate the cause for the failure of the attack. For the above reasons, the values of k can be computed. From the value of k, it is also possible to obtain the number of bits in the nonces.

The distribution of the nonce bitsize as a function of the signing time is reported in Figure 5.6. It is evident that there is not a correlation between the number of bits in the nonces and the signing time. In other words, the signatures generated by nonces with the most significant bits equal to zero are not faster than the signatures obtained with the highest nonce bitsize. This means that selecting the faster signatures do not correspond to take the signatures obtained with nonce with a certain number of most significant bits equal to zero. Therefore, the hypothesis where the lattice attack is founded is not fulfilled and this prevent the attack to success.

This further result obtained confirms the robustness of the MbedTLS library from the timing-lattice attack. In addition, it also allows to understand what does not guarantee the success of the attack. The lacking of a dependence of the signing times from the bitsize in the nonces can be attribute to the presence of countermeasures making the implementation of the ECDSA signature generation independent from the actual nonce values.

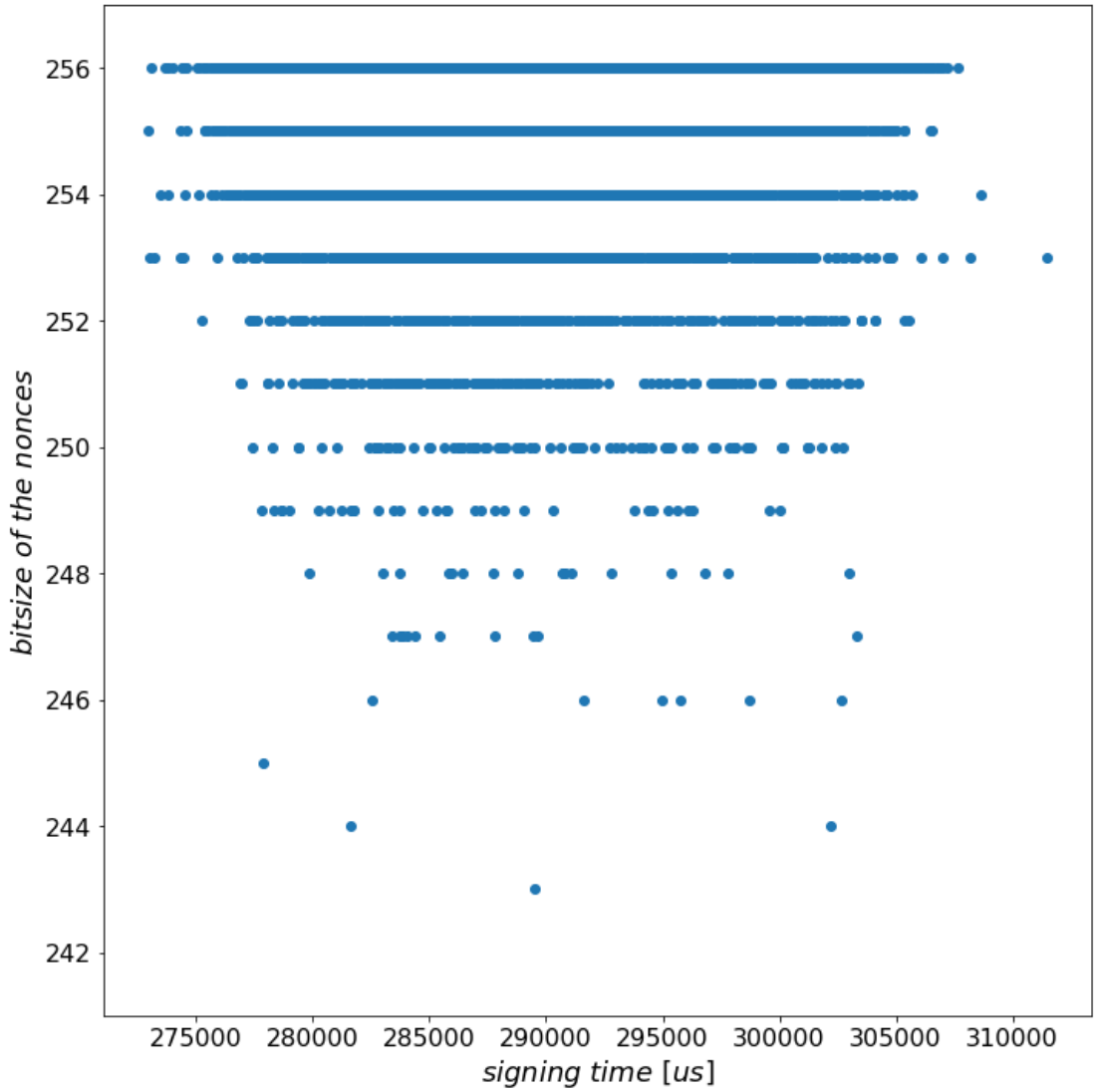


FIGURE 5.6: Distribution of the number of bits in the nonces k as a function of the signing times for 10,000 ECDSA signatures.

Conclusions

In this thesis, three aspects of the IoT security against side-channel attacks were faced: (i) improvement of power attack efficiency, (ii) improvement of Vulnerability Assessments tests, and (iii) techniques for robustness evaluation of cryptographic algorithms.

The chapter 2 addresses the improvement of power attack efficiency topic. The success rate of the power attacks is significantly affected by the signal-to-noise ratio (SNR) of the power traces. Power analysis attacks are very sensitive to the magnitude of these signals to recover the value of the secret key. Therefore, it is important to eliminate noise effectively and improve the SNR of power traces to extract the secret key with minor effort. A filtering operation was employed in order to enhance the test attack effectiveness. The power traces are filtered by a lowpass digital filter, which makes each sample a weighted average of the previous and the current sample. The proposed filter is able to enhance the effectiveness of the power attack. In fact, the use of a filter allows to discover the entire key unlike the lack of pre-processing with the same number of traces. Moreover, the filter is able to reduce the number of traces needed for a successful attack. Decreasing the number of power traces also reduces the time needed to find the secret key.

The chapter 3 proposes an improvement of power attack by identifying the best configuration of attacking parameters. The approach is based on the fractional experimental design, able to identify the best configuration with the lower number of experiments compared to other experimental designs favouring the reduction in experimental times and cost, without penalizing the statistical significance of the results. The best configuration, resulting in 500 for the filter weight, 500 kSa/s for the resampling frequency, and 400 for the number of power traces, contributes in reducing the effort of the attack in terms of resources and time. In other resampling frequency and filter weight configurations, 400 power traces are not enough to find the whole key. Therefore, the proposed method is able to identify the optimal configuration among the parameters, in order to obtain the least number of power traces necessary to discover the secret key. The results obtained by such approach highlights also an improvement in the repeatability and reproducibility. In fact, the optimum configuration deletes the 1-sigma repeatability respect to the other configurations and produces an enhancement in 1-sigma reproducibility by eliminating the contribution of uncertainty due to the variability of the considered parameters.

In the chapter 4, the improvement of Vulnerability Assessments tests was faced. The proposed method assesses the security of IoT devices at varying the power countermeasures by considering the minimum number of power traces needed to discover the secret key. In particular, a synthetic parameter is obtained for each countermeasure in order to quantify the security level introduced. This

parameter, identified as strength factor (SF) is obtained by normalizing the minimum number of power traces for each countermeasure to the minimum number of power traces needed to disclose the secret key in absence of countermeasures. Random delay strengthens the AES of a 1.3 factor with respect to no countermeasure condition; the strengthening factor for random SBox is 208, while more than 318 for masking. The comparison of strength factors highlights masking as the most effective over other power countermeasures as it increases the number of power traces needed to success in the attack to a greater extent than the other countermeasure.

Finally, the chapter 5 makes a robustness evaluation of the most used asymmetric cryptographic algorithm employed in Internet of Things framework from the timing attack. The the Elliptic Curve Digital Signature Algorithm (ECDSA) function of the widely used cryptographic library, namely MbedTLS, is implemented in an IoT microcontroller and its robustness is demonstrated experimentally by measuring the execution times of the cryptographic algorithm and applying a classical method proposed in literature, based on the use of a mathematical object known as lattice. The attack applied on 10,000 digital signatures obtained by repeated runs of the cryptographic algorithm does not have success in revealing the secret key highlighting the robustness of the ECDSA implementation under test.

References

[1] Tasneem Yousuf, Rwan Mahmoud, Fadi Aloul, and Imran Zualkernan. "Internet of Things (IoT) Security: Current status, challenges and countermeasures". In: *International Journal for Information Security Research (IJISR)* 5.4 (2015), pp. 608–616.

[2] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "The internet of things: A survey". In: *Computer networks* 54.15 (2010), pp. 2787–2805.

[3] Niharika Kumar. "IoT architecture and system design for healthcare systems". In: *2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon)*. IEEE. 2017, pp. 1118–1123.

[4] E Ray Dorsey and Eric J Topol. "State of telehealth". In: *New England Journal of Medicine* 375.2 (2016), pp. 154–161.

[5] Helen C Noel, Donna C Vogel, Joseph J Erdos, David Cornwall, and Forrest Levin. "Home telehealth reduces healthcare costs". In: *Telemedicine Journal & e-Health* 10.2 (2004), pp. 170–183.

[6] Melania F Bause, Hannah Forbes, Farnaz Nickpour, and Dirk Schaefer. "Towards a Health 4.0 Framework for the Design of Wearables: Leveraging Human-Centered and Robust Design". In: *Procedia CIRP* 91 (2020), pp. 639–645.

[7] Vania V Estrela et al. "Health 4.0: Applications, management, technologies and review". In: *Medical Technologies Journal* 2.4 (2018), pp. 262–276.

[8] Marco Bassoli, Valentina Bianchi, Ilaria De Munari, and Paolo Ciampolini. "An IoT approach for an AAL Wi-Fi-based monitoring system". In: *IEEE Transactions on Instrumentation and Measurement* 66.12 (2017), pp. 3200–3209.

[9] Sandro Pinto, Jorge Cabral, and Tiago Gomes. "We-care: An IoT-based health care system for elderly people". In: *2017 IEEE International Conference on Industrial Technology (ICIT)*. IEEE. 2017, pp. 1378–1383.

[10] Dimitra Azariadi, Vasileios Tsoutsouras, Sotirios Xydis, and Dimitrios Soudris. "ECG signal analysis and arrhythmia detection on IoT wearable medical devices". In: *2016 5th International conference on modern circuits and systems technologies (MOCAST)*. IEEE. 2016, pp. 1–4.

[11] Ionel Zagan, Vasile Gheorghita Gaitan, Adrian-Ioan Petrariu, and Adrian Brezulianu. "Healthcare IoT m-GreenCARDIO remote cardiac monitoring system–concept, theory of operation and implementation". In: *Advances in Electrical and Computer Engineering* 17.2 (2017), pp. 23–31.

[12] Luca Fanucci et al. "Sensing devices and sensor signal processing for remote monitoring of vital signs in CHF patients". In: *IEEE Transactions on Instrumentation and Measurement* 62.3 (2012), pp. 553–569.

[13] Yuan Jie Fan, Yue Hong Yin, Li Da Xu, Yan Zeng, and Fan Wu. "IoT-based smart rehabilitation system". In: *IEEE transactions on industrial informatics* 10.2 (2014), pp. 1568–1577.

[14] Emilio Sardini, Mauro Serpelloni, and Viviane Pasqui. "Wireless wearable T-shirt for posture monitoring during rehabilitation exercises". In: *IEEE Transactions on Instrumentation and Measurement* 64.2 (2014), pp. 439–448.

[15] Wen-Tsai Sung and Sung-Jung Hsiao. "The application of thermal comfort control based on Smart House System of IoT". In: *Measurement* 149 (2020), p. 106997.

[16] F Abate, M Carratù, C Liguori, and V Paciello. "A low cost smart power meter for IoT". In: *Measurement* 136 (2019), pp. 59–66.

[17] Amir H Alavi, Pengcheng Jiao, William G Buttlar, and Nizar Lajnef. "Internet of Things-enabled smart cities: State-of-the-art and future trends". In: *Measurement* 129 (2018), pp. 589–606.

[18] Joaquın Gutiérrez, Juan Francisco Villa-Medina, Alejandra Nieto-Garibay, and Miguel Ángel Porta-Gándara. "Automated irrigation system using a wireless sensor network and GPRS module". In: *IEEE transactions on instrumentation and measurement* 63.1 (2013), pp. 166–176.

[19] Haydar Ozkan et al. "A Portable Wearable Tele-ECG Monitoring System". In: *IEEE Transactions on Instrumentation and Measurement* (2019).

[20] Li Da Xu, Wu He, and Shancang Li. "Internet of things in industries: A survey". In: *IEEE Transactions on industrial informatics* 10.4 (2014), pp. 2233–2243.

[21] Shancang Li, Li Da Xu, and Shanshan Zhao. "5G Internet of Things: A survey". In: *Journal of Industrial Information Integration* 10 (2018), pp. 1–9.

[22] Jyoti Deogirikar and Amarsinh Vidhate. "Security attacks in IoT: A survey". In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE. 2017, pp. 32–37.

[23] Mukrimah Nawir, Amiza Amir, Naimah Yaakob, and Ong Bi Lynn. "Internet of Things (IoT): Taxonomy of security attacks". In: *2016 3rd International Conference on Electronic Design (ICED)*. IEEE. 2016, pp. 321–326.

[24] Hezam Akram Abdul-Ghani and Dimitri Konstantas. "A Comprehensive Study of Security and Privacy Guidelines, Threats, and Countermeasures: An IoT Perspective". In: *Journal of Sensor and Actuator Networks* 8.2 (2019), p. 22.

[25] Joan Daemen and Vincent Rijmen. "AES proposal: Rijndael". In: (1999).

[26] Paul C Kocher. "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems". In: *Annual International Cryptology Conference*. Springer. 1996, pp. 104–113.

[27] Paul Kocher, Joshua Jaffe, and Benjamin Jun. "Differential power analysis". In: *Annual International Cryptology Conference*. Springer. 1999, pp. 388–397.

[28] Yongdae Kim, Takeshi Sugawara, Naofumi Homma, Takafumi Aoki, and Akashi Satoh. "Biasing power traces to improve correlation power analysis attacks". In: *First international workshop on constructive side-channel analysis and secure design (cosade 2010)*. Citeseer. 2010, pp. 77–80.

[29] Wei Liu, Liji Wu, Xiangmin Zhang, and An Wang. "Wavelet-based noise reduction in power analysis attack". In: *2014 Tenth International Conference on Computational Intelligence and Security*. IEEE. 2014, pp. 405–409.

[30] Benjamin Hettwer, Stefan Gehrer, and Tim Güneysu. "Profiled power analysis attacks using convolutional neural networks with domain knowledge". In: *International Conference on Selected Areas in Cryptography*. Springer. 2018, pp. 479–498.

[31] Girish B Ratanpal, Ronald D Williams, and Travis N Blalock. "An on-chip signal suppression countermeasure to power analysis attacks". In: *IEEE Transactions on Dependable and Secure Computing* 1.3 (2004), pp. 179–189.

[32] Thomas Popp, Stefan Mangard, and Elisabeth Oswald. "Power analysis attacks and countermeasures". In: *IEEE Design & test of Computers* 24.6 (2007), pp. 535–543.

[33] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. "An AES smart card implementation resistant to power analysis attacks". In: *International conference on applied cryptography and network security*. Springer. 2006, pp. 239–252.

[34] Hans Delfs, Helmut Knebl, and Helmut Knebl. *Introduction to cryptography*. Vol. 2. Springer, 2002.

[35] Valtteri Niemi and Kaisa Nyberg. *UMTS security*. Wiley Online Library, 2003.

[36] Data Encryption Standard et al. "Data encryption standard". In: *Federal Information Processing Standards Publication* (1999), p. 112.

[37] Whitfield Diffie and Martin Hellman. "New directions in cryptography". In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654.

[38] Don Johnson, Alfred Menezes, and Scott Vanstone. "The elliptic curve digital signature algorithm (ECDSA)". In: *International journal of information security* 1.1 (2001), pp. 36–63.

[39] Neal Koblitz. "Elliptic curve cryptosystems". In: *Mathematics of computation* 48.177 (1987), pp. 203–209.

[40] Victor S Miller. "Use of elliptic curves in cryptography". In: *Conference on the theory and application of cryptographic techniques*. Springer. 1985, pp. 417–426.

[41] Douglas Robert Stinson and Maura Paterson. *Cryptography: theory and practice*. CRC press, 2018.

[42] Jean-Jacques Quisquater and David Samyde. "Electromagnetic analysis (ema): Measures and counter-measures for smart cards". In: *International Conference on Research in Smart Cards*. Springer. 2001, pp. 200–210.

[43] Dan Boneh, Richard A DeMillo, and Richard J Lipton. "On the importance of checking cryptographic protocols for faults". In: *International conference on the theory and applications of cryptographic techniques*. Springer. 1997, pp. 37–51.

[44] Jun Wu, Yiyu Shi, and Minsu Choi. "Measurement and evaluation of power analysis attacks on asynchronous S-box". In: *IEEE Transactions on Instrumentation and Measurement* 61.10 (2012), pp. 2765–2775.

[45] Hugues Thiebeauld, Georges Gagnerot, Antoine Wurcker, and Christophe Clavier. "Scatter: A new dimension in side-channel". In: *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer. 2018, pp. 135–152.

[46] Stefan Tillich, Christoph Herbst, and Stefan Mangard. "Protecting AES software implementations on 32-bit processors against power analysis". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2007, pp. 141–157.

[47] Paul Kocher, Joshua Jaffe, Benjamin Jun, and Pankaj Rohatgi. "Introduction to differential power analysis". In: *Journal of Cryptographic Engineering* 1.1 (2011), pp. 5–27.

[48] *ATMega-163 8 bit microcontroller*. URL: http://ww1.microchip.com/downloads/en/devicedoc/doc1142.pdf.

[49] *Riscure Inspector SCA*. URL: https://www.riscure.com/security-tools/inspector-sca/.

[50] Marco Bucci et al. "Enhancing power analysis attacks against cryptographic devices". In: *IET circuits, devices & systems* 2.3 (2008), pp. 298–305.

[51] Alain Merle and Jessy Clediere. "Security testing for hardware products: the security evaluations practice". In: *11th IEEE international On-line Testing Symposium*. IEEE. 2005, pp. 122–125.

[52] Eric Brier, Christophe Clavier, and Francis Olivier. "Correlation power analysis with a leakage model". In: *International workshop on cryptographic hardware and embedded systems*. Springer. 2004, pp. 16–29.

[53] Noura Benhadjyoussef, Hassen Mestiri, Mohsen Machhout, and Rached Tourki. "Implementation of CPA analysis against AES design on FPGA". In: *2012 International Conference on Communications and Information Technology (ICCIT)*. IEEE. 2012, pp. 124–128.

[54] Juan Ai, Zhu Wang, Xinping Zhou, and Changhai Ou. "Improved wavelet transform for noise reduction in power analysis attacks". In: *2016 IEEE International Conference on Signal and Image Processing (ICSIP)*. IEEE. 2016, pp. 602–606.

[55] Zeyu Wang, Wei Zhang, Peng Ma, and Xu An Wang. "Power consumption attack based on improved principal component analysis". In: *International Conference on Broadband and Wireless Computing, Communication and Applications*. Springer. 2019, pp. 787–799.

[56] Kealeboga Mpalane, Naison Gasela, BM Esiefarienrhe, and HD Tsague. "Vulnerability of advanced encryption standard algorithm to differential power analysis attacks implemented on ATmega-128 microcontroller". In: *2016 Third International Conference on Artificial Intelligence and Pattern Recognition (AIPR)*. IEEE. 2016, pp. 1–5.

[57] P Saravanan, Nithya Rajadurai, and P Kalpana. "Power analysis attack on 8051 microcontrollers". In: *2014 IEEE International Conference on Computational Intelligence and Computing Research*. IEEE. 2014, pp. 1–4.

[58] Utsav Banerjee, Lisa Ho, and Skanda Koppula. "Power-based side-channel attack for aes key extraction on the atmega328 microcontroller". In: *Computer Systems Security* (2015).

[59] Youssef Souissi, Sylvain Guilley, Jean-luc Danger, Sami Mekki, and Guillaume Duc. "Improvement of power analysis attacks using Kalman filter". In: *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2010, pp. 1778–1781.

[60] Lejla Batina, Jip Hogenboom, and Jasper GJ van Woudenberg. "Getting more from PCA: first results of using principal component analysis for extensive power analysis". In: *Cryptographers' track at the RSA conference*. Springer. 2012, pp. 383–397.

[61] Shyam Kumar Karna, Rajeshwar Sahai, et al. "An overview on Taguchi method". In: *International journal of engineering and mathematical sciences* 1.1 (2012), pp. 1–7.

[62] P Arpaia. "Experimental optimisation of flexible measurement systems". In: *IEE Proceedings-Science, Measurement and Technology* 143.2 (1996), pp. 77–84.

[63] Pasquale Arpaia and Nello Polese. "Uncertainty reduction in measurement systems by statistical parameter design". In: *6th IMEKO TC-4 Int. Symp., Lisboa*. 2001.

[64] *Statistical software JMP*. URL: https://www.jmp.com/it_it/software/data-analysis-software.html.

[65] Raghu N Kacker, Eric S Lagergren, and James J Filliben. "Taguchi's orthogonal arrays are classical designs of experiments". In: *Journal of research of the National Institute of Standards and Technology* 96.5 (1991), p. 577.

[66] Raquel Gómez-Chabla, Karina Real-Avilés, César Morán, Paola Grijalva, and Tanya Recalde. "IoT applications in agriculture: A systematic literature review". In: *2nd International conference on ICTs in agronomy and environment*. Springer. 2019, pp. 68–76.

[67] Jesús Martın Talavera et al. "Review of IoT applications in agro-industrial and environmental fields". In: *Computers and Electronics in Agriculture* 142 (2017), pp. 283–297.

[68] Tehreem Yaqoob, Haider Abbas, and Mohammed Atiquzzaman. "Security Vulnerabilities, Attacks, Countermeasures, and Regulations of Networked Medical Devices—A Review". In: *IEEE Communications Surveys & Tutorials* 21.4 (2019), pp. 3723–3768.

[69] H Fouad, Azza S Hassanein, Ahmed M Soliman, and Haytham Al-Feel. "Analyzing patient health information based on IoT sensor with AI for improving patient assistance in the future direction". In: *Measurement* (2020), p. 107757.

[70] George Mois, Silviu Folea, and Teodora Sanislav. "Analysis of three IoT-based wireless sensors for environmental monitoring". In: *IEEE Transactions on Instrumentation and Measurement* 66.8 (2017), pp. 2056–2064.

[71] Jun Yang, Zhengtai Xie, Li Chen, and Mei Liu. "An acceleration-level visual servoing scheme for robot manipulator with IoT and sensors using recurrent neural network". In: *Measurement* 166 (2020), p. 108137.

[72] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks: Revealing the secrets of smart cards*. Vol. 31. Springer Science & Business Media, 2008.

[73] Hamad Marzouqi, Mahmoud Al-Qutayri, and Khaled Salah. "Review of gate-level differential power analysis and fault analysis countermeasures". In: *IET Information Security* 8.1 (2013), pp. 51–66.

[74] Ioan Tudosa, Francesco Picariello, Eulalia Balestrieri, Luca De Vito, and Francesco Lamonaca. "Hardware security in IoT era: The role of measurements and instrumentation". In: *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4. 0&IoT)*. IEEE. 2019, pp. 285–290.

[75] Ali Galip Bayrak, Francesco Regazzoni, David Novo, and Paolo Ienne. "Sleuth: Automated verification of software power analysis countermeasures". In: *International Conference on Cryptographic Hardware and Embedded Systems*. Springer. 2013, pp. 293–310.

[76] Paul Kocher. "Design and validation strategies for obtaining assurance in countermeasures to power analysis and related attacks". In: *Proceedings of the NIST Physical Security Workshop*. Vol. 46. Citeseer. 2005.

[77] Konstantinos Nomikos et al. "On a Security-oriented Design Framework for Medical IoT Devices: The Hardware Security Perspective". In: *2020 23rd Euromicro Conference on Digital System Design (DSD)*. IEEE. 2020, pp. 301–308.

[78] Zahra Kazemi, Athanasios Papadimitriou, David Hely, Mahdi Fazcli, and Vincent Beroulle. "Hardware Security Evaluation Platform for MCU-based Connected Devices: Application to healthcare IoT". In: *2018 IEEE 3rd International Verification and Security Workshop (IVSW)*. IEEE. 2018, pp. 87–92.

[79] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. "Wireless sensor network survey". In: *Computer networks* 52.12 (2008), pp. 2292–2330.

[80] Arvinderpal S Wander, Nils Gura, Hans Eberle, Vipul Gupta, and Sheueling Chang Shantz. "Energy analysis of public-key cryptography for wireless sensor networks". In: *Third IEEE international conference on pervasive computing and communications*. IEEE. 2005, pp. 324–328.

[81] Haodong Wang and Qun Li. "Efficient implementation of public key cryptosystems on mote sensors (short paper)". In: *International Conference on Information and Communications Security*. Springer. 2006, pp. 519–528.

[82] Mishall Al-Zubaidie, Zhongwei Zhang, and Ji Zhang. "Efficient and Secure ECDSA Algorithm and its Applications: A Survey". In: *arXiv preprint arXiv:1902.10313* (2019).

[83] Marco Carratù, Matteo Ferro, Antonio Pietrosanto, and Vincenzo Paciello. "Smart Power Meter for the IoT". In: *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE. 2018, pp. 514–519.

[84] Daniel Genkin, Lev Pachmanov, Itamar Pipman, Eran Tromer, and Yuval Yarom. "ECDSA key extraction from mobile devices via nonintrusive physical side channels". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2016, pp. 1626–1638.

[85] Yuval Yarom and Naomi Benger. "Recovering OpenSSL ECDSA Nonces Using the FLUSH+ RELOAD Cache Side-channel Attack." In: *IACR Cryptology ePrint Archive* 2014 (2014), p. 140.

[86] Jörn-Marc Schmidt and Marcel Medwed. "A fault attack on ECDSA". In: *2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*. IEEE. 2009, pp. 93–99.

[87] Billy Bob Brumley and Nicola Tuveri. "Remote timing attacks are still practical". In: *European Symposium on Research in Computer Security*. Springer. 2011, pp. 355–371.

[88] David Wong. "Timing and Lattice Attacks on a Remote ECDSA OpenSSL Server: How Practical Are They Really?" In: *IACR Cryptology ePrint Archive* 2015 (2015), p. 839.

List of Figures

List of Tables