Université Catholique de Louvain
École Polytechnique de Louvain
Computer Engeneering Department

Analysis of XY Electromagnetic Radiations for Side-Channel Attacks

Promoter:
Standaert Francois-Xavier
Co-promoter:
Verleysen Michel
Reader:
Durvaux Francois
Reader:
van Oldeneel tot Oldenzeel Loïc

Master's thesis submitted for the graduation of
Master 120 ECTS in Computer Science
option Artificial Intelligence
by Dieuzeide Thomas

August 2015

Acknowledgements

Any senior student at university will be faced with its master thesis, its last and biggest work to show that the years of studying bore fruit. In the beginning, I knew I was capable of achieving this kind of project but the amount of work that was awaiting me was scary. But as time passed, my understanding and interest in the topic grew and made this work what it is in the end. That would not have been possible if it wasn't for the people whom I thank next.

First, I would like to thank my promoter, Francois-Xavier Standaert, for his time, for setting a good pace of work at the beginning of the year and for the relevance of his suggestions to refine the subject of my work. I would also like to thank Michel Verleysen for being my co-promoter.

Second, I wish to thank the two assistants and readers, François Durvaux and Loïc van Oldeneel tot Oldenzeel, for their great help along the year. Thank you both for giving a lot of time to explain and re-explain core concepts, review my algorithms and give me technical tips.

Also special thanks to Denis Flandre for reading my work and making interesting remarks.

Finally, I am thankful to my family and friends for their support.

Contents

List of Notations

General notations

X	a random variable
x	a specific realisation of the random variable X
$\mathcal{X}$	the set of all possible values for X
$\lvert\mathcal{X}\rvert$	the number of possible values for X
$f(X), F(X)$	functions of the random variable X
$\hat{x}, \hat{f}$	approximation or estimation of a variable/function
$\mathbf{X}$, $\mathbf{x}$	a vector of random variables
$Pr[X = x]$	probability that the value of X is x, also noted $Pr[x]$
$Pr[X = x \mid Y = y]$	conditional probability that the value of X is x knowing that $Y = y$

Cryptographic notations

$\mathtt{P}, \mathtt{C}, \mathtt{K}$	the plaintext, ciphertext and secret key
P,C,K	random variables corresponding to parts of $\mathtt{P}, \mathtt{C}$ and $\mathtt{K}$
$\mathtt{S}$	a S-box function (substitution table look-up)
L	the set of traces
l_i^y	the trace $l \in L$ corresponding to the value y and experiment i

Abbreviations

AES	Advanced Encryption Standard
CPA	Correlation Power Analysis
DPA	Differential Power Analysis
PCA	Principal Component Analysis
pdf	probability of density function
PP	Projection Pursuit
SCA	Side-channel Attack
SNR	Signal to Noise Ratio
TA	Template Attack
PSTA	Principal Subspace Template Attack

Glossary

A trace

a trace (or leakage trace) is a succession of N_s data measurements in a given time.

A sample

a sample designates a measurement at a specific time t for a set of traces.

A coordinate

a coordinate (or location) corresponds to a pair (x, y) referring to the width and height relative to the starting point of measurements (x_0, y_0) where the measurements were performed. Sometimes, this term can also refer to the set of traces taken at one coordinate.

Chapter 1

Introduction

1.1 Context

In this era of digital revolution, computer security is at the center of many issues. Passwords are used by most people in their cellphones, computers and bank accounts. Regarding their efficiency, the more passwords are securing important data, the more they should be hard to retrieve for adversaries. Cryptography, the ciphering of secret messages, began thousands of years ago with simple methods using pen and paper or little mechanical tools. With modern computers and electronic devices, cryptography is more than ever a field of great interest, using way more complicated schemes than before. Moreover, cryptanalysis, the art of breaking the ciphered codes, has always been growing as fast as cryptography.

More recently, attacks that do not deal with the cipher but rather with its implementation were designed. These are called side-channel attacks. As opposed to cryptanalysis which uses brute force or theoretical weaknesses in order to decipher the codes, side-channel attacks observe physical effects induced by the system performing cryptography. There are various types of side-channel attacks like observing the power consumption, observing the time of computation and the one of interest in this work: observing the electromagnetic radiations.

1.2 Objectives of this thesis

An easy way of observing the electromagnetic radiations of a chip performing encryption is to place a probe somewhere near the device and start measuring the EM fields

induced on the probe by the current in the chip. Placing the probe close to the supply voltage pin will lead to similar results as the analysis of power traces obtained by monitoring the difference in tension across a resistor with one end of the probe on the supply voltage pin. But what if there were some other locations leaking interesting information concerning the encryption uncorrelated with the one corresponding to the power consumption? What if it was possible to enhance existing attacks by combining the information obtained at several locations? These are the main questions we address across this work.

1.3 Organisation of this thesis

In Chapter 2, we introduce the theoretic concepts about cryptography, block ciphers, side-channel attacks and dimensionality reduction. Next, Chapter 3 presents the setup used for our experiments and the implementation targeted by our attacks. Chapter 4 consists in the analysis of a 8×8 XY-grid of measurements of electromagnetic radiations. We first investigate the level of information at each location and the correlation between the traces for each coordinate to finally perform different types of template attacks and analyse their efficiencies. Chapter 5 focuses on the comparison and the combination of power traces and XY-electromagnetic radiations. Finally, Chapter 6 shows how to use dimensionality reduction techniques to combine interesting informations from several locations.

Chapter 2

Background

2.1 Cryptography and block ciphers

Cryptography is the art of protecting information from anyone but the people who are allowed to have access to it (mostly people possessing the key to retrieve the information). Since the advent of computer, modern cryptography has essentially been based on mathematical theory, computer science and electrical engineering. From the protection of data against eavesdropper to the authentication of a user, many fields of information security depend on this science. In the wide range of cryptographic techniques, the ones of interest in this work are the encryption algorithms and in particular, block ciphers.

A block cipher is a deterministic symmetric-key algorithm operating on groups of bits of fixed length called blocks. It consists of an encryption function E and a decryption function D. Both algorithms take as input a block of size n bits and a key of size k bits and yield as output a block of size n bits. The encryption of a text P called plaintext with secret key K returns a text C called ciphertext, with $E_K(P) = C$. The decryption function performs the inverse operation which gives $D_K(C) = P$. The purpose of block ciphers is that, knowing the key K, it is easy to perform encryption or decryption but recovering K from a pair of function $E(P) = C$ and $D(C) = P$ is hard. We say that a block cipher should be a pseudorandom permutation indistinguishable from a random permutation against adaptive chosen plaintext attacks. Block ciphers are made of small easily invertible operations like the bitwise XOR or substitutions using tables but once many of those operations are executed sequentially, inverting them becomes hard. Figure 2.1 shows a simple example of block cipher taking one byte x_i as input going through two operations. The first one is a bitwise XOR between the input and one byte the key s, obtaining the intermediate value y_i. The second one is a substitution table called S-box (which will be explained later) taking as input y_i and outputing the final value z_i. Most

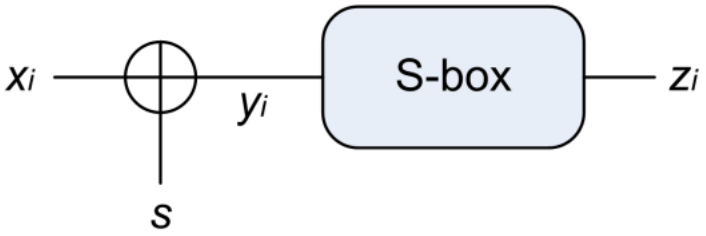

FIGURE 2.1: Simple Example of block cipher encryption. Image modified from: [1]

of the block ciphers work in rounds, meaning the encryption and decryption function are composed of mostly similar subfunctions applied to the plaintext successively. In this thesis, the target block cipher is the well-known Advanced Encryption Standard described in the following section.

2.1.1 Advanced encryption standard

The advanced encryption standard (AES) is a ciphering algorithm based on Rijndael cipher [2] which won in 2000 the AES contest launched by the National Institute of Standards and Technology in 1997 and was chosen in 2001 to replace the old Data Encryption Standard (DES [3]). The algorithm takes as input a 16 bytes (128 bits) State block and a 128, 192 or 256 bits key (below we only consider a 128 bits key). The ten rounds (plus an initial round) structure of the block cipher is represented in Figure 2.2. We can observe that the AES operates on a 4×4 bytes block called the State. At first, the State is filled column-wise with the 16 bytes of plaintext. The key is stored similarly in a 16 bytes block structure. The State then goes through the initial round consisting of a simple AddRounKey described in section 2.1.1.1. The next nine following rounds consist of four transformation described in section 2.1.1.1 to 2.1.1.4. Last round only applies three of the four transformations to wind up the algorithm. In the following sections, we explain the different transformations performed on the State during the AES.

2.1.1.1 AddRoundKey

The first transformation to be performed during the AES is the AddRoundKey. Since in this thesis we're only going to use one round of the AES, the AddRoundKey only consists in a bitwise XOR between the key and the State performed during the initial round in Figure 2.2. For a more complete and detailed explanation of this operation and the key schedule, see [2].

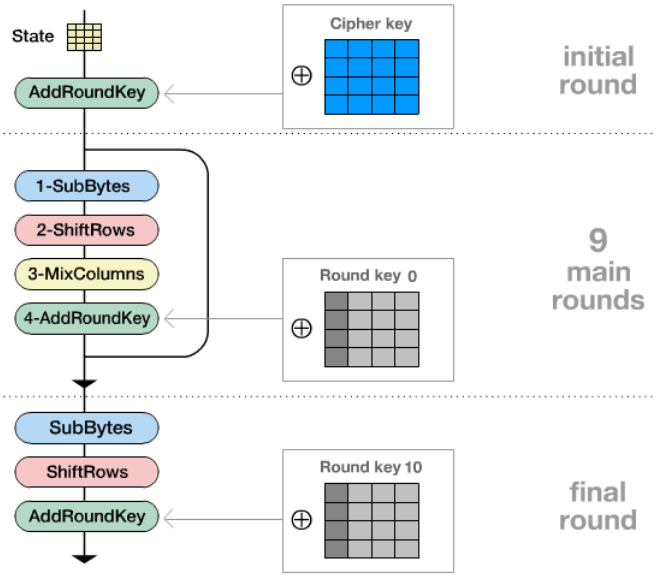

FIGURE 2.2: Development of the AES 10 rounds. Image modified from: [4]

2.1.1.2 SubBytes

The second and subsequent transformation of the AES is the SubBytes. SubBytes is a non-linear substitution of each byte of the State using a table (S-box) as shown on Figure 2.3. Appendix A presents the Rijndael S-box in hexadecimal form.

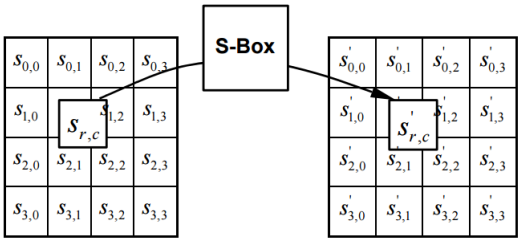

FIGURE 2.3: Modification of the State byte-by-byte during SubBytes.
Image Source: [2]

For example, if we have a State filled with $S_{x,y} = \texttt{ab}$, for each one we look in the table at column `a` and line `b` and we substitute the old value `ab` (*e.g.* `00`) with the new one from the hexadecimal form table $\texttt{a}'\texttt{b}'$ (*e.g.* `63`). After doing so for the 16 bytes, we obtain a new State filled with substituted values $S'_{x,y} = \texttt{a}'\texttt{b}'$ and the AES can process the next transformation on the State.

2.1.1.3 ShiftRows

The third transformation of the AES is called ShiftRows. The ShiftRows transformation operates, as its name suggests, on the rows of the State. It shifts the bytes of each row

by a certain offset. In our case, the offset is simply increased by one as the rows go down, starting from 0 in the first row. Figure 2.4 shows an example of ShiftRows transformation.

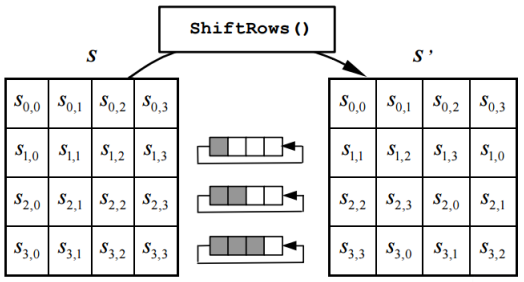

FIGURE 2.4: Modification of the State row-by-row during ShiftRows. Image modified from: [2]

2.1.1.4 MixColumns

The MixColumns transformation takes the four bytes of a column as input and outputs four bytes, where each input byte affects all four output bytes by a matrix multiplication. This type of transformation is also called a diffusion process. To be more precise, MixColumns performs the following matrix multiplication over each column c of the State as shown in Figure 2.5:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \tag{2.1}$$

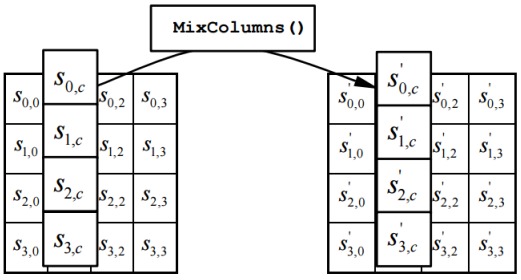

FIGURE 2.5: Modification of the State column-by-column during MixColumns. Image Source: [2]

Now that the encryption algorithm we are targeting has been explained we are going to explain in the subsequent sections the theoretic concept necessary to understand the attack we will perform.

2.2 Side-channel attacks

In the classical cryptanalysis setup against block ciphers, the adversary tries to recover the key based on eavesdropped messages, typically pairs of plaintexts and the corresponding ciphertext. In this case, the cryptographic algorithm is considered a black box: the plaintexts and ciphertexts are the only source of information for the adversary.

However, if the user has physical access to an embedded device, the black box assumption does not hold anymore. It then becomes possible for the user to monitor the physical behaviour of the device while it performs encryption. Attacks using such monitoring to find information on the secret key are called side-channel attacks (SCA) and have been a threat to embedded devices since the late 90's[5]. Figure 2.6 shows the three main steps of SCA and presents several interesting questions about each of them.

The first step in any SCA is the acquisition of measurements. Amongst the most used measured data are timing of computations[5], power consumption[7] and EM radiations [8]. Measurements are obtained by placing the device on a measurement board linked to an oscilloscope that monitors the desired criterion during the encryption process. We then obtain a signal at the output of the oscilloscope called leakage. The second step of SCA is the extraction of information from the physical leakage. The leakage can be pre-processed and dimensionality reduction can be used to reduce its size before the actual extraction of information[9]. The third and final step is the exploitation of the informations in order to retrieve the secret key used for encrypting. SCAs use divide-and-conquer strategies, meaning they will focus on finding part of the whole secret key rather than the whole key at once. Let's introduce now the main SCA used in this thesis.

2.2.1 Template attack

In this section, we define the concept of Template attacks (TA), the most powerful SCA from an information theoretic point of view [10]. As opposed to other simpler SCAs like Differential Power Analysis(DPA)[7], TA will extract as much information as possible

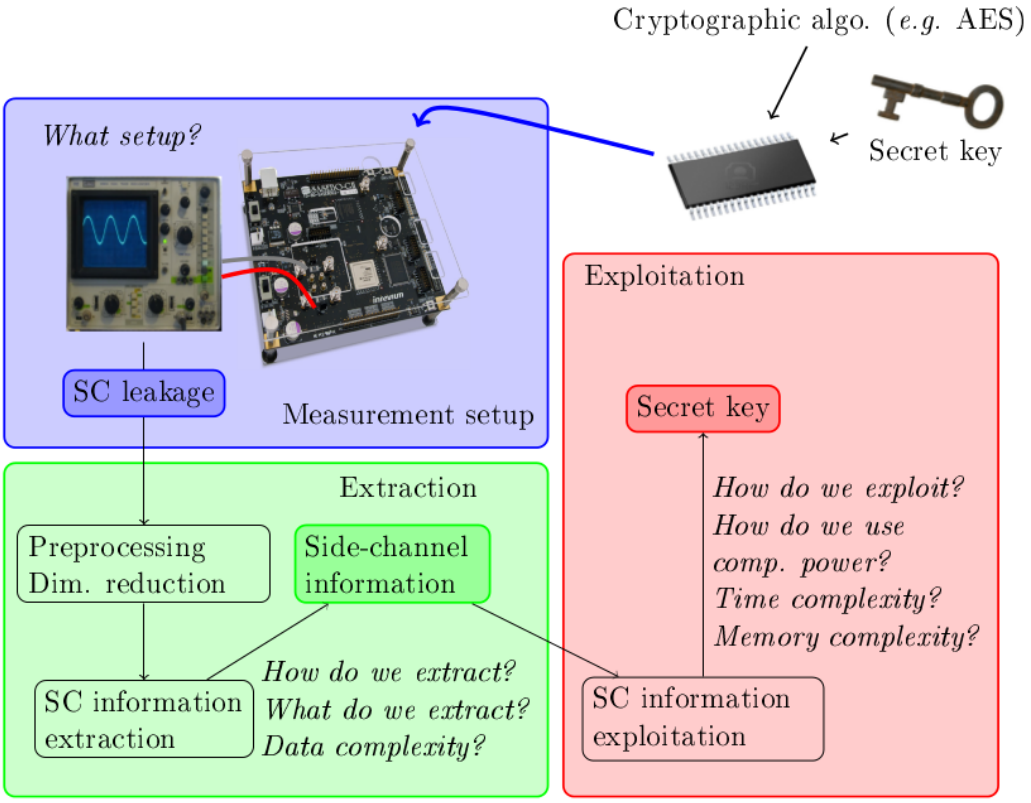

FIGURE 2.6: Side-channel attacks. Image source: [6]

from the leakage samples to build its probabilistic noise model $\mathcal{M}$ in order to match the device leakage function $\mathcal{L}$.

2.2.1.1 Templates construction

In TAs, we completely characterize the device leakage function $\mathcal{L}$ during the profiling phase. To this end, we compute an estimation of the probability density function(pdf) for each processed value $x \in X$ called template. TAs make the assumption that the leakage L while a specific x is processed can be expressed as

$$L = \mathcal{L}(x) = f(x) + N \tag{2.2}$$

with $f(x)$ a deterministic part and a randomly distributed noise N. Following the central limit theorem, we can say that L follows Gaussian distribution. The multivariate

Gaussian distribution with d the dimension of L [1] is the following:

$$L \sim \mathcal{N}(t|\mu_x, \Sigma_x) = \frac{1}{(2\pi)^{d/2}\,|\Sigma_x|^{1/2}}\; e^{-\frac{1}{2}(t-\mu_x)^\top \Sigma_x^{-1}(t-\mu_x)} \tag{2.4}$$

with μ_x the mean and Σ_x the covariance matrix. The noise distribution corresponding to the processing of the value $x \in X$ is completely characterized by the set of parameters (μ_x, Σ_x). Moreover, we'll have to estimate this set of parameters to build our template. A standard approach is to select the template corresponding to the time sample with parameters maximizing the likelihood of the traces under the noise model. Since the logarithm is a strictly increasing function over the range of the likelihood, the values which maximize the likelihood will also maximize its logarithm. Hence, we obtain the time sample $\hat{t}$ to build our templates:

$$\hat{t} = \underset{t \in N_s}{\arg\max} \sum_{x \in X} \log \mathcal{N}(t|\mu_x, \Sigma_x) \tag{2.5}$$

Other possible ways of selecting the time samples of interest used in this work are introduced in the further Section 2.2.3.

2.2.1.2 Attack

During the attack phase, we use the templates to compute, for each measurement, the probability of observing the leakage l assuming the processed value is x. By Bayes's theorem we obtain

$$Pr\left[L = l|X = x\right] = \frac{\mathcal{N}(l|\mu_x, \Sigma_x)}{\sum_{x' \in X} \mathcal{N}(l|\mu_{x'}, \Sigma_{x'})} \tag{2.6}$$

Knowing the plaintext $p \in P$, we can estimate the encryption subkey $s \in S$ by translating the probabilities from X to S. The function that enables us to find the most probable subkey guess $\hat{s}$ is:

$$\hat{s} = \underset{s' \in S}{\arg\max}\, Pr\left[S = s'|l\right] = \underset{s' \in S}{\arg\max}\, Pr\left[l|S = s'\right] Pr\left[S = s'\right] \tag{2.7}$$

The next section contains a brief explanation of the different types of countermeasures to SCAs.

[1] In case $d = 1$, we have a univariate Gaussian distribution defined as follows:

$$\mathcal{N}(t|\mu_x, \sigma_x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} e^{-\frac{(t-\mu_x)^2}{2\sigma_x^2}} \tag{2.3}$$

characterized by the set of parameters (μ_x, σ_x) with μ_x the mean and σ_x the variance

2.2.2 Countermeasures

The known countermeasures to SCAs are divided in two main categories:

- The first type of countermeasures focuses on the elimination or reduction of the information in the leakage. Typical examples are the adding of noise to jam the signal or the adding of a time delay to counter timing attacks. A physical protection is also considered as a countermeasure of this type.
- The second type of countermeasures is about eliminating or reducing the correlation between the leakage and the secret information. A typical example for this category is called blinding [11]. In summary, blinding computes a function for a client in an encoded form without knowing either the real input or the real output. Hence, the resulting leakage loses its correlation with the secret information. Another renowned countermeasure of this category is called masking [12]. In a nutshell, masking consists in adding a mask to the plaintext and key prior to the execution of the encryption scheme and then execute it with the masked data. Those masks can be boolean (*i.e.* a bitwise XOR) or arithmetic (*i.e.* a byte substraction). New masks are randomly chosen for each run of the algorithm which makes power consumption analysis inadequate to discriminate between key candidates.

In this work, none of the countermeasures stated above will be used and we will simply focus on an unprotected implementation of the AES. Next section explains how to select interesting time samples to build our templates.

2.2.3 Point of interests search

Building a model in order to perform a TA requires the selection of d interesting time samples with d the chosen dimension of the model. In this work, we only consider the univariate ($d = 1$) and bivariate ($d = 2$) case. In the following sections we are going to go over the different methods used in this work to select the time samples in our various TAs.

2.2.3.1 Correlation power analysis

In profiled Correlation Power Analysis(CPA)[13], we estimate the first order moments for each corresponding y intermediate value in a set of N_s traces. We denote l_y^s the traces corresponding to the intermediate value y at time s and $\hat{\mu}_y^s = \hat{E}(l_y^s)$ as their first

order moment, with $\hat{E}$ the sample mean operator. There are 256 intermediate y values in the AES Rijndael (see Section 2.2), meaning we compute $256 \times N_s$ means (one for each time sample for each y) giving us $\hat{\mu}s_y$. Then we compute the correlation between the means and the samples coming from a set of test traces l_y^s, $\hat{\rho}(\hat{\mu}_y^s, l_y^s)$, with $\hat{\rho}$ the Pearson's correlation.

In non-profiled CPA, we simply compute the correlation between a model we estimate close to reality (*e.g.* the Hamming weight of the output of a simplified encryption scheme) and a set of test traces. CPA can also be used as an evaluation metric in the univariate case[1].

2.2.3.2 Signal to noise ratio

Another discriminant for first-order moments is the Signal to Noise Ratio (SNR)[14]. SNR compares the level of a desired signal compared to the background noise. It's defined as follows:

$$\hat{SNR} = \frac{\hat{var}_y(\hat{E}_y(l_y^s))}{\hat{E}_y(\hat{var}_y(l_y^s))} \tag{2.8}$$

with $\hat{E}$ the sample mean operator and $\hat{var}$ the sample variance operator. SNR can also be used as an evalutation metric in the univariate case[1].

Once we have selected the time samples of interest, we can directly build our model and proceed to the attack. Next section tell us more about how to assess the performances of such an attack.

2.2.4 Evaluation metrics

2.2.4.1 Success rate

The general definition for success rate is the proportion of success among a number of attempts. In cryptanalysis, a success rate relates to the probability that the correct key is found by the adversary. Since in this work, we're only going to attack one byte of key s, the success rate r for N trials will be defined as follows:

$$r = \frac{\sum_{i=1}^{N}(\hat{s}_i == s)}{N} \tag{2.9}$$

With $\hat{s}_i$ the estimate key at trial i computed by exploiting both the black box and physical information in the SCA.

2.2.4.2 Perceived information

The perceived information (PI) can be used to evaluate the leakage of a cryptographic device. We introduce $H[S]$ the entropy of the key $S \in \mathcal{S}$ defined as the measure of its uncertainty during an experiment. The formula for the entropy is the following:

$$H[X] = -\sum_{x \in \mathcal{X}} Pr[X = x] \log_2(Pr[X = x]) \tag{2.10}$$

We now introduce the sample definition for PI:

$$\hat{PI}(S; X, L) = H[S] + \sum_{s \in \mathcal{S}} Pr[s] \sum_{x \in \mathcal{X}} Pr[x] \sum_{l_y^i \in \mathcal{L}_y^t} Pr_{chip}\left[l_y^i | s, x\right] \log_2(\hat{Pr}_{model}\left[s | x, l_y^i\right]) \tag{2.11}$$

With $S \in \mathcal{S}$ the key, $X \in \mathcal{X}$ the plaintext and $L \in \mathcal{L}$ the set of traces. As opposed to CPA and SNR, PI can be extended to multivariate attacks. It can be interpreted as the amount of information leakage that will be exploited by an adversary using an estimated model. The PI is a good predictor for the success rate of an actual TA exploiting the best $\hat{Pr}_{model}$ obtained through the profiling of a target device [15].

2.2.4.3 K-fold cross-validation

Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. The goal of cross-validation is to define a dataset to "test" the model in the training phase (*i.e.*, the validation dataset), in order to limit problems like overfitting and give an insight on how the model will generalize to an independent dataset (*i.e.*, an unknown dataset, for instance from a real problem). In k-fold cross-validation, the original sample is randomly partitioned into

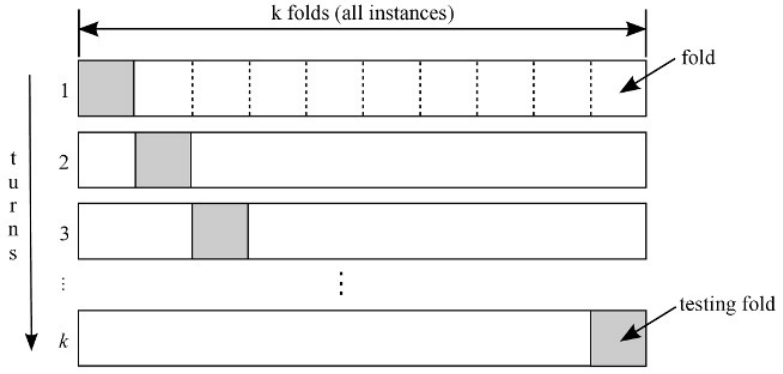

FIGURE 2.7: Diagram of k-fold cross-validation. Image Source: [16]

k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k - 1$ subsamples are used as

training data as shown on Figure 2.7. The cross-validation process is then repeated k times, with each of the subsamples used exactly once as the validation data. The k results from the folds can then be averaged to produce a single estimation. K-fold cross-validation can be used both in the univariate and multivariate case.

Next section, introduces us to the concept of dimensionality reduction, a process really useful when we are manipulating huge data which are not all interesting like in the case of our multiple traces.

2.3 Dimensionality reduction

Dimensionality reduction is the process of reducing the number of random variables. It can be divided into two domains: feature selection and feature extraction. Feature selection consists in, as its name suggests, selecting interesting features according to a criterion and getting rid of the rest of the data. This approach won't be used in our work. We focus on the second type of process: feature extraction. Feature extraction is about transforming data in the high-dimensional space to a space of fewer dimensions.

When we work in a high-dimensional space, for an ordinary TA, we look for interesting time samples from an information point of view. If these points are too close to each other or too correlated then building templates using all of them lacks of interests. This is why, in the multivariate case, we usually only take weakly correlated points separated by at least one clock cycle in order to avoid information redundancy. Another approach explained in this section consists in performing a projection of the leakage traces on a principal subspace.

A principal subspace can be viewed as a lower dimensional subspace in the data space where each axis successively indicates the direction in which the data maximize a criteria of interest (*e.g.* variability or SNR). After this step, the subspace contains interesting projected traces (features) that we can use to build our templates without having to worry about the fact that they are too close to each other. However, the correlation between features has to be taken into account when we perform TA. This approach is called Principal Subspace based Template Attack (PSTA)[9].

In the next section we present two statistical tools for finding the principal subspace of a data set: principal component analysis (PCA)[17] and projection pursuit (PP)[18].

2.3.1 Principal component analysis

PCA[17] is a standard statistical procedure in dimensionality reduction. It uses an orthogonal transformation that projects high-dimensional data into a low-dimensional subspace while conserving the variance of the data. The axis of the projection are called principal components. Figure 2.8 illustrates an example of PCA with a projection of the starting axis (t_1, t_2) onto the two orthogonal directions maximizing the variance of the data (v_1, v_2).

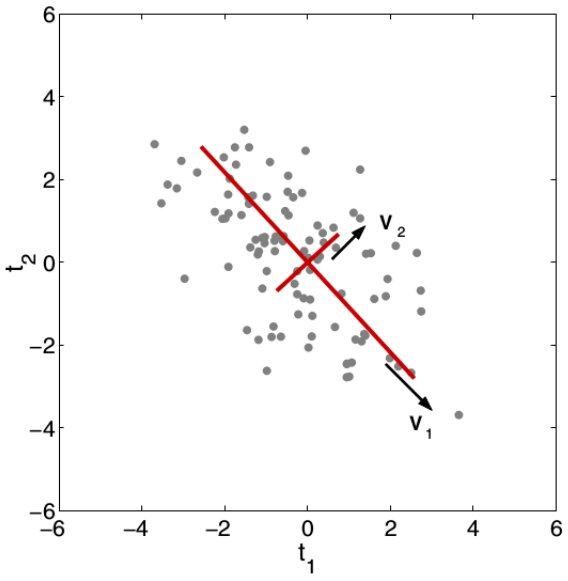

FIGURE 2.8: Example of projection with v_1 (resp. v_2) the first (resp. second) principal component

There are two main steps in PCA. The first one is about centering and standardizing the data, then finding a rotation to an orthogonal coordinate system such that the features are uncorrelated and in decreasing order of variance. The second one is simply keeping the C most important principal components in order to reduce the dimensionality. The variability in the principal components that are not kept is then considered as noise. Hence, if part of it was useful information, it's lost.

2.3.2 Projection pursuit

The main goal of Projection Pursuit (PP) is to select interesting low-dimensional projections of a high-dimensional data space by maximizing a chosen objective function. It essentially works by looking for the improvements of the projection when modifying it with small random perturbations.

PP is about finding a weight vector α that will project the N_s samples of a leakage vector l_y^i to a single sample λ_y^i in the following way:

$$\lambda_y^i = \sum_{t=0}^{N_s-1} \alpha(t) l_y^i(t) \tag{2.12}$$

This must be done in a way that maximizes the efficiency of univariate TA exploiting the λ_y^i's. This essentially requires to define an objective function that measures the level of information of these samples. In this thesis we choose the SNR (See Section 2.2.3.2) as objective function for our PP. This choice was done arbitrarily since all the evaluation metrics for SCA are equivalent [1].

Chapter 3

Practical Context

In this chapter, we describe the equipment used to perform our measurements and the way it was configured or assembled. We then explain the way we swept a square shaped area on the chip in order to retrieve measurements at multiple coordinates. Next, we go over how we modified the AES Rijndael to be able to efficiently record our sets of traces. Finally, we show a trace and explain the dimensions of the data we work with.

3.1 Measurements Setup

Our target device for this thesis is an 8-bit Atmel AVR (ATMega644P) microcontroller working at a 20 MHz clock frequency. In order to get accurate EM measurements, the chip was slightly depackaged [1]. We monitored the EM leakages with a small hand-made loop probe shown on Figure 3.1. It was built with a 2 mm diameter and 1.2 cm long toothpick around which we coiled two loops of 1.5 mm diameter copper wire soldered to a BNF connector. The signal was then amplified with a R& SHZ16 pre-amplifier and sampled with a Lecroy HRO66ZI oscilloscope sampling at 500 MHz and providing 12-bit samples[2]. In order to move the probe on a XY-coordinates grid, we used two motors controlled by an ESP300 Motion Controller. The probe was then fixed to an arm magnetized to the ESP300 and put as close as possible avoid any friction with the Atmel. The probe was held perpendicular to the device and its orientation wasn't a parameter we evaluated in this work. The step between each measurement was 0.5 mm in both X and Y directions, forming a square shaped area. Figure 3.2 shows the 8×8 coordinates

[1] We didn't manage to see the silicon die because every time we did, the chip was malfunctioning as a result. However, the depackaging was still necessary as the distance to the device influences greatly the quality of the measurements.

[2] In order to gain great transfer time, the traces were concatenated using the 'Sequence' mode of the oscilloscope.

FIGURE 3.1: The hand-made probe used for our experiments

(the intersections of red lines) at which we took measurements on the AtMega644P. The total area is thus $49 \times 0.25 = 12.25\,\text{mm}^2$. The whole setup can be seen on Figure 3.3.

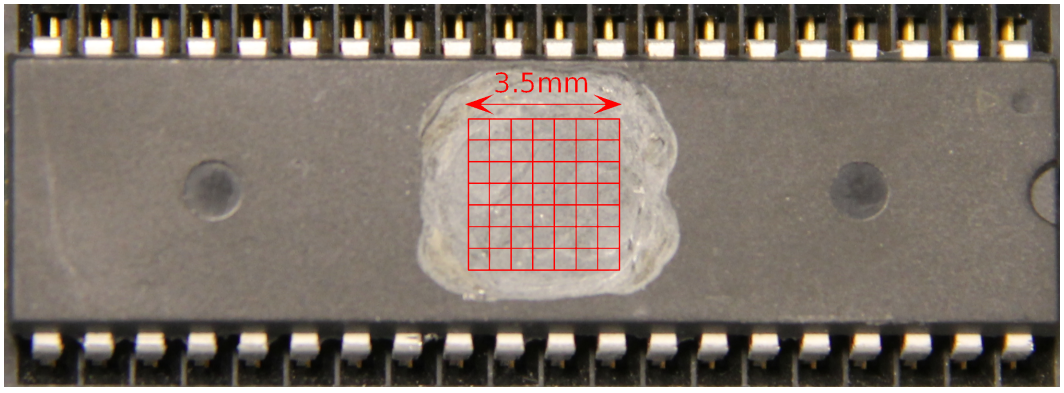

FIGURE 3.2: Area of measurements on the AtMega644P

3.2 Target Implementation

In order to minimize the time taken for one encryption (sending of plaintext and reception of the ciphertext included), we restricted the AES Rijndael block cipher (see Section 2.2) to only one the first round of encryption (the initial round followed by the first three transformations) for one byte of plaintext. In order to perform Rijndael block cipher on

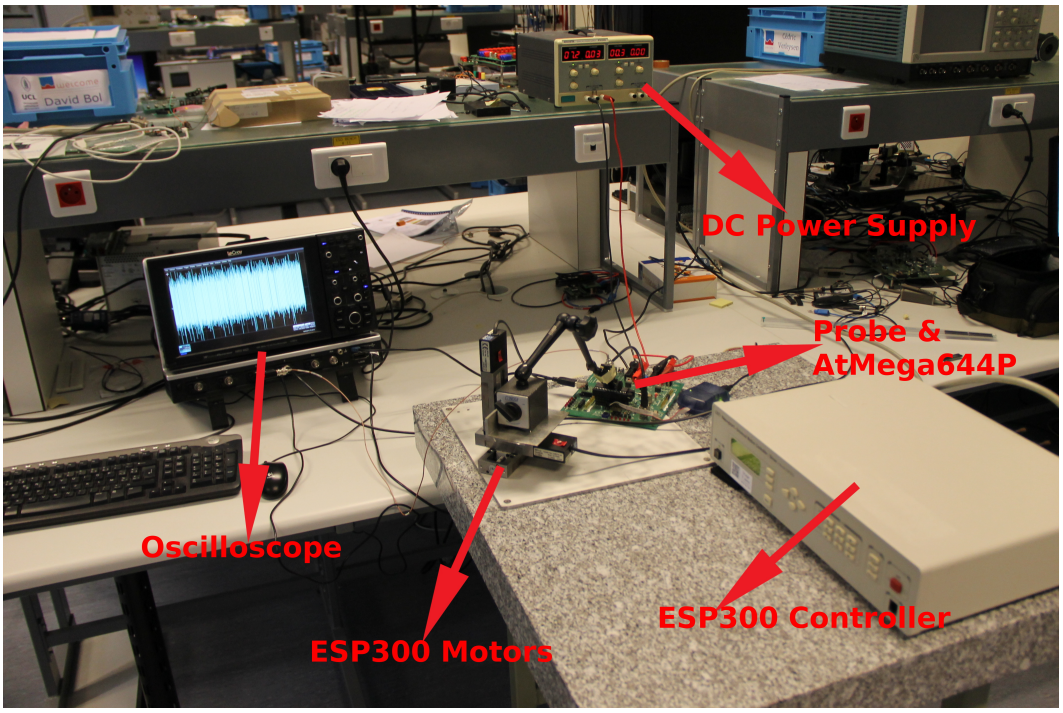

FIGURE 3.3: Setup for the experiments of this thesis

one byte we need four bytes of State[3] and four bytes of key (since we do a bitwise XOR between the key and the State). At first, we send the key only once and then for each encryption we send four bytes of plaintext (the State) and we retrieve one byte of cipher corresponding to the first byte of plaintext encrypted. Figure 3.4 represents the four bytes of States along our reduced AES and the cipher we retrieve in red. Next section describes the number of experiments and the data we work with.

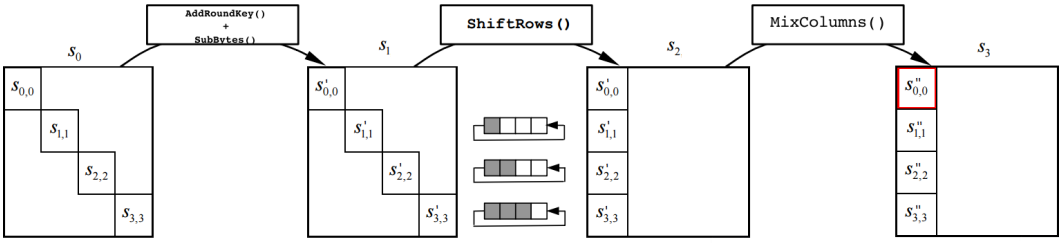

FIGURE 3.4: Evolution of the State across our reduced AES

3.3 Data Specifications

One coordinate by one, we perform a total of $N_e = 100$ experiments for each of the 256 y intermediate values. One execution of our encryption scheme takes 54 clock

[3]stored in the diagonal of it in order to form a column after the ShiftRows to perform the matrix multiplication in MixColumns

cycles, multiplied by the number of measurements per clock cycle we chose $N_c = 25$, we obtain $N_s = 54 \times N_c = 1350$ which we rounded to 2500 measurements in time (*i.e.* we recorded for sometimes after our encryption is over) because it was the closest number available in the oscilloscope. To sum up, a set of traces for one coordinate is of size $N_e \times |y| \times N_s$ or $100 \times 256 \times 2500$. Hence, the total number of data points is $64 \times 100 \times 256 \times 2500 = 4.096 \times 10^9$. Figure 3.5 shows one trace for both one full round

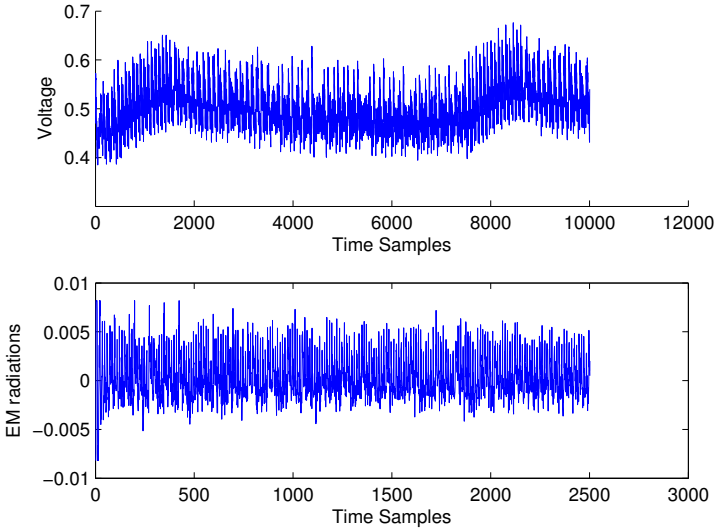

FIGURE 3.5: (Above) Example of trace for one round of full AES with power measurements (Below) Example of trace for one round of AES for one byte with EM measurements

of AES with power measurements and one round of reduced AES for one byte with EM measurements. The higher number of time samples for the power measurements are simply due to the higher number of clock cycles of one round of AES with a State of 16 bytes.

Chapter 4

Analysis of EM radiations over time and space

In this chapter, we explain our approach to build an efficient bivariate TA between two sets of EM traces taken at two different coordinates. In order to get a better global understanding, we first compute the SNR for the traces at each coordinate. Then, we evaluate the linear correlation between the traces from different coordinates. The second part of this chapter is centered around the different TAs we perform. We start with a simple univariate TA on the coordinate possessing the most informative sample according to a chosen criteria: the SNR. We follow with two different types of bivariate TAs: the first one is performed on two different samples from the traces of one coordinate and the second one is performed on the same time sample for two different coordinates. For each of them, we describe our choices for the methods of selection for the parameters of interest and how we built our templates. Finally, we compare the efficiency of our bivariate and univariate TAs and discuss our results.

4.1 Intuitive overview of our measurements

4.1.1 Quantity of information over space

To give a first overview of our measurements, we chose to present the maximum level of information at each coordinate whatever the time sample it comes from. It means that, for each coordinate, we compute the SNR and select the maximum value as represented on Figure 4.1.

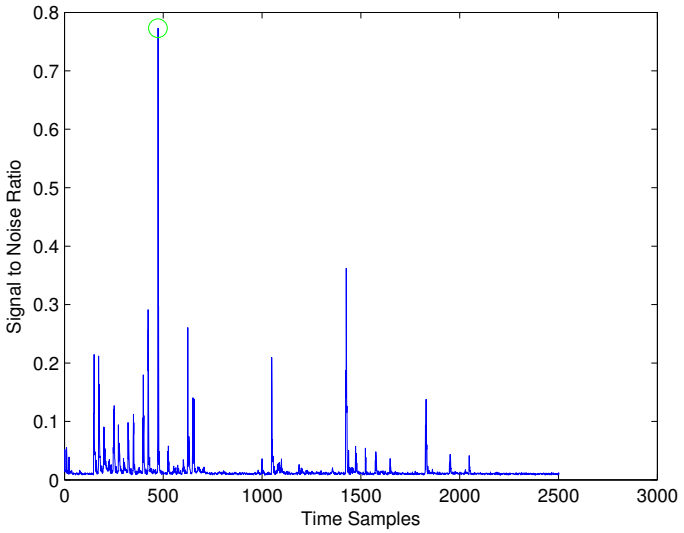

FIGURE 4.1: SNR for coordinate $(3, 2)$ and its maximum

Figure 4.2 shows the maximum SNR at each location if we neglect the time dimension. We can observe the growing shape culminating at a clear peak situated on the $(6, 6)$ coordinate for a SNR of value $\simeq 4.5$. The more distant in space the coordinate is from that peak, the smaller is its maximum SNR. That observation seems worrying since we

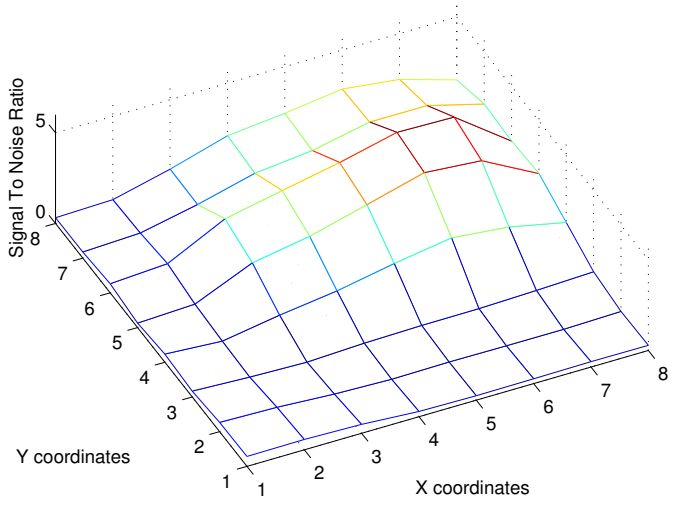

FIGURE 4.2: Maximum SNR for each location regardless of the time dimension

expect to find interesting information at several locations to upgrade classical SCA using only one coordinate. Fortunately, we'll see in the rest of this chapter and the following ones that lower levels of information don't mean that traces from those coordinates don't contain useful and/or complementary informations.

4.1.2 Correlation between traces of different coordinates

Working with an unprotected AES Rijndael cipher block means most of the information is located in the first moment: the mean μ [19]. Hence, in order to correlate the information between the EM radiations measured at two different locations we're simply going to take the linear correlation between their means. Figure 4.3 shows as example the correlations between each location and the location with the largest SNR at the time sample where it reaches its maximum. Most of the locations are strongly correlated with the most informative one. This result is partly due to the smallness of our step (0.5 mm) and preciseness of our handmade probe but it still gives us the intuition we wanted. Next, we are going to explain how we built our different TAs.

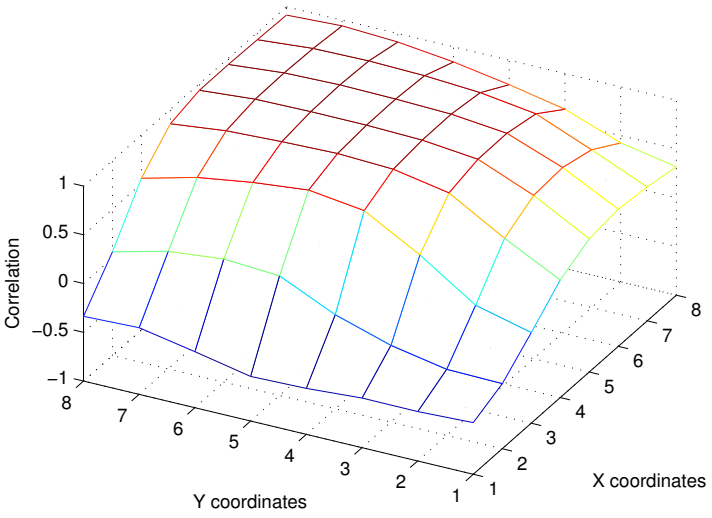

FIGURE 4.3: Correlation between each coordinate and the one with the largest SNR

4.2 Univariate template attack

We first proceed to a simple univariate TA. In order to build our model, we have to select a coordinate and a time sample. The coordinate with the time sample possessing the largest SNR described in the previous section is the best candidate. Figure 4.4 shows the SNR of the $(6, 6)$ coordinate and its maximum at time $t = 1050$ which we select to build our model on. Building our model is done by computing the 256 means μ_k and variances σ_k corresponding to the intermediate values y_k. Results of this attack are shown at the end of the chapter.

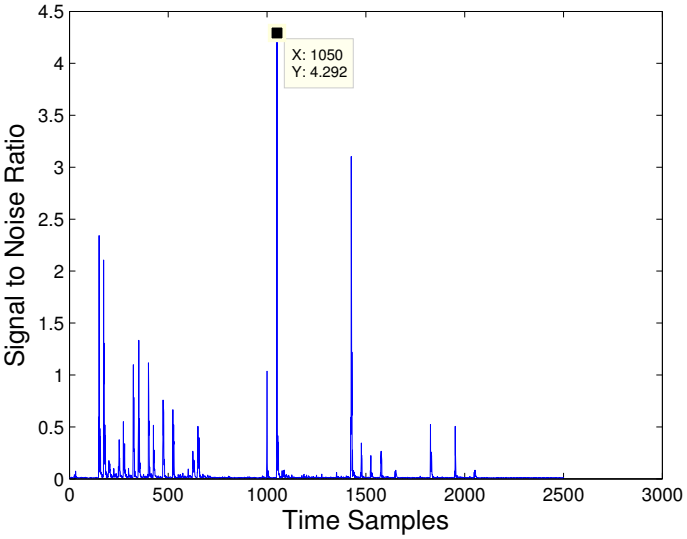

FIGURE 4.4: Signal to noise ratio for coordinate $(6, 6)$

4.3 Bivariate template attack in time

In this section, we are going to explain how we selected two time samples to build our bivariate model for TA. We had to select a set of traces coming from one location. We arbitrarily chose the same coordinate as in the previous section in order to avoid the time complexity induced by brute force search. We made this assumption because our model would be amongst the best ones possible considering that the selected coordinate possesses the sample with the highest level of information.

The following step was about selecting the two samples to build our model. Again, the brute way would mean selecting every possible pairs of time samples which would be too costly in terms of time. To avoid this huge time complexity we used our knowledge about the encryption scheme and about SCAs to get good time samples candidates. We used CPA to find the best time samples candidate regarding the correlation with the Hamming weight of $Y = \mathtt{P} \oplus \mathtt{K}$ and $Z = \mathtt{S}(\mathtt{P} \oplus \mathtt{K})$ respectively. Figure 4.5 shows the two correlations between our traces and the Hamming weight of Y and Z.

Once we obtained the two time samples, we were able to build our model and proceed to the attack. Results concerning this bivariate TA are shown at the end of this chapter.

4.4 Bivariate template attack in space

In this section, we are going to explain how we selected two (x, y) coordinates amongst the $\frac{n(n-1)}{2} = \frac{64 \times 63}{2} = 2016$ possible pairs such that they would both, at one given time,

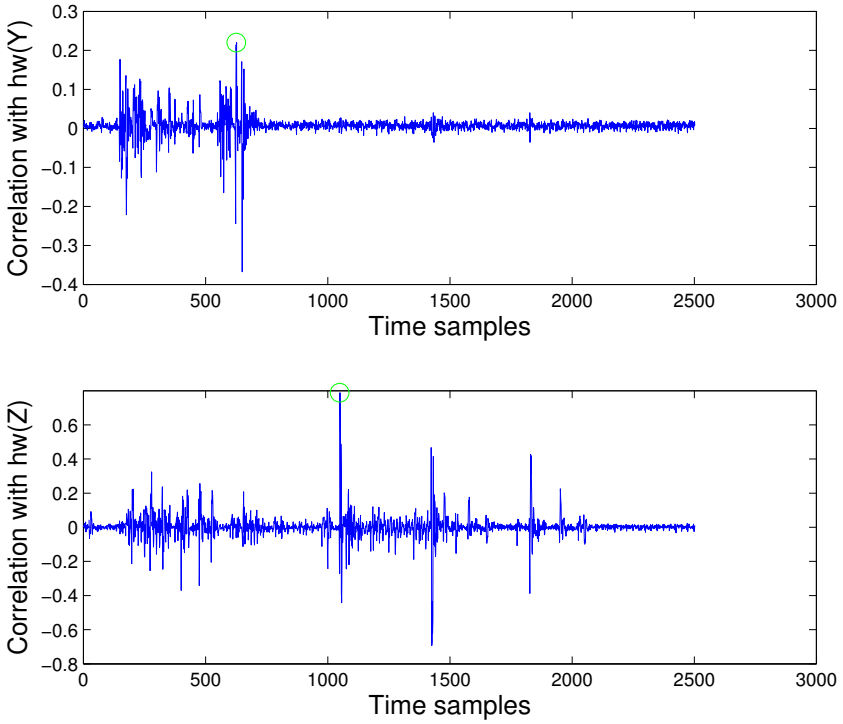

FIGURE 4.5: Correlation between our traces and the Hamming weight of Y and Z

possess interesting and somewhat independent information. In order to find this pair, we proceeded in the following way:

- We didn't want to evaluate every N_s time samples for each pair of points because the cost in terms of time would have been too high. Remained two options: choosing the most interesting time sample using a specific metric or using dimensionality reduction to project our traces and get rid of the time dimension. We chose the latter and performed PP (see Section 2.3.2) for each of the XY-EM radiations. The resulting projected traces were of size $256 \times N_e$, with N_e the number of experiments.

- Then, for each 2016 possible pairs of projected traces we performed 10-fold cross-validation to compute the PI (see Section 2.2.4.2) between the key and their bivariate template. For each cross-validation iteration we separated our traces L^y in two sets: a profiling set L^y_p containing 90% of L^y used to build the template and an evaluation set L^y_e containing 10% of L^y used to evaluate the probabilities in each template in order to compute the PI.

- Finally, for each pair we evaluate the mean over the ten PIs obtained during the cross-validation and we find our best pair: $\{(1, 2); (3, 3)\}$.

The following section is going to present and explain the result of the three attacks explained above.

4.5 Results

In this final section of our fourth chapter we are going to analyse the success rates of the three SCA we elaborated previously. Figure 4.6 shows the three success rates (defined in Equation (2.9)) corresponding to each TA at once in order to be able to compare them better. We notice that, while our univariate TA provides good results, we would expect for the bivariate TAs to outperform it which is hardly the case. In order to quantify the gain in information between the univariate and bivariate models we computed the PIs and obtained the following result: 10.99 for the univariate model, 12.07 for the bivariate model in time and 12.62 for the bivariate model in space.

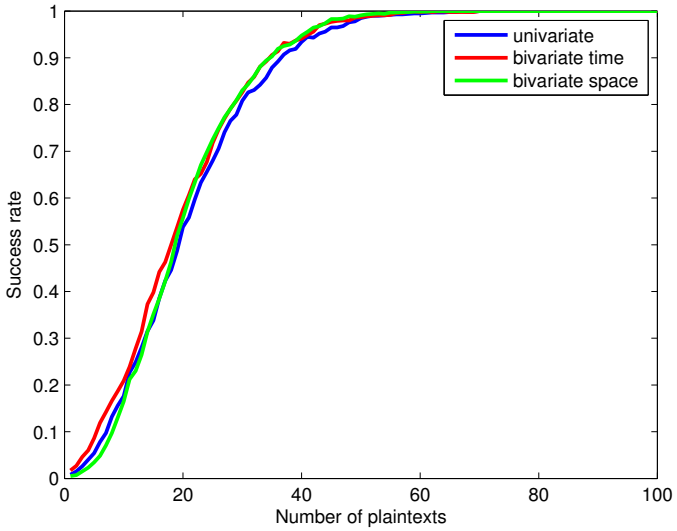

FIGURE 4.6: Success rate for the three TAs explained in this chapter

Looking closely at the SNR in Figure 4.4 and the correlations in Figure 4.5, we can draw conclusions concerning the Hamming weight hypothesis for the bivariate TA in time. On one hand, we see that the correlation with Z is significant (> 0.7) on the selected time sample corresponding to the most informative point $t = 1050$ mentioned earlier. On the other hand, the correlation with Y is really low and the time sample with the highest correlation has a SNR value of $\simeq 0.1$. This small level of information would explain the very little gain in efficiency of the attack. Another approach we tried consists in maximizing the PI in order to search for another interesting pair of time samples. The resulting pair was $t_1 = 1050$ and $t_2 = 150$ for which the perceived information was $PI = 13.47$. Unfortunately, the efficiency of the resulting attacks was poor. This can

be explained by the fact that t_2 is the time sample most correlated with the plaintexts which gives no information on the key whatsoever. To sum up, Figure 4.4 has peaks strongly correlated with the plaintexts and Z but not with Y which may be the cause of the stagnation of the results since the univariate TA already exploit the information at $t = 1050$.

Concerning the bivariate TA in space, We see here that PP doesn't work as well as in the future Section 6.1. Seeing the results we proceeded to another method to see if we would find something more effective. We selected, for each coordinate, the most informative time sample. Then we evaluated the PI for the bivariate model at that time sample for the coordinate and every other ones. Doing this for the 4032 possibilities, we just had to select the pair of coordinates and time sample for which the PI would have been maximum. Unfortunately, this method yielded similar results as the univariate TA as the information at two different coordinates in one time sample concerns the same operation and cannot give a fundamentally different information from one another.

The aim of this chapter was to get interesting results out of our huge data using hypothesis and heuristics to counter the time and space complexities caused by the size of the data. We encountered relative success as our multivariate TAs failed to improve significantly the quality of our univariate TA. The limitations of our approach are that the heuristics or hypothesis must be consistent in order to gain significant information and that we absolutely wanted to avoid brute force. Maybe performing a brute force selection once would have guided us towards better hypothesis and heuristics as we would have known the best possible answers already.

Chapter 5

XY-Electromagnetic radiations and power analysis

This chapter is built around different interactions between classic power analysis and our XY-EM analysis. Firstly, we compare the efficiency of both the univariate and bivariate TAs for both types of measurements. Secondly, we evaluate (for certain interesting time samples) the correlation between power and EM in order to try an attack with a model built from power (resp. EM) traces on an attack set of EM (resp. power) traces. Finally, we combine power and EM information in a bivariate TA and compare the efficiency of this new attack with the previous ones from this work.

5.1 Comparison between EM and Power efficiency

In this section, we describe how we built our models for univariate and bivariate TAs with power measurements. Then, we compare their performance against the corresponding attacks whose models were built with EM measurements from Chapter 4.

5.1.1 Univariate case

To obtain our univariate templates (256 mean μ_k and variance σ_k as explained in Section 2.2.1), we have to select a time sample maximizing a chosen criterion: the SNR.

Figure 5.1 shows the SNR of our power traces and its maximum circled in green at time $t = 1728$. After selection of the time sample of interest we can build our model and proceed to the attack. Figure 5.2 represents the comparison between the attack using

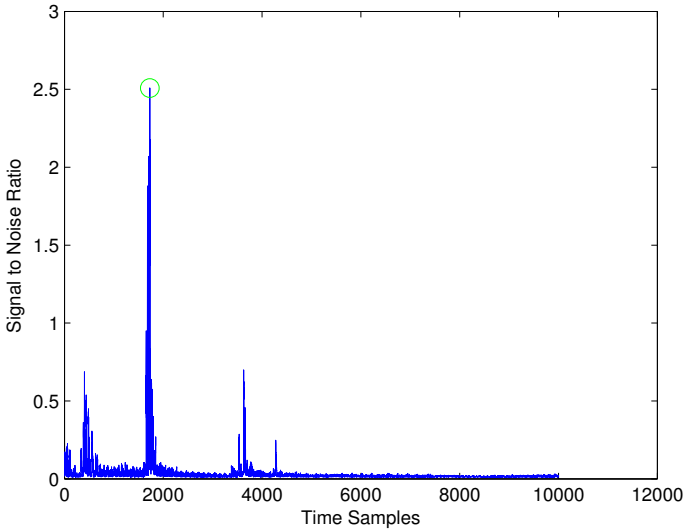

FIGURE 5.1: Success rate for the univariate template attack for power and EM measurements

the model we just built and the univariate TA built in section 4.2. We can observe that the attacks of the EM TA are more efficient than the ones from the power TA. These results were predictable since the level of SNR for the selected time sample in EM (resp. power) in Figure 4.4 (resp. 5.1) is $\simeq 4.5$ (resp. $\simeq 2.7$) and every metric for evaluating univariate SCAs are equivalent [1].

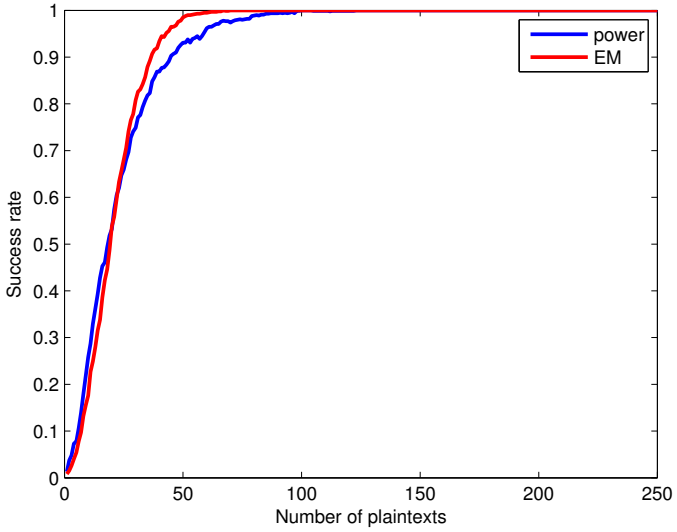

FIGURE 5.2: Success rate for the univariate template attack for power and EM measurements

5.1.2 Multivariate case

In this section, we describe how we selected the two time samples to build our bivariate model. As for the EM bivariate TA in time from Section 4.3, we used CPA to find the best time samples candidate regarding the correlation with the Hamming weight of $Y = \mathtt{P} \oplus \mathtt{K}$ and $Z = \mathtt{S}(\mathtt{P} \oplus \mathtt{K})$ respectively. Figure 5.3 shows the two correlations between our traces and the Hamming weight of Y and Z and the two time samples we selected circled in green. Once we selected our two time samples, we can build our

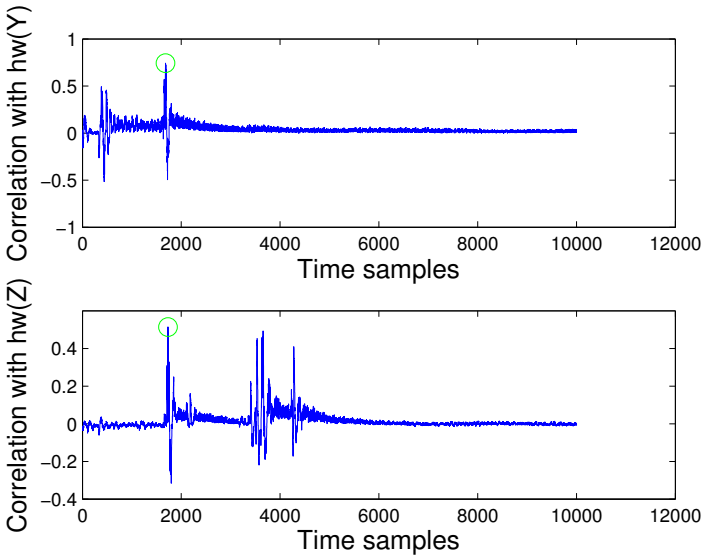

FIGURE 5.3: Correlation between our power traces and the Hamming weight of Y and Z

templates and proceed to the attack. Figure 5.4 shows three success rate curves for the TA built in this section and the two bivariate TA from the previous chapter.

Though previous section showed that univariate TA was more efficient for EM, we can observe very clearly that the bivariate TA from power is more efficient then the two bivariate TAs from EM. It means that the addition of information in the power case was significantly higher then for our two EM approaches. Indeed, the increase in PI goes from 9.11 for univariate to 14.09 for bivariate which is a significantly greater increase than for EM. We see that in this case, the Hamming weight hypothesis gives great results by selecting two adequate time samples.

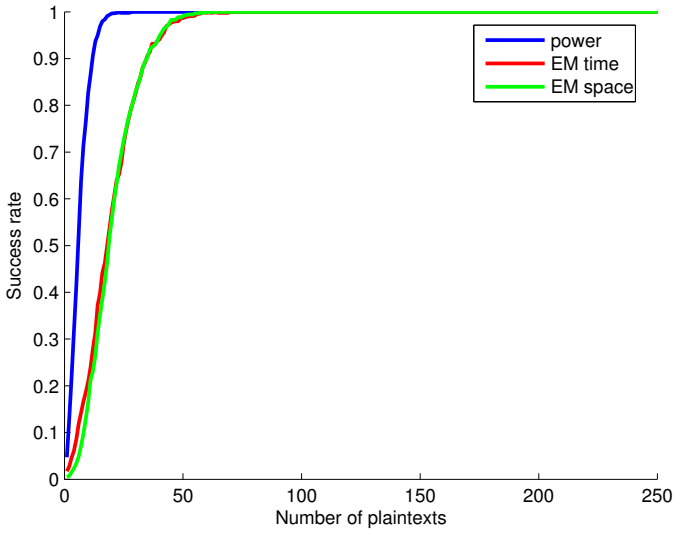

FIGURE 5.4: Success rate for the bivariate template attacks for power and EM measurements

Chapter 6

Extraction of information using dimensionality reduction

In this chapter, we use dimensionality reduction techniques to perform TAs containing informations from multiple coordinates. To do so, we're going to handle bigger data obtained by concatenation of traces coming from different locations. The selection of the whole 8×8 coordinates being impossible because of the time and space complexity, we restrain ourself to 8 locations selected to cover a lot of space and not be too close to each other. Those coordinates are represented in green on Figure 6.1.

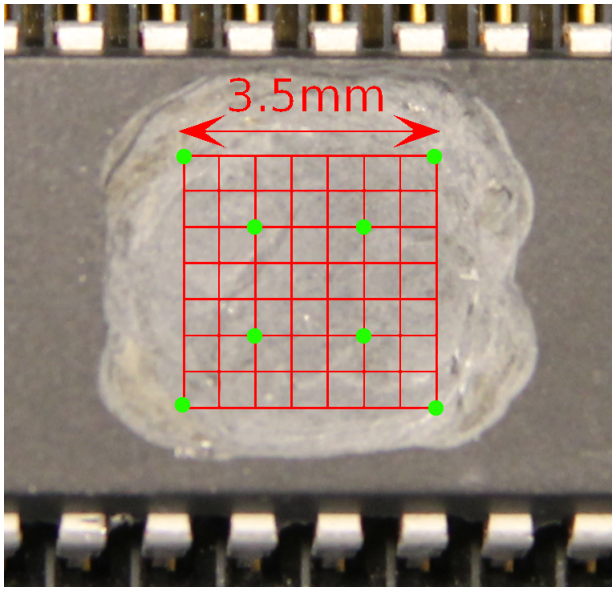

FIGURE 6.1: The eight locations chosen to perform dimensionality reduction

Figure 6.2 gives us a look at the new traces we work on after concatenation, we see that the number of time samples is now $N_s \simeq 2 \times 10^4$. Next section explains how we built our templates on reduced data using PP (see Section 2.3.2).

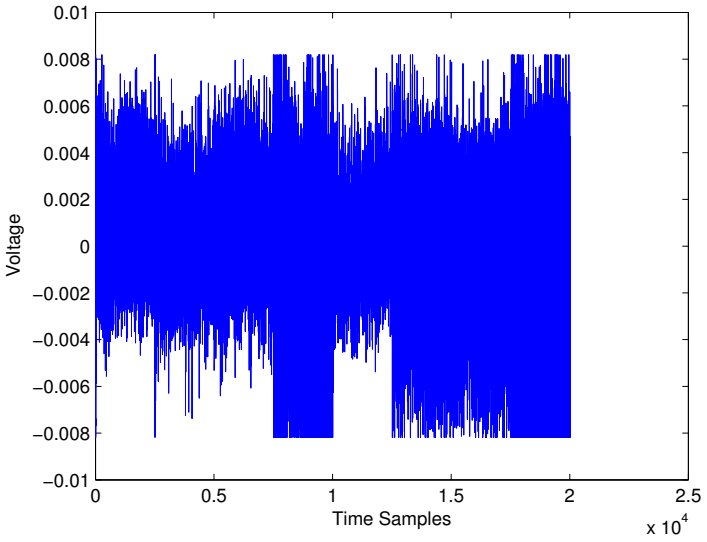

FIGURE 6.2: Concatenation of traces coming from eight different coordinates

6.1 Projection Pursuit

6.1.1 Dimensionality reduction and model building

In order to reduce the data to a lower dimensional space (*i.e.* a one dimensional space), we used PP to determine a weight vector $\alpha(t)$ maximizing the SNR over a certain number of iterations. We first initialize $\alpha(t) = \alpha_0$. Then, at each of the 4×10^4 iterations (two times the total number of time samples), we selected a random time sample t_r and computed the value of $\alpha(t_r)$ maximizing de SNR of the projected data $\lambda_y^i = \sum_{t=0}^{N_s-1} \alpha(t) l_y^i(t)$. Figure 6.3 shows the evolution of the maximum SNR through our PP algorithm, with a final value of approximatively 72. Figure 6.4 shows the final weight vector after 4×10^4 iterations. The more a weight at a time sample is big the more our algorithm estimated this time sample to give information. Once we obtained the λ_y^i, obtaining our templates (μ_k, σ_k) was trivial and we could proceed to the attack. Next section is going to introduce the results of the model built in this section.

6.1.2 Results

Figure 6.5 shows us the success rates for the bivariate TA with power traces, the bivariate TA in space with EM traces and finally, the TA using dimensionality reduction explained in the previous section. We see a clear improvement between the bivariate TA in space and the new TA, meaning that, as we suspected, we were able to find new interesting information in the concatenated traces. However, we see that the bivariate TA with

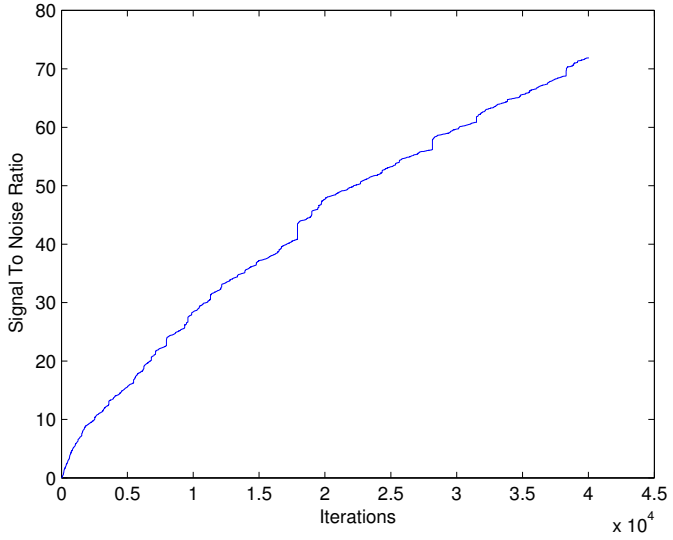

FIGURE 6.3: Evolution of the maximum SNR during our PP algorithm

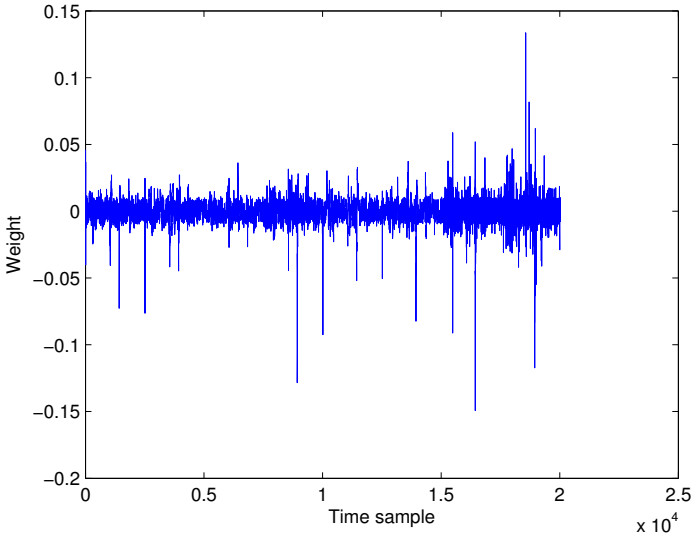

FIGURE 6.4: Final weight vector $\alpha(t)$ of PP

power traces is still more efficient. The possible reasons for this are explained in the latter Section 7.2.

Another observation concerns the efficiency of our new TA compared to its metrics. A model with SNR $\simeq 72$ should yield better results than these shown in Figure 6.5. This contradiction is mainly due to overfitting. Maximizing the SNR of a set of profiling traces without measuring the corresponding evolution of the SNR of a set of test traces (using the same weight vector) led to an overestimation of the information.

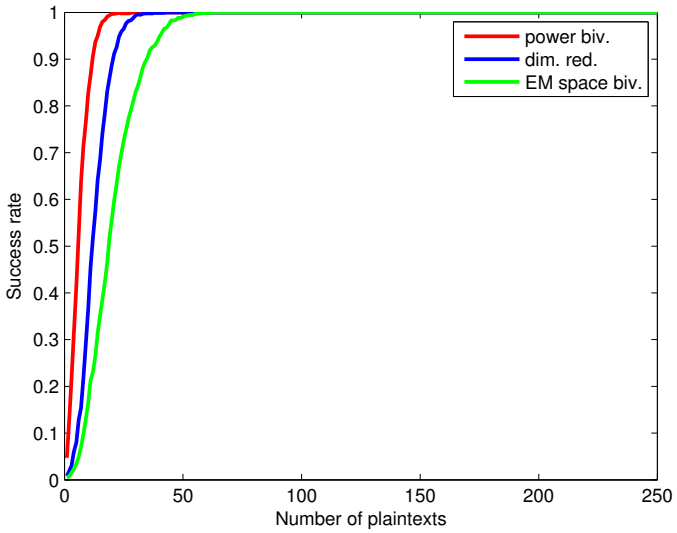

FIGURE 6.5: Results for the best attacks of the two previous chapters and the TA of this section (in blue)

6.2 Using Principal Component Analysis

6.2.1 Dimensionality reduction and model building

As explained in Section 2.3.1, PCA looks for the C first principal directions w_c forming an orthonormal basis of the C-dimensional subspace maximizing the variance of the data. In this section, the data on which we perform the PCA is the empirical mean of our concatenated traces l_i^y: $\hat{\mu}_y = \hat{E}(l_i^y)$ with $\hat{E}$ the sample mean operator.

It can be shown that the principal components(PC) are the eigenvectors of the empirical covariance matrix [17]:

$$\overline{S} = \frac{1}{|y|} \sum_{y=1}^{|y|} (\hat{\mu}_y - \overline{\mu})(\hat{\mu}_y - \overline{\mu})^T \tag{6.1}$$

with $|y|= 256$ and $\overline{\mu} = \frac{1}{|y|} \sum_{y=1}^{|y|} \hat{\mu}_y$ the average of the empirical mean traces. Computing the matrix $\overline{S}$ for a great number of dimensions (in our case $\mathcal{O}(n^4)$) his very expensive. However, it's possible to compute it more efficiently using a technique called small sample size PCA [9].

In order to find our first PCs, we are going to look for the C vectors corresponding to the C largest eigenvalues of the empirical covariance matrix. Doing so, we obtain our orthogonal basis for the C-dimensional subspace $W \in \mathbb{R}^{N_s \times C}$ containing our C PCs. Building our PSTA is then simply about estimating the projected means ν_y and the covariance matrices of the projected traces along the first selected principal directions

Λ_y given by:

$$\nu_y = W^T \hat{\mu}_y \text{ and } \Lambda_y = W^T \hat{\Sigma}_y W \tag{6.2}$$

with $\hat{\Sigma}_y$ the covariance matrix of the traces corresponding to the intermediate value y. Now that the templates are built, if we want to attack a new trace l_{new}, we just have to project it in our new subspace $l_{proj} = W^T l_{new}$ and then the process is the same as for previous TAs.

6.2.2 Results

Before looking at the actual efficiency of the TAs of dimension $C = 1, 2, ..$ (corresponding to the number of selected PCs), let's analyze the principal subspace resulting of our PCA using the eigenvalues characterizing its axis. In Figure 6.6, we show the curve of the 256 eigenvalues sorted in decreasing order. We can observe that around the twentieth eigenvalue, every subsequent eigenvalue is very small compared to the largest one $\simeq 1.5 \times 10^{-4}$. Figure 6.7 zooms on the 20 first eigenvalues in order to have a better idea of the shape of the graph. We can see that the variance is conserved in the 17 first eigenvalues after which the eigenvalues are equal to zero. Figure 6.8 shows the percentage of the total variance contained in each dimension. We can see that the first dimension of the new subspace only conserved 20% of the variance of the original space. We can expect that a univariate TA using only the first PC to project in its one dimensional subspace won't get good results. Figure 6.9 gives the cumulative percentage of variance for a C-dimensional subspace, with C the number of principal directions selected. It seems to be a better clue as to how many PCs we need to select to conserve most of the variance but we'll see that the facts unfortunately don't match the theory.

Concerning the efficiency of the multivariate PSTAs of dimension $C = 1, 2, 3$ or 4, Figure 6.10 shows us the success rates for the attacks performed in the C-dimensional subspace using the corresponding four PCs for projection. We see that as the number of dimensions rises, the efficiency of the PSTA follows. Looking back at Figure 6.9, we see that selecting four PCs means we conserve approximatively 60% of the original variance which would intuitively mean that we could still have better results by performing higher dimensional PSTA. Unfortunately, selecting further PCs to project our original high-dimensional space to a lower one results in less efficient results. Figure 6.11 shows us the success rates for the PSTAs with four and eight dimensions respectively. We observe that the highest dimensional subspace gives poorer results than the other whereas it conserves more of the variance $\simeq 80\%$ as shown in Figure 6.9. This type of issue called the curse of dimensionality [20] means that as we increase the dimension in the data

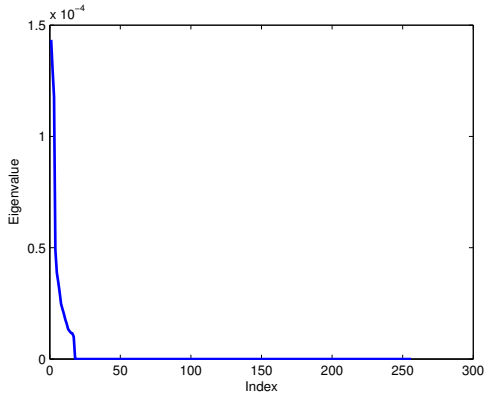

FIGURE 6.6: Eigenvalues

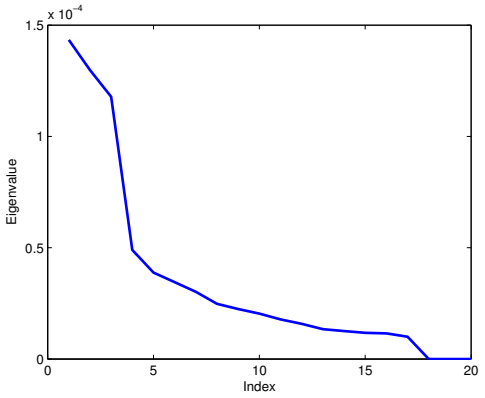

FIGURE 6.7: Largest eigenvalues

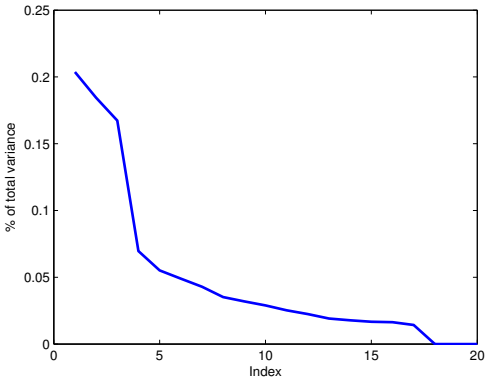

FIGURE 6.8: Percentage

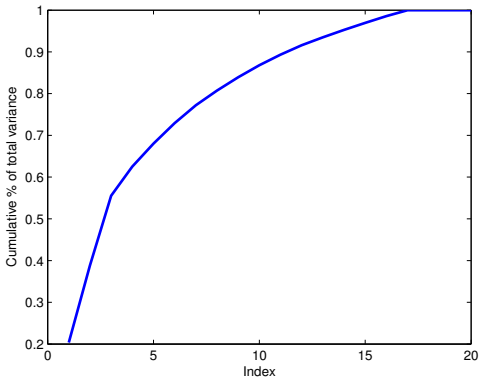

FIGURE 6.9: Cumulative percentage

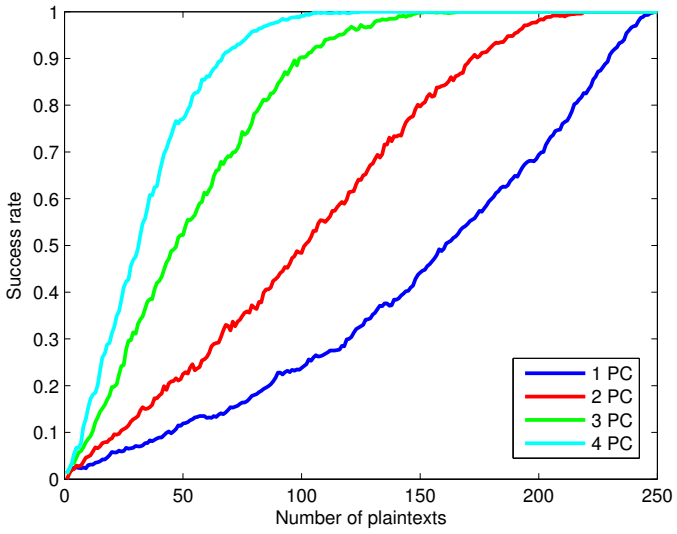

FIGURE 6.10: Success rates for our PSTA with $C = 1, 2, 3$ & 4 PCs

space, the space seems to become emptier and interpolation becomes harder. However, the issue will not be addressed in this work and we will stick to the actual results.

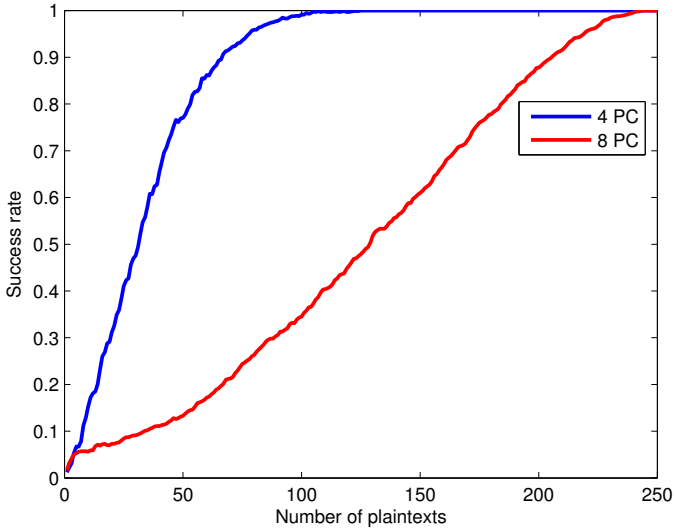

FIGURE 6.11: Curse of dimensionality for higher level PCAs

Chapter 7

Conclusion

At the end of this work, we have a better global understanding on the possibilities offered by a grid of electromagnetic measurements when it comes to side-channel cryptanalysis. We have taken knowledge of new techniques making use of this new data material: the main ideas, their practical limitations, the hypothesis made to counter the limitations and their efficiencies relative to one another.

Looking over our work as a whole, the main idea was to exploit the XY coordinates and see if we could extract more information from them than in a unique set of traces, may it be power or EM traces. I think that point has been well made with the success of the PP in Section 6.1 which could be refined to be more efficient.

Overall, we can summarise our approach as follows:

- We were faced with a first purely technical challenge of retrieving the traces representing the EM radiations for the encryption of plaintexts at various coordinates over a chip. Faced with timing constraints, we decided to reduce the encryption scheme (see AES Rijndael in Section 2.2 and reduction in Section 3.2). Still because of the timing, to which we can add the constraint of the probe's precision, we decided to take measurements at 64 different locations on a grid canvassing the embedded device. At this point, we obtained the essential material of our work.
- Our second challenge was to think about how we were going to use our new data composed of trillions of points while imposing ourselves some time and space constraints. we began with simpler attacks and we spiced things up as we progressed. The first idea was to look for the best time sample ,all coordinates included, using POIs search (see Section 2.2.3). This first trial encountered success as it was both quick and didn't use a lot of space in addition to be an efficient attack. However,

one of the original aim of our work was to combine several dimensions of our data to try to extract information from them.

- The next challenge was then to find ways to select those dimensions while we still respected the constraints. We first thought of something similar to what we do most of the time for classical power analysis: a bivariate attack selecting two time samples using the Hamming weight hypothesis. This method respected our constraints but it didn't fare well with our previous results as explained in Section 4.5. In another attempt, we thought having space dimensions in the data is what made it special from other works and we tried to use it by selecting two coordinates to perform a bivariate attack. We had to use dimensionality reduction (see Section 2.3) to respect our constraints. Also, we used PI (see Section 2.2.4.2) as evaluation metric to determine our coordinates. Again, the results weren't conclusive.

- Nevertheless, we gave ourselves as last challenge to combine even more locations aiming to, in the ideal case without constraints, combine all of it together. Our idea to reach this goal was to concatenate traces from different coordinates and perform dimensionality reduction on them. Since dimensionality reduction can be costly in time and we were manipulating bigger traces, we had to restrain ourselves to a few locations. Although faced with typical issues for the reduction of dimensionality and regression (*i.e.* curse of dimensionality and overfitting), we managed to obtain good results using PP (see Section 2.3.2). Moreover, these results are improvable both in their computation time and efficiency as we discuss in Section 7.2.

Having summarized our work we can now assess its strengths and weaknesses:

Strengths

- The content of the data in itself, especially the space dimensions proved useful.
- The dimensionality reduction that could be combined with some other optimisation.
- The modularity allows heuristics and approximations to be swapped for others in our approach.
- The reproducibility of the work granted by our explanations.
- Basic ground for the subject is covered, opening the road for innovative and more thorough techniques.

Weaknesses

- The time to obtain the data necessary to perform our analysis.
- The size of the data.
- The struggle for avoiding huge time complexities.
- The inconsistency of known hypothesis and heuristics like the Hamming weight.
- Technical dependences (the probe and the depackaging).

The remainder of this chapter contains a summary of the contributions of this work and a discussion of the interesting possibilities for future works in the continuity of this one.

7.1 Contributions

In Chapter 3, we detailed our experiments setup, our reduced version of the Rijndael AES and the huge matrix of data we manipulate. The aim of this chapter was, on one hand, to explain the context for the experiments of our work. On the other hand, it was crucial for it to be precise enough so that any future work based on it could reproduce it or produce some other experiment based on it.

In Chapter 4, 5 and 6 we introduced and compared different TAs whose performances and computation times are summarised in Table 7.1 (resp. 7.2) for the attacks based on EM (resp. power) measurements. First column of the tables specifies the TA in question introduced in this work. Second column gives us the average classification rate. Third column is filled with, for each TA, the first number of plaintext N_{P} with perfect success rate and all subsequent success rates are also perfect. Last columns gives us an approximation of the time needed to compute the parameters and the templates. Timings of the attacks was not taken into account because it only varies between univariate and bivariate (*i.e.* two univariate (resp. bivariate) attacks will take the same amount of time).

Model	Average classification rate	Success rate = 1 at	Computation time
univariate TA	0.79	$N_{\mathrm{P}} = 70$	$\simeq$ 30sec
bivariate TA (time)	0.80	$N_{\mathrm{P}} = 66$	$\simeq$ 2min
bivariate TA (space)	0.81	$N_{\mathrm{P}} = 61$	$\simeq$ 1h
PSTA (PP)	0.87	$N_{\mathrm{P}} = 43$	$\simeq$ 10h
PSTA (PCA)	0.73	$N_{\mathrm{P}} = 125$	$\simeq$ 30min

TABLE 7.1: Recapitulative table of the different EM TAs from this work

Various conclusions can be taken from looking at those tables. First and foremost we see that, in Table 7.1, the average classification rate of our TAs $\in [0.73; 0.87]$ and that for a given time (*e.g.* 2min) and a chosen number of plaintext (*e.g.* 66), we can obtain with a 100% probability the encryption key from the reduced AES explained in Section 3.2. Still in Table 7.1, average classification rate of the TAs is positively correlated in an exponential way with their computation times[1]. It means we developed different kinds of TAs: the faster, less efficient ones and the slower, more efficient ones. Depending on the concrete cryptographic problem, one might have to choose between one or the other. The third and last conclusion is an objective but less positive one. When we compare the results from both tables, we see that while we have a better univariate TA in EM, we still didn't manage to gain as much information as with the power traces between the univariate and the bivariate case. It means that there is room for improvement in this way as we discuss amongst other topic in the next section.

Model	Average classification rate	Success rate = 1 at	Computation time
univariate TA	0.78	$N_{\mathtt{P}} = 100$	$\simeq$ 30sec
bivariate TA	0.94	$N_{\mathtt{P}} = 29$	$\simeq$ 2min

TABLE 7.2: Recapitulative table for the two power TAs from this work

7.2 Future works

In this section, we're going to browse through the preceding chapters of this work, starting from Chapter 3 to Chapter 6, with the aim of pointing out interesting directions for possible future works.

We start with Chapter 3 about the setup for our experiments and the basic concept of this work: the XY-grid of EM measurements. While in this work we introduced mainly the idea of XY-EM measurements and a few techniques to try to use this diversity of locations, the quality of our measurements was not as high as we could expect.

[1] Except for PCA who suffered from the curse of dimensionality causing its results to be poorer. Even with that the PSTA performed was of dimension 4 and didn't fare well with other bivariate TAs. We conclude that PCA wasn't appropriate in the context of big concatenated traces and we won't talk of this model further on.

For starters, with a more precise probe we would obtain better traces more distinct frome one another. Hence, it would have been interesting to shorten the length of the step between coordinates to obtain more interesting data. Of course, this would have meant manipulating larger data which was already an issue in this work. Hence, the preprocessing of data is also an interesting question. Secondly, The depackaging of the chip was not total (as mentioned in the concerned chapter we could not see the silicon die) causing the measurements to be less precise. Thirdly and lastly, performing the measurements on a full AES to be able to analyse the results and compare them with existing works would be of great interest. Again, this was an issue in terms of time and space complexity, a recurrent issue in this work.

Following is Chapter 4, in which we explain our first steps in the use of our traces. The main changes that could be brought to the experiences of this chapter is about the fact that on two occasions (for both bivariate TAs) we used hypothesis and approximations to perform our experiments. It would be interesting to compare how our choices fare with the bruteforce results and try some other techniques for example. In a second time it could also be interesting to go beyond bivariate with dimension $n = 3, 4, ...$ and try to combine different time and different coordinates in a multivariate TA. On an whole other topic slightly related to this chapter, it would be really interesting to compare the architecture of a depackaged AtMega644P and the evolution of the SNR at each coordinates and each clock cycle relating to the corresponding assembly instruction. Doing so would allow to connect the hardware to the software in a unique way.

In Chapter 5, where we compare our TAs between EM and power measurements, there are two essential ideas going out of this work. The first one concerns the generality of the models. More precisely, the idea is about analyzing the attack of power (resp. EM) test traces with a model built based on EM (resp. power) traces. The second idea goes in an opposite direction: instead of confronting or comparing the different types of measurements, the idea is to combine both. A multivariate TA using both EM and power could be very interesting.

Finally, Chapter 6 uses dimensionality reduction techniques to extract information from more dimensions, whether they're time or space dimensions. A straightforward improvement from our work would be to design an heuristic to determine, from $N_{coordinates}$ on which we want to perform dimensionality reduction, which coordinates we choose to improve the TAs. Since we only could afford a rather small number of coordinates in our concatenated trace (8 over 64), an interesting statistic would also be the comparison between the efficiency of the attacks for $N_{coordinates} = 1, 2, ..., 64$. Another point would be, as explained in the previous paragraph, the combination of EM and power traces this time in the concatenated traces.

Appendix A

Rijndael S-box

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

FIGURE A.1: Rijndael Substitution box for the byte **xy** in hexadecimal

Bibliography

[1] Stefan Mangard, Elisabeth Oswald, and F-X Standaert. One for all–all for one: unifying standard differential power analysis attacks. *IET Information Security*, 5 (2):100–110, 2011.

[2] Joan Daemen and Vincent Rijmen. Aes proposal: Rijndael. 1998.

[3] Data Encryption Standard. Data encryption standard. *Federal Information Processing Standards Publication*, 1999.

[4] Enrique Sabala. The Rijndael Inspector. `http://www.formaestudio.com/rijndaelinspector/archivos/Rijndael_Animation_v4_eng.swf`, 2008.

[5] Paul C Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology—CRYPTO'96*, pages 104–113. Springer, 1996.

[6] Mathieu Renauld. *Advanced Extraction and Exploitation of Side-Channel Information in Cryptographic Implementations*. PhD thesis, UCL, 2012.

[7] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology—CRYPTO'99*, pages 388–397. Springer, 1999.

[8] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic analysis: Concrete results. In *Cryptographic Hardware and Embedded Systems—CHES 2001*, pages 251–261. Springer, 2001.

[9] Cédric Archambeau, Eric Peeters, F-X Standaert, and J-J Quisquater. Template attacks in principal subspaces. In *Cryptographic Hardware and Embedded Systems-CHES 2006*, pages 1–14. Springer, 2006.

[10] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. Template attacks. In *Cryptographic Hardware and Embedded Systems-CHES 2002*, pages 13–28. Springer, 2003.

[11] Geoffrey Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. In *Computer Security Foundations Symposium (CSF), 2010 23rd IEEE*, pages 44–56. IEEE, 2010.

[12] Thomas S Messerges. Securing the aes finalists against power analysis attacks. In *Fast Software Encryption*, pages 150–164. Springer, 2001.

[13] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *Cryptographic Hardware and Embedded Systems-CHES 2004*, pages 16–29. Springer, 2004.

[14] Stefan Mangard. Hardware countermeasures against dpa–a statistical analysis of their effectiveness. In *Topics in Cryptology–CT-RSA 2004*, pages 222–235. Springer, 2004.

[15] François Durvaux, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. How to certify the leakage of a chip? In *Advances in Cryptology–EUROCRYPT 2014*, pages 459–476. Springer, 2014.

[16] Joshua Eckroth. Classification Evaluation. `http://cse3521.artifice.cc/classification-evaluation.html`, 2012.

[17] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[18] J.W. Tukey J.H. Friedman. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on computers*, 23(9):881–890, 1974.

[19] Santos Merino Del Pozo, François-Xavier Standaert, Dina Kamel, and Amir Moradi. Side-channel attacks from static power: when should we care? In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, pages 145–150. EDA Consortium, 2015.

[20] Michel Verleysen and Damien François. The curse of dimensionality in data mining and time series prediction. In *Computational Intelligence and Bioinspired Systems*, pages 758–770. Springer, 2005.