UNIVERSITY NAME

DOCTORAL THESIS

Thesis Title

Author:
John SMITH

Supervisor:
Dr. James SMITH

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

October 18, 2017

Contents

List of Figures

List of Tables

List of Abbreviations

SCA	**S**ide **C**hannel **A**ttack
AES	**A**dvanced **E**ncryption **S**tandard
NIST	**N**ational **I**nstitute of **S**tandards and **T**echnology

List of Symbols

$GF(2^b)$ Galois Field of order 2^b

Part I

Context and State of the Art

Chapter 1

Introduction

1.1 Introduction to Cryptography

The terms *Cryptography* (from the Greek *kryptòs* (secret) and *graphein* (writing)) and *Cryptanalysis*, denotes two branches of a science named *Cryptology*, or *science of the secret*. Cryptography initially refers to the art of *encrypting* messages, which means writing meaningful messages in such a way to appear nonsense to anyone unaware of the encryption process. In general, cryptography aims to construct protocols to secure communication, while cryptanalysis studies the resistance of cryptographic techniques, developing *attacks* to break the cryptosystems' security claims. These two complementary domains evolve in parallels, since the evolution of attack techniques allows conceiving more resistant cryptographic algorithms, and inversely the resistance of such algorithms requires the conception of more sophisticated attacks.

The art of cryptography is very ancient, probably as ancient as the language, but only the development of information technology made cryptology take the shape of a proper science, sometimes referred to as *Modern cryptology*. The last be seen as a branch of different disciplines, such as applied mathematics, computer science, electrical engineering, and communication science. Modern cryptosystems exploit algorithms based on mathematical tools and are implemented as computer programs, or electronic circuits. Their goal is to provide security functionality for communications that use *insecure channels*, for example the internet. In particular, modern cryptosystems are designed in order to ensure at least one of the four following information security properties:

a. *confidentiality*: the transmitted message must be readable only by a chosen pool of authorized entities;

b. *authenticity*: the receiver can verify the identity of the sender of a message;

c. *non-repudiation*: the sender of a message cannot deny having sent the message afterwards;

d. *data integrity*: the receiver can be convinced that the message has not been corrupted during the transmission.

Two branches of cryptography may be distinguished: the *symmetric cryptography* and the *asymmetric cryptography*. The first one historically appeared before and is based on the hypothesis that the two communicating entities share a common secret, or private key; for this reason this is also called *secret key cryptography*. The second one, introduced around 1970, allows any entity to encrypt a message in such a way that only a unique chosen other entity could decrypt it; this is also called *public key cryptography*.

A general principle in cryptography, nowadays widely accepted by cryptography researchers, is the one given by Kerckhoff in in 19th century: it states that cryptosystems should be secure even if everything about the system, except the key, is public knowledge. Following this principle, today many industrials and governmental agencies exploit for their security services cryptosystems based over standardized algorithms. Such algorithms are of public domain, thus have been tested and tried to be broken by a large amount of people, before, during and after the standardization process. Resistance to many attempts of attacks is actually the strengths of standard algorithms.

In the following part of this section a description of the two standard cryptographic primitives, *i.e.* building block algorithms used to build cryptographic protocols, that will be used in this thesis will; a symmetric one, the AES, and an asymmetric one, the RSA.

1.1.1 Description of AES

The *Advanced Encryption Standard* (AES) has been standardized in 2001 by the United States governmental agency *National Institute of Standards and Technology* (NIST) through the *Federal Information Processing Standards Publication 197* (FIPS PUB 197) [NIS]. It is a symmetric *block cipher*, *i.e.* an algorithm operating on fixed-length groups of bits.[1] The AES operates on blocks of 128 bits of plaintext, and can use keys of size 128, 192 or 256 bits. The encryption is done by rounds. The number of executed rounds depends on the key size (10 rounds for 128 bits, 12 for 192 and 14 pour 256). The basic unit for processing in the AES algorithm is a byte. For AES internal operations, bytes are arranged on a two-dimensional array of bytes called the *state*, denoted s. Such a state has 4 rows and 4 columns, thus contains 16 bytes. The byte lying at the i-th row, j-th column of s will be denoted by $s_{i,j}$ for $i, j \in \{0, 1, 2, 3\}$. The 16 input bytes and the 16 output bytes are indexed column-wise as shown in Fig. 1.1. Each element $s_{i,j}$ of a state is mathematically seen as an element of the *Rjindael finite field*, defined as $GF(2^8) = \mathbb{Z}/2\mathbb{Z}[X]/P(X)$ where $P(X) = X^8 + X^4 + X^3 + X + 1$. Five functions are performed during the AES, named KeySchedule, AddRoundKey, SubBytes, ShiftRow and MixColumn. At high level the AES algorithm is described hereafter:

Key Expansion: derivation of round keys from secret key through the KeySchedule function

Round 0: AddRoundKey

Rounds 1 to penultimate: SubBytes
ShiftRows
MixColumns
AddRoundKey

Last Round: SubBytes
ShiftRows
AddRoundKey

A description of the five functions is provided hereafter.

[1] in contrast with *stream ciphers*, which operate over a single plaintext bit at time

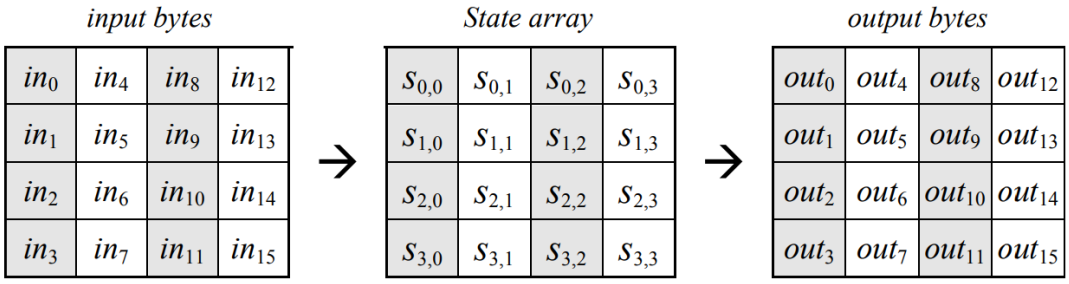


FIGURE 1.1: State array input and output. Source: [NIS].

KeySchedule

The key round of the initial round of AES coincides with the secret encryption key $\boldsymbol{K} = (k_{0,0}, k_{0,1}, \ldots, k_{0,3}, k_{1,0}, \ldots, k_{1,3}, \ldots, k_{3,3})$. The i-th round key is given by

$$\boldsymbol{K_i} = (k_{4i,0}, k_{4i,1}, \ldots, k_{4i,3}, k_{4i+1,0}, \ldots, k_{4i+1,3}, \ldots, k_{4i+3,3}),$$

where, for $i > 3$

$$\begin{cases} k_{a,b} = k_{a-4,b} \oplus k_{a-1,b} & \text{if } a \not\equiv 0 \bmod 4 \\ k_{a,b} = k_{a-4,b} \oplus \text{Sbox}(k_{a-1,(b+1)\bmod 4}) \oplus \text{Rcon}(a) & \text{if } a \equiv 0 \bmod 4, \end{cases}$$

where $Rcon(a) = 2^{a-1}$ in the Rjindael finite field.[2]

AddRoundKey

SubBytes

ShiftRows

MixColumns

1.1.2 Description of RSA

1.2 Secured Components

As we have seen in the previous section, modern cryptography proposes solutions to secure communications that ask for electronic computations and repose their security over some secret keys. Keys are represented as long bit strings, impossible to be memorized by users. Thus, keys need to be stored in a secure medium, and never delivered in clear over insecure channels. Smart cards were historically conceived as a practical solution to such a key storage issue: they consist in small devices a user can easily carry around with, which not only store secret keys, but also are able to internally perform cryptographic operations, in such a way that keys are never asked to be delivered. The registrations of a first patent by Roland Moreno in 1974 and of a second one by Michel Ugon in 1977 are often referred to date the smart card invention, finally produced for the first time in 1979. Smart cards are pocket-sized plastic-made cards equipped with a secured component, which is typically an integrated circuit containing a some computational units and some memories.

[2] where $2 = (00000010)_2$ is represented by the polynomial x

Today, about 40 years after its invention, they still have a huge diffusion, both in terms of applicative domains and in terms of quantity of exemplars. Indeed, they serve as credit or ATM cards, healthy cards, ID cards, public transport payment cards, fuel cards, identification and access badges, authorization cards for pay television, etc. Slightly changing the card support, we find other applications of the same kind of integrated circuits, for example the mobile phone SIMs (*Subscriber Identity Module* and the electronic passports. In terms of quantity, it seems that in 2014 8.8 billion smart cards have been sold, *i.e.* the same order of magnitude of the global population.

In addition to smart cards, the recent growing and variation of security needs lead to the development and specification of other kinds of secured components, for example the *Trusted Execution Environment* (TEE), which as a secured part of the main processor of a smartphone or tablet, and the *Trusted Platform Module* (TPM), which is a secure element providing cryptographic functionalities to a motherboard.

1.2.1 Embedded Cryptography Vulnerabilities

Side-Channel Attacks

Until the middle of nineties, the security of embedded cryptosystems was considered as equivalent to the mathematical security of the cryptographic algorithm. In classical cryptanalysis an attacker has usually the knowledge of the algorithm (in accordance to the Kerckhoff's principle) and of some inputs and/or outputs. Starting from this data, his goal is to retrieve the secret key. This attack model considers the algorithm computation as a black box, in the sense that no internal variable can be observed during execution, only inputs and/or outputs. With his seminal paper about Side-Channel Attacks (SCAs) in 1996, Paul Kocher showed that such a black-box model fails once the algorithm is implemented over a material component [Koc96]: an attacker can indeed inspect its component during the execution of the cryptographic algorithm, monitor some physical quantities, for example the execution time [Koc96] or the instantaneous power consumption [KJJ99] and deduct information about internal variables of the algorithm. Depending on the attacked algorithm, making inference over some well chosen internal variable (the so-called *sensitive variables* of the algorithm) is sufficient to retrieve the secret key. After these first works it was shown that other observable physical quantities contained *leakages* of sensitive information, for example the electromagnetic radiation emanating from the device [GMO01; QS01] and the acoustic emanations [GST14]. Moreover, if until few years ago it was thought that only small devices, equipped with slow microprocessors and in a small-size architecture, such as smart cards were vulnerable to this kind of Side-Channel Attacks, this recent work about acoustic emanations, together with other works exploiting electromagnetic fluctuations pointed out that much faster and bigger devices, *i.e.* laptops and desktop computers, are vulnerable as well [Gen+15; GPT15; Gen+16].

A Classification of the Attacks to Secured Components

The Side-Channel Attacks outlined in previous paragraph, and which are the main concern of this thesis, belong to a much bigger family of attacks that can be performed to break cryptographic devices security claims. First of all software attacks and hardware attacks must be distinguished. Software attacks only exploit vulnerability coming from the way the component's code is written. Hardware attacks

TABLE 1.1: Classification of Attacks

Hardware Attacks	Passive	Active	
			Invasive
			Semi-Invisive
	SCAs	FAs	Non-Invasive
Software Attacks			

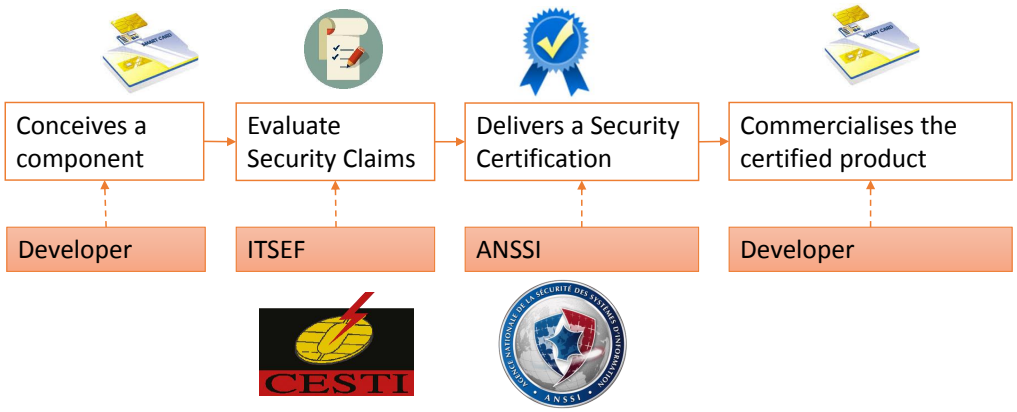


FIGURE 1.2: The actors of French Certification Scheme

are still commonly classified on the base of two criteria: on one hand we can distinguish passive and active attacks, on the other hand we can distinguish invasive, semi-invasive, non-invasive attacks.

Passive attacks: in passive attack, the device is let run respecting its specifications. The attacker observes its behaviour without provoking any alteration;

Active attacks: in active attacks a special manipulation is performed in order to make the normal behaviour of the device change.

Invasive attacks: in invasive attack, the device is depackaged and inspected at the level of the components technology. The circuit can be modified, broken, signals can be accessed via a probing station, etc. There is no limits to the manipulations attacker can do to the components;

Semi-invasive attacks: as in invasive attacks the device is depackaged, but in contrast to them, no direct electrical contact to the chip is done;

Non-invasive attacks: in non-invasive attacks the device is not modified and only accessible interfaces are exploited.

In literature, the term Side-Channel Attacks commonly denotes the passive non-invasive attacks. In the same way, active non-invasive attacks are often referred to as *Fault Injection Attacks*.

1.2.2 Certification of a Secure Hardware - The Common Criteria

In previous paragraphs we have evoked the great diffusion of the cryptographic devices, which implies consequent great risks in case of vulnerabilities of such largely diffused devices, and the existence of a wide range of attacks exploiting vulnerabilities coming from the way cryptography is embedded. These factors justify the importance and necessity to ensure reliability on the security claims of commercialised

secured components and thus the arise of several guidelines and standards for their evaluation. The international standard ISO/IEC 15408, also known as *Common Criteria for Information Technology Security Evaluation* (abbreviated as *Common Criteria* or simply *CC*) represents one of the stronger efforts in standardization, unifying in 1999 three previously existing standards:

- the *Trusted Computer System Evaluation Criteria* (TCSEC - United States - 1983)
- the *Information Technology Security Evaluation Criteria* (ITSEC - France,Germany, Netherlands, United Kingdom - 1990)
- the *Canadian Trusted Computer Product Evaluation Criteria* (CTCPEC - Canada - 1993).

The actors

The CC define four actors of the evaluation process of a secured component:

- **The Developer**, who conceives a product and wish sell it a certified secured product. He send a request for evaluation to the certification body and, once the request is accepted, he contacts an evaluation laboratory
- **The ITSEF** is the *IT Security Evaluation Facility*; in France it is named *Centre d'Evaluation de la Securité des Technologies de l'Information* (CESTI). It is an evaluation laboratory, in possession of a certification body agreement, which performs the security tests to assess the resilience of the product
- **The Certification Body** is often a governmental organism, the *Agence National de la Securité des Systèmes d'Information* (ANSSI) in France, or the *Bundesamt für Sicherheit in der Informationstechnik* (BSI) in Germany. It ensures the quality of the evaluation and delivers a certificate to the developer
- **The end user**, who buys the product and follows its security guidelines.

The Target of Evaluation and the security objectives

To start the certification process, the developer compiles a document called *Security Target* (ST). Such a document begins specifying the (part of the) device subjected to evaluation, the so-called *Target of Evaluation* (TOE) and then lists its *Security Functional Requirements* (SFR), choosing by those proposed by the CC. In practice, and to ease de redaction of the ST, the choice of the SFRs is not open, but guided by the typology of the component. In particular, the CC propose a catalogue of *Protection Profiles* (PP), associated with the required SFRs; for example *smart card* or *TEE* designate some precise PPs.

Evaluation Assurance Level and Security Assurance Requirements

In CC seven *Evaluation Assurance Level* (EAL) are defined, and determine the quantity and complexity of the tasks the evaluator has to effectuate, thus specifying the insurance strength. The EAL are defined in insurance increasing order, so that the EAL1 has the lowest verification exigences while EAL7 has the highest ones. In Table 1.2 the objectives given by the CC for each EAL are resumed.

TABLE 1.2: Evaluation Assurance Levels

EAL	Description
EAL1	Functionally tested
EAL2	Structurally tested
EAL3	Methodically tested and checked
EAL4	Methodically designed, tested and reviewed
EAL5	Semi-formally designed and tested
EAL6	Semi-formally verified design and tested
EAL7	Formally verified design and tested

During the process of evaluation, the SFRs of the TOE have to be verified according to the claimed EAL. To this end, the evaluation is divided into six classes of *Security Assurance Requirement* (SAR). Five of this classes are the so-called *conformity* classes, and one in called the *attack* class. Each class is sub-divided in several *families* (excepted the attack class, which only contains one family), and the evaluators are charged to check each requirement corresponding to these families. The table 1.3 resumes the SAR classes and their families. For each family a grade is assigned following precise specifications detailed in CC, and the obtention of a certain EAL depends on the grades obtained for each family, as reported in Table 1.4. An EAL can also be *augmented*, meaning that the product achieves all the required SAR grades to obtain a certain EAL and some upper grades for certain families. For example, smart cards are usually protected at level EAL4+AVA_VAN5+ALC_DVS2, and chips for e-passport application are usually protected at level EAL5+AVA_VAN5+ALC_DVS2. In case of banking smart cards, the card also need to respect the EMVco norms, being EMVco a consortium of six companies (Visa, MasterCard, JCB, American Express, China UnionPay, and Discover) that manages private certification schemes for banking cards, payment terminal and automated teller machines.

The AVA_VAN family and the Attack Potential

The AVA_VAN is the solely family of the vulnerability assessment SAR. The goal of such a SAR is to make the connection between the conformity of the TOE, verified via the analysis of its documentation, and the efficacy of its the protections and countermeasures. This is the step of the evaluation in which the actual resilience of the TOE against the *penetration tests* is measured. In this phase the attacks outlined in Sec. 1.2.1 are taken into account, and the so-called *attack potential* of such attacks is stated. The attack potential is a notion appearing in CC whose aim is to reflect the realism of succeeding a certain attack, and thus its realistic dangerousness. Indeed in the context of physical attacks, many possible attack paths require unrealistic conditions, amounts of time and/or money to be actually performed and do not represent in reality a great risk. For example, invasive attacks such as probing attacks which appears in theory the most dangerous ones, ask in general for some very expensive instruments, a huge expertise, much time and many broken samples before succeeding. Their attack potential can thus result not so wondering. For this evaluation phase, the evaluator is in charge to prepare a testing plan, listing the possibly dangerous attack path, basing on a code analysis, and on the state-of-the-art attacks list in general provided by working groups dedicated to the secured component considered. Once the testing plan is ready he practically tests each attack. For each succeeded attack he fills a *cotation table* in order to assign a score to the attack,

TABLE 1.3: Security Assurance Requirements

Class	Family	Description
Development	ADV_ARC	Security architecture
	ADV_FSP	Functional specification
	ADV_IMP	Implementation representation
	ADV_INT	TOE Security Functions internals
	ADV_SPM	Security policy modelling
	ADV_TDS	TOE design
Guidance Documents	AGD_OPE	Operational user guidance
	AGD_PRE	Preparative procedures
Life-cycle support	ALC_CMC	Configuration Management capabilities
	ALC_CMS	Configuration Management scope
	ALC_DEL	Delivery
	ALC_DVS	Development security
	ALC_FLR	Flaw remediation
	ALC_LCD	Life-cycle definition
	ALC_TAT	Tools and techniques
ST evaluation	ASE_CCL	Conformance claims
	ASE_ECD	Extended components definition
	ASE_INT	ST introduction
	ASE_OBJ	Security objectives
	ASE_REQ	Security requirements
	ASE_SPD	Security problem definition
	ASE_TSS	TOE summary specification
Tests	ATE_COV	Coverage
	ATE_DPT	Depth
	ATE_FUN	Functional tests
	ATE_IND	Independent testing
Vulnerability assessment	AVA_VAN	Vulnerability analysis

TABLE 1.4: Required grades for the obtention of each EAL.

Family	Assurance Components by EAL						
	EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
ADV_ARC		1	1	1	1	1	1
ADV_FSP	1	2	3	4	5	5	6
ADV_IMP				1	1	2	2
ADV_INT					2	3	3
ADV_SPM						1	1
ADV_TDS		1	2	3	4	5	6
AGD_OPE	1	1	1	1	1	1	1
AGD_PRE	1	1	1	1	1	1	1
ALC_CMC	1	2	3	4	4	5	5
ALC_CMS	1	2	3	4	5	5	5
ALC_DEL		1	1	1	1	1	1
ALC_DVS			1	1	1	2	2
ALC_FLR							
ALC_LCD			1	1	1	1	2
ALC_TAT				1	2	3	3
ASE_CCL	1	1	1	1	1	1	1
ASE_ECD	1	1	1	1	1	1	1
ASE_INT	1	1	1	1	1	1	1
ASE_OBJ	1	2	2	2	2	2	2
ASE_REQ	1	2	2	2	2	2	2
ASE_SPD		1	1	1	1	1	1
ASE_TSS	1	1	1	1	1	1	1
ATE_COV		1	2	2	2	3	3
ATE_DPT			1	1	3	3	4
ATE_FUN		1	1	1	1	2	2
ATE_IND	1	2	2	2	2	2	3
AVA_VAN	1	2	2	3	4	5	5

on the base of several criteria. The goal of the cotation table is to provide a metric able to compare very different kind of attacks. The guidelines for the cotation table are given by the *Common Methodology for Information Technology Security Evaluation* (CEM).

In the case of smart cards the evaluation systematically includes the AVA_VAN5 grade, thus the testing plan is asked to be as complete as possible. The state-of-the-art of the attacks is periodically upgrades by the *JIL*[3] *Hardware Attacks Subgroup*, a subgroup of the working committee *Senior Officials Group Information Systems Security* (SOG-IS) which coordinate the standardization of CC. Moreover, the JHAS produces the *Application of Attack Potential to Smartcards* [Lib13] of the JIL, which is an interpretation of the CEM in the special case of smart cards. The cotation table factors specified by the JHAS are detailed in Table 1.5. The evaluation is divided in two parts, an *identification* part, that reflects the difficulty in finding the attack path, and an *exploitation* part, that reflects the difficulty in actually perform the attack. The total score of an attack is the sum of scores assigned to each factor. To obtain the AVA_VAN5 grade every attack tested by the evaluators must have been rated at least 31.

The Evaluation Technical Rapport

The evaluation ends with the redaction by the evaluators of an *Evaluation Technical Rapport*, which is transmitted to the certification body. The last analyses the ETR and, in case the security claims of the TOE are verified, issues a *certificate*. The ETR is kept confidential. Concerning the penetration testing of a certified smart card, the ETR contains all the cotation tables of the succeed attacks. If the component is certified it means that the score of such attacks was higher than 31, and such vulnerability are kept as *residual vulnerabilities*. The ETR will probably be read annually by the evaluators in charge of the surveillance of the certificate. For the penetration testing, the evaluators are in particular asked each year to verify that the cotation of the attacks presented in the ETR did not drop.

1.2.3 This thesis objectives

Among the factors observable in the cotation table 1.5 we find *open samples*, sometimes interpretable as *samples with known secrets*. Indeed, for an evaluation scope it is sometimes possible for an ITSEF to have access to a device identical to the TOE but where the evaluator can fix certain variables, for example some random numbers used by cryptographic algorithm, of load specific software. The evaluator generally exploits this possibility to ease the task of characterization of the behaviour of the device.

In the context of Side-Channel Attacks, when such an characterization phase is possible we talk about *profiling attacks*. Due to the favourable condition of this attacks their are commonly considered the more dangerous, allowing a sort of worst-case security analysis. As we will see in next Chapters, this thesis is mainly focused over the profiling scenario. Indeed, we will address the problems an evaluator deals with when he is in such a favourable case and he wonders how to optimally exploit such a characterization phase in order to be able in the proper exploitation phase to extract

[3] Joint Interpretation Library

TABLE 1.5: Factors of the *Attack Potentials for Smartcards*

Factors	Identification	Exploitation
Elapsed Time		
<one hour	0	0
<one day	1	3
<one week	2	4
<one month	3	6
>one month	5	8
Expertise		
Layman	0	0
Proficient	2	2
Expert	5	4
Multiple Expert	7	6
Knowledge of the TOE		
Public	0	0
Restricted	2	2
Sensitive	4	3
Critical	6	5
Very critical hardware design	9	NA
Access to TOE		
<10 samples	0	0
<30 samples	1	2
<100 samples	2	4
>100 samples	3	6
Equipement		
None	0	0
Standard	1	2
Specialized	3	4
Bespoke	5	6
Multiple Bespoke	7	8
Open Samples		
Public	0	NA
Restricted	2	NA
Sensitive	4	NA
Critical	6	NA

as much information as possible from his signals. One of these issues is the selection of the so-called *Points of Interest* (PoI), strictly link to the more general problem of dimensionality reduction. Indeed,

Chapter 2

Introduction to Side-Channel Attacks

2.1 Introduction to Side-Channel Attacks

2.1.1 Terminology and Generalities

Divide-and-Conquer

- The goal of an SCA is to retrieve a secret part of a cryptographic algorithm, typically the secret key.
- No matter the size of an algorithm inputs, outputs and internal variables, an hardware implementation always operates over variables of a bounded size. Such a bound size depends on the hardware architecture. For example, in an 8-bit architecture an RSA with 1024-bit-sized key, modulo and plaintext will be somehow implemented as multiple operations over 8-bit blocks of observable data.
- This fact allows an attacker to apply the *divide-and-conquer* strategy: if his goal is retrieving the full 128-bit AES key or 1024-bit RSA key, he will smartly divide his problem in retrieving small parts of such keys at time, namely some *subkeys*.
- An attack consists in finding the right association between a trace (or a set of traces) and the value assumed by a target *sensitive variable* Z during the acquisition of such trace/traces.

Sensitive Variable

- Sensitive variable is a piece of info that tells something about a secret of the implementation. Actually, it would be better to call it sensitive target, since it might not be variable
- example: $Z = K$ with K a secret subkey. This is the most direct sensitive target one can imagine, nevertheless it is often not variable, since in some cases a device exploits always the same key for a given embedded primitive. When the target is not variable we are performing a *simple attack* (see later)
- other examples of sensitive variables:
  - the most classical: a cryptographic variable that depends over a sufficiently small subkey

  - any function of of a cryptographic variable (ex: $HW(Z)$), we will see in the discussion about the leakage model in which sense it can be interesting to not target a variable but a non-injective function of a variable
  - an operation (ex:square, multiply)
  - a register (ex: the register used to store results of operations in a montgomery ladder implem of rsa)

- actually, in this thesis we will try as much as possible to abstract from the form of the sensitive variable, thinking of any entity Z that can assume values in a finite set $\mathcal{Z}$ and whose value permit an attacker to make inference about a secret of the implemented algorithm

Leakage Models

- The underlying hypothesis of a SCA is that some information about internal variables (or parts of internal variables) of the implemented algorithm leak during its execution through some observable *side* channels. Such leakages are collectable in the form of signals, observing such channels.

- Depending on the observed channel (*e.g. power consumption, electromagnetic irradiation, time, ...*), different properties might influence the form of the leakage, and should be taken into account for the construction of a leakage model

- If we allow a Side-Channel attacker making use of a probing station to directly access the circuit wires and monitor the exact values of some intermediate values, this attacker will observe leakages following the so-called *probing model*. To define this model no further hypothesis are needed, for example no noise is taken into account. As explained in 1.2.1, SCAs typically refer to non-invasive attacks, so in Side-Channel literature the probing model has been introduced as a worst-case abstract model, and is mainly considered in order to provide formal security proofs for some kinds of countermeasures. More precisely the d-probing model [**ishai2003private**], in which an attacker can probe d different wires at a time, provides a good model to exhibit security proofs for d-th order masking schemes (cf. 2.2.3).

- The most common passive leaking channel considered in literature being the power consumption, for such a physical quantity many efforts have been done to propose adherent leakage models. A detailed modelling for power consumption of CMOS circuits is proposed in the *DPA book* [**mangard2008power**]. After a description of the physical factors influencing the power consumption (divided into static and dynamic) of single logic cells, the authors propose to assume two different points of view to model and develop simulations of the power consumption: the designer point of view can bring to a quite accurate and detailed model, essentially based over his circuit transistor netlists. On the contrary an attacker would be satisfied by considering some easier modelisation, often based over the *Hamming-Distance* (HD) or the *Hamming-Weight* (HW) of internal variables. Indeed these two functions well-fit the consumption behaviour of circuits registers and buses.

- When an attacker has chosen its sensitive target Z and deals with concrete acquisitions, he does not need a complete power model, but only a way to modelise the relative differences between leakages for different values of Z. A

statistical model is then sufficient to him. Thus for an attacker, the wider considered model in Side-Channel community is the one sometimes called *noisy leakage model*. In this model the leakage is a random variable obtained by the sum of a deterministic function of the sensitive variable Z and a random noise. In general the noise has a Gaussian distribution of null mean and variance σ^2:

$$X = \phi(Z) + B\,, \tag{2.1}$$

with $B \approx \mathcal{N}(0, \sigma^2)$.

Simple vs Advanced SCAs

Points of Interest

Vertical vs Horizontal SCAs

Profiled vs Non-Profiled SCAs

Side-Channel Algebraic Attacks

Distinguishers

SCA Metrics

2.2 Main Side-Channel Countermeasures

2.2.1 Random Delays and Jitter

2.2.2 Shuffling

2.2.3 Masking

2.3 Higher-Order Attacks

2.3.1 Higher-Order Moments Analysis and Combining Functions

2.3.2 Profiling Higher-Order Attacks

Profiling with Masks Knowledge

Profiling without Masks Knowledge

Chapter 3

Introduction to Machine Learning

3.1 Basic Concepts of Machine Learning

3.1.1 The Task, the Experience and the Performance

3.1.2 Supervised, Semi-Supervised, Unsupervised Learning

3.1.3 Training, Validation and Test Sets

3.1.4 Underfitting, Overfitting and Regularization

3.1.5 Data Augmentation

3.1.6 No Free Lunch Theorem

3.2 Machine Learning Applications in Side-Channel Context

3.2.1 Profiled Attack as a Classification Problem

Support Vector Machine

Random Forest

Neural Networks

Part II

Contributions

Chapter 4

Introduction

4.1 Foreword of this Thesis: Research of Points of Interest

4.2 Dimensionality Reduction Approach

4.2.1 Linear Methods for First-Order Attacks

4.2.2 Kernel Methods for Higher-Order Attacks

4.3 Neural Network Approach

4.3.1 Toward Getting Rid of Information-Loosing Preprocessing

Chapter 5

Points of Interest

5.1 Motivations

5.1.1 The Curse of Dimensionality

5.2 Selection on Points of Interest: Classical Statistics

5.3 Related Issues: Leakage Detection and Leakage Assessment

test statistici agli ordini superiori, soprattutto per masking scheme hardware, CHES 2017 e precedenti

5.4 Generalized SNR for Multi-Variate Attacks

5.5 Observations Leading to Take a Dimensionality Reduction Approach

Chapter 6

Linear Dimensionality Reduction

6.1 Introduction

6.1.1 Principal Component Analysis

6.1.2 Linear Discriminant Analysis

6.1.3 Projection Pursuits

6.2 Principal Component Analysis

The Principal Component Analysis (PCA) is a technique for data dimensionality reduction. The same PCA algorithm can be deduced from two different points of view, a statistical one and a geometrical one. In the former PCA aims to project orthogonally the data onto a lower-dimensional linear space, the so-called *principal subspace*, such that the variance of the projected data is maximized. In the latter, PCA aims to project data onto a lower-dimensional linear space in such a way that the average projection cost, defined as the mean square distance between the data and their projections, is minimized. In the following it is shown how the PCA algorithm is deduced by the statistical definition. In Appendix B the equivalence between the two approaches is depicted. An example of 2-dimensional data projected over their 1-dimensional principal subspace is depicted in Fig. 6.2

Let $(\mathbf{x}_i)_{i=1..N}$ be a set of D-dimensional measurements (or observations, or data), i.e. realizations of a D-dimensional zero-mean random vector $\vec{X}$, and collect them as columns of an $D \times N$ matrix $\mathbf{M}$, so that the empirical covariance matrix of $\vec{X}$ can

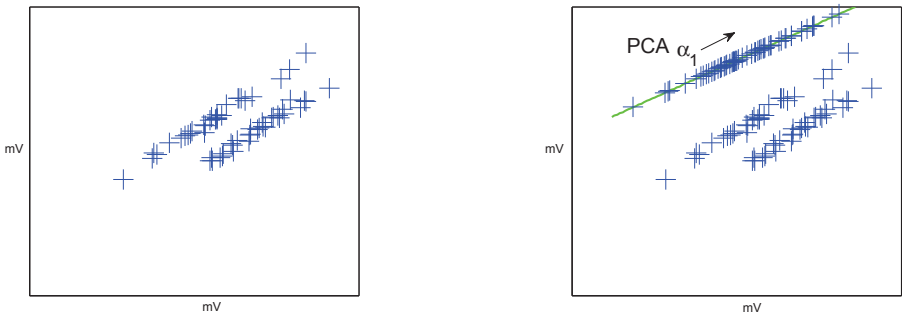


FIGURE 6.1: PCA: some 2-dimensional data projected into their 1-dimensional principal subspace.

be computed as

$$\mathbf{S} = \frac{1}{N}\mathbf{M}\mathbf{M}^\intercal \,. \tag{6.1}$$

Let us assume for the moment that we have fixed the dimension $C < D$ of the principal subspace we are looking for.

Compute the First Principal Component Suppose in a first time that $C = 1$, i.e. that we want to represent our data by a unique variable $Y_1 = \boldsymbol{\alpha}_1\vec{X}$, i.e. projecting data over a single $1 \times D$ vector $\boldsymbol{\alpha}_1$, in such a way the the variance of the obtained data is maximal. The vector $\boldsymbol{\alpha}_1$ that provides such a linear combination is called *first principal component*. To avoid misunderstanding we will call *j-th principal component* (PC) the projecting vector $\boldsymbol{\alpha}_i$, while we will refer to the variable $Y_j = \boldsymbol{\alpha}_j\vec{X}$ as *i-th Principal Variable (PV)*. Realizations of the PVs are given by the measured data projected over the j-th PC, for example we have N realizations of Y_1:

$$y_1[i] = \boldsymbol{\alpha}_1\mathbf{x}_i \text{ for } i = 1, \ldots, N \,. \tag{6.2}$$

Let us collect these realizations in a vector $\mathbf{y}_1 = \boldsymbol{\alpha}_1\mathbf{M}$; the mean of these realizations will be zero as they are linear combinations of zero-mean variables, and the variance turns to be expressible as

$$\frac{1}{N}\mathbf{y}_1\mathbf{y}_1^\intercal = \frac{1}{N}\boldsymbol{\alpha}_1\mathbf{M}\mathbf{M}^\intercal\boldsymbol{\alpha}_1^\intercal = \boldsymbol{\alpha}_1\mathbf{S}\boldsymbol{\alpha}_1^\intercal \,. \tag{6.3}$$

To compute $\boldsymbol{\alpha}_1$ we have to look for the vector that maximize such a variance.

Obviously, we can attend a value for the variance as high as wished, raising the modulo $\|\boldsymbol{\alpha}_1\| = \sqrt{\boldsymbol{\alpha}_1^T\boldsymbol{\alpha}_1}$. In order to let the maximization problem have a solution, we impose a restriction over it: $\boldsymbol{\alpha}_1^T\boldsymbol{\alpha}_1 = 1$.
Let us handle this constrained optimization problem making use of Lagrange multipliers:

$$\Lambda(\boldsymbol{\alpha}_1, \lambda) = \boldsymbol{\alpha}_1\mathbf{S}\boldsymbol{\alpha}_1^\intercal - \lambda(\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\intercal - 1) \tag{6.4}$$

and let us compute de partial derivative of Λ with respect to $\boldsymbol{\alpha}_1$

$$\frac{\partial\Lambda}{\partial\boldsymbol{\alpha}_1} = 2\mathbf{S}\boldsymbol{\alpha}_1^\intercal - 2\lambda\boldsymbol{\alpha}_1^\intercal \,. \tag{6.5}$$

Thus, stationary points of Λ verify

$$\mathbf{S}\boldsymbol{\alpha}_1^\intercal = \lambda\boldsymbol{\alpha}_1^\intercal \,, \tag{6.6}$$

which implies that $\boldsymbol{\alpha}_1^\intercal$ must be an eigenvector of $\mathbf{S}$, with λ its correspondent eigenvalue. Multiplying both sides of Eq. (6.6) by $\boldsymbol{\alpha}_1$ on the left, we remark that

$$\boldsymbol{\alpha}_1\mathbf{S}\boldsymbol{\alpha}_1^\intercal = \lambda\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\intercal = \lambda, \tag{6.7}$$

which means that the variance of the obtained variable $\mathbf{y}_1$ equals λ. For this reason $\boldsymbol{\alpha}_1$ must be the leading eigenvector of $\mathbf{S}$.

Compute the Second and following Principal Component The PCs following the first one are defined in an incremental fashion by choosing new directions orthogonal to those already considered and such that the sum of the projected variances

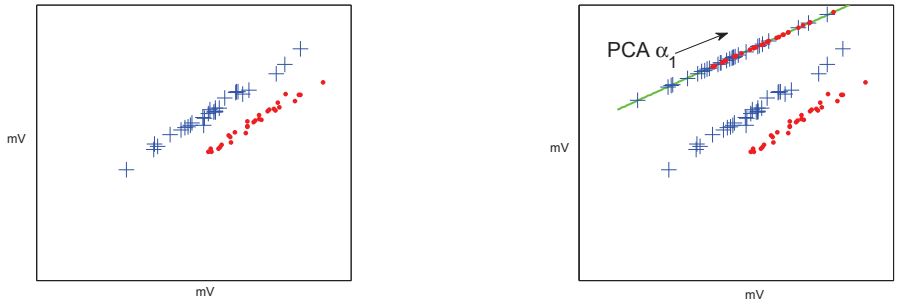


FIGURE 6.2: PCA: some 2-dimensional data projected into their 1-dimensional principal subspace.

over each direction is maximal. Explicitly, if we look for two PCs, *i.e.* $C = 2$, we look for a 2-dimensional variable $\mathbf{Y} = [\begin{smallmatrix}\boldsymbol{\alpha}_1\\ \boldsymbol{\alpha}_2\end{smallmatrix}]\vec{X}$ such that the trace of its covariance matrix, *i.e.* the sum of variances $\mathrm{Var}(Y_1) + \mathrm{Var}(Y_2)$, is maximal. It can be shown that the same result would be obtained maximizing the so-called *generalized variance* of $\mathbf{Y}$, which is defined as the determinant of its covariance matrix, instead of its trace.

Let us write, as in previous case, the Lagrangian of the problem

$$\Lambda = \boldsymbol{\alpha}_1 \mathbf{S} \boldsymbol{\alpha}_1^\intercal + \boldsymbol{\alpha}_2 \mathbf{S} \boldsymbol{\alpha}_2^\intercal - \lambda_1(\boldsymbol{\alpha}_1 \boldsymbol{\alpha}_1^\intercal - 1) - \lambda_2(\boldsymbol{\alpha}_2 \boldsymbol{\alpha}_2^\intercal - 1)\,. \tag{6.8}$$

Partial derivatives with respect to $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ attend zero under the following conditions:

$$\mathbf{S}\boldsymbol{\alpha}_1^\intercal = \lambda_1 \boldsymbol{\alpha}_1^\intercal \tag{6.9}$$
$$\mathbf{S}\boldsymbol{\alpha}_2^\intercal = \lambda_2 \boldsymbol{\alpha}_2^\intercal\,, \tag{6.10}$$

which means that $\boldsymbol{\alpha}_1^\intercal$ and $\boldsymbol{\alpha}_2^\intercal$ must be eigenvectors of $\mathbf{S}$ with correspondent eigenvalues given by λ_1 and λ_2. Moreover, as before, λ_1 and λ_2 equal the variances of variable components Y_1 and Y_2, and since we want to maximize the sum of such variables we have to choose $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ as the two leading vectors of $\mathbf{S}$.
Let us remark that the covariance between Y_1 and Y_2 is given by $\boldsymbol{\alpha}_1^T \mathbf{S} \boldsymbol{\alpha}_2$ which equals zero, since $\boldsymbol{\alpha}_1^T \boldsymbol{\alpha}_2 = 0$, by orthogonality. In particular the principal variables are uncorrelated, which is a remarkable property of the PCA.

In the general case of a C-dimensional projection space, it can be shown by induction that the PCs would corresponds to the C leading eigenvectors of the covariance matrix $\mathbf{S}$.

6.3 Application of PCA in SCAs

6.3.1 Original vs Class-Oriented PCA

We remark that the classical version of PCA method does not take into account the information about the value assumed by the target variable during the acquisition of data: actually our attacker is provided not only with a set of data $(\mathbf{x}_i)_{i=1..N}$, but she can group such traces depending on the target value: this let her obtain a set of traces $(\mathbf{x}_i)_{i=1..N}$ for each value of the target variable Z.

In this context, and for the sake of distinguish the target value assumed by the variable Z in new executions, we want to consider *equivalent* all traces coming from each group, i.e. obtained by a same characteristic form plus a random noise. To estimate the characteristic form we compute the sample means of the traces of each group obtaining $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1..N}\mathbf{x}_i$. These sample means will take the place of data in PCA: we collect them as rows in the data matrix $\mathbf{M}$, shift the empirical mean of each component to zero, compute the covariance matrix $\mathbf{S}$ et select the leading C eigenvectors of $\mathbf{S}$ as optimal projecting directions.
In this way we focus the attention on information that discriminate classes, i.e. target values; this is why we will refer to this version of PCA as *class-oriented version*.

6.3.2 The Choice of the Principal Components

The introduction of the PCA method in SCA context (either in its classical or class-oriented version) has raised some important questions: *how many* principal components and *which ones* are sufficient/necessary to reduce the trace size (and thus the attack processing complexity) without losing important discriminative information?

Until now, an answer to the questions above has been given in [CK14], linked to the concept of *explained variance* (or *explained global variance*, EGV for short) of a PC $\boldsymbol{\alpha}_i$:

$$\mathrm{EGV}(\boldsymbol{\alpha}_i) = \frac{\lambda_i}{\sum_{k=1}^{r}\lambda_k}\,, \tag{6.11}$$

where r is the rank of the covariance matrix $\mathbf{S}$, and λ_j is the eigenvalue associated to the j-th PC $\boldsymbol{\alpha}_j$. $\mathrm{EGV}(\boldsymbol{\alpha}_i)$ is the variance of the data projected over the i-th PC (which equals λ_i) divided by the total variance of the original data (given by the trace of the covariance matrix $\mathbf{S}$, *i.e.* by the sum of all its non-zero eigenvalues). By definition of EGV, the sum of all the EGV values is equal to 1; that is why this quantity is often multiplied by 100 and expressed as percentage. Exploiting the EGV to choose among the PCs consists in fixing a wished *cumulative explained variance* β and in keeping C different PCs, where C is the minimum integer such that

$$\mathrm{EGV}(\boldsymbol{\alpha}_1) + \mathrm{EGV}(\boldsymbol{\alpha}_2) + \cdots + \mathrm{EGV}(\boldsymbol{\alpha}_C) \geq \beta\,. \tag{6.12}$$

However, if the adversary has a constraint for the reduced dimension C, the EGV notion simply suggests to keep the first C components, taking for granted that the optimal way to chose PCs is in their natural order. This assumption is not always confirmed in SCA context: in some works, researchers have already remarked that the first components sometimes contain more noise than information [BHW12; Spe+15] and it is worth discarding them. For the sake of providing a first example of this behaviour on publicly accessible traces, we applied a class-oriented PCA on 3000 traces from the DPA contest v4 [Par]; we focused over a small 1000-dimensional window in which, in complete knowledge about masks and other countermeasures, information about the first Sbox processing leaks (during the first round). In Fig. 6.3 the first and the sixth PCs are plotted. It may be noticed that the first component indicates that one can attend a high variance by exploiting the regularity of the traces, given by the clock signal, while the sixth one has high coefficients localised in a small time interval, very likely to signalize the instants in which the target sensitive variable leaks.

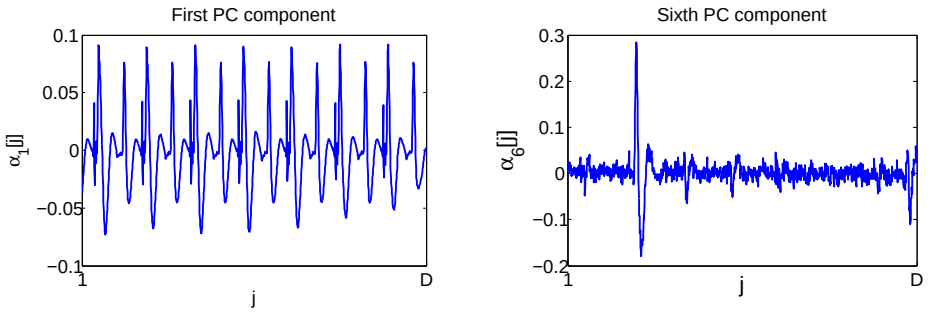


FIGURE 6.3: First and sixth PCs in DPA contest v4 trace set (between time samples 198001 and 199000)

To the best of our knowledge, a single method adapted to SCA context has been proposed until now to automatically choose PCs [Mav+12] while dealing with the issue raised in Fig. 6.3. It is based on the following assumption:

Assumption 1. The leaking side-channel information is localised in few points of the acquired trace.

In the rest of the paper, we conduct our own analyses under Assumption 1 that we think to be reasonable in SCA contexts where the goal of the security developers is to minimize the number of leaking points. Under this assumption, the authors of [Mav+12] use for side-channel attack purposes the *Inverse Participation Ratio* (IPR), a measure widely exploited in Quantum Mechanics domain (see for example [**guhr1998random**]). They propose to use such a score to evaluate the eigenvectors *localization*. It is defined as follows:

$$\mathrm{IPR}(\boldsymbol{\alpha}_i) = \sum_{j=1}^{D} \boldsymbol{\alpha}_i[j]^4 \,. \tag{6.13}$$

The authors of [Mav+12] suggest to collect the PCs in decreasing order with respect to the IPR score.

The selection methods provided by the evaluation of the EGV and of the IPR are somehow complementary: the former is based only on the eigenvalues associated to the PCs and does not consider the form of the PCs themselves; the latter completely discards the information given by the eigenvalues of the PCs, considering only the distribution of their coefficients. One of the contributions of the present paper is to propose a new selection method, that builds a bridge between the EGV and the IPR approaches. As we will argue, our method, based on the so-called *explained local variance*, does not only lead to the construction of a new selection criterion, but also permits to modify the PCs, choosing individually the coefficients to keep and those to discard.

6.3.3 The Explained Local Variance Selection Method

Riprendere le notazioni e mettere apposto i newcommand

Definition 1. The *Explained Local Variance* of a PC $\boldsymbol{\alpha}_i$ in a sample j, is defined by

$$\mathrm{ELV}(\boldsymbol{\alpha}_i, j) = \frac{\lambda_i \boldsymbol{\alpha}_i[j]^2}{\sum_{k=1}^{r} \lambda_k} = \mathrm{EGV}(\boldsymbol{\alpha}_i)\boldsymbol{\alpha}_i[j]^2 \,. \tag{6.14}$$

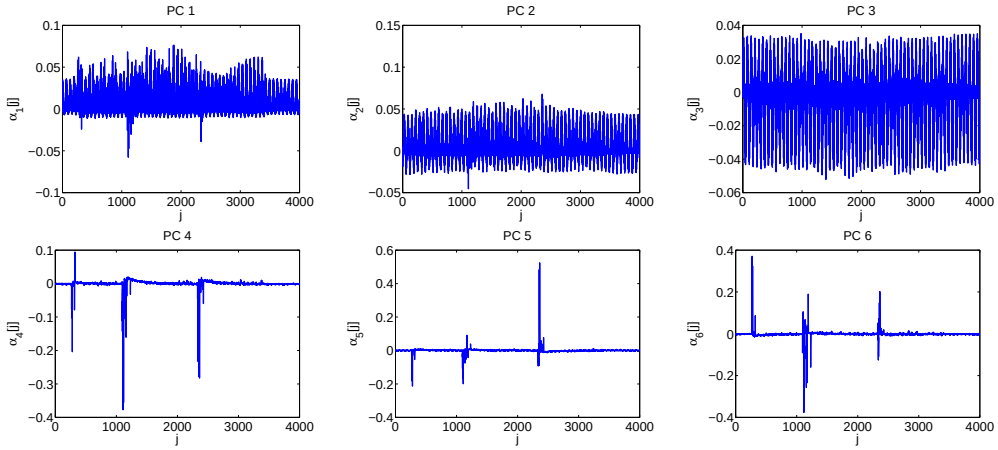


FIGURE 6.4: The first six PCs. Acquisition campaign on an 8-bit AVR Atmega328P (see Sec. 6.6).

Let $\mathcal{J} = \{j_1^i, j_2^i, \ldots, j_D^i\} \subset \{1, 2, \ldots, D\}$ be a set of indexes sorted such that $\mathrm{ELV}(\boldsymbol{\alpha}_i, j_1^i) \geq \mathrm{ELV}(\boldsymbol{\alpha}_i, j_2^i) \geq \cdots \geq \mathrm{ELV}(\boldsymbol{\alpha}_i, j_D^i)$. It may be observed that the sum over all the $\mathrm{ELV}(\boldsymbol{\alpha}_i, j)$, for $j \in [1, \ldots, D]$, equals $\mathrm{EGV}(\boldsymbol{\alpha}_i)$. If we operate such a sum in a cumulative way following the order provided by the sorted set $\mathcal{J}$, we obtain a complete description of the trend followed by the component $\boldsymbol{\alpha}_i$ to achieve its EGV. As we can see in Fig. **??**, where such cumulative ELVs are represented, the first 3 components are much slower in achieving their final EGV, while the 4^{th}, the 5^{th} and the 6^{th} achieve a large part of their final EGVs very quickly (*i.e.* by adding the ELV contributions of much less time samples). For instance, for $i = 4$, the sum of the $\mathrm{ELV}(\boldsymbol{\alpha}_4, j_k^4)$, with $k \in [1, \ldots, 30]$, almost equals $\mathrm{EGV}(\boldsymbol{\alpha}_4)$, whereas the same sum for $i = 1$ only achieves about the 15% of $\mathrm{EGV}(\boldsymbol{\alpha}_1)$. Actually, the EGV of the 4^{th}, the 5^{th} and the 6^{th} component only essentially depends on a very few time samples. This observation, combined with Assumption 1, suggests that they are more suitable for SCA than the three first ones. To validate this statement, it suffices to look at the form of such components (Fig. 6.4): the leading ones are very influenced by the clock, while the latest ones are well localised over the leaking points.

Operating a selection of components *via* ELV, in analogy with the EGV, requires to fix the reduced space dimension C, or a threshold β for the cumulative ELV. In the first case, the maximal ELVs of each PC are compared, and the C components achieving the highest values of such ELVs are chosen. In the second case, all pairs (PC, time sample) are sorted in decreasing order with respect to their ELV, and summed until the threshold β is achieved. Then only PCs contributing in this sum are selected.

We remark that the ELV is a score associated not only to the whole components, but to each of their coefficients. This interesting property can be exploited to further remove, within a selected PC, the non-significant points, *i.e.* those with a low ELV. In practice this is done by setting these points to zero. That is a natural way to exploit the ELV score in order to operate a kind of *denoising* for the reduced data, making them only depend on the significant time samples. In Sec. 6.6 (scenario 4) we test the performances of an attack varying the number of time samples involved in the computation of the reduced data, and showing that such a denoising processing might impact significantly.

6.4 Linear Discriminant Analysis

6.4.1 Statistical Point of View

6.4.2 Geometrical Point of View

6.5 Application of LDA in SCAs

6.5.1 The Small Sample Size problem

6.6 Experimental Results

In this section we compare the different extractors provided by the PCA and the LDA in association with the different techniques of components selection. Defining an universal criterion to compare the different extractors would not make sense since the latter one should encompass a lot of parameters, sometimes opposite, that vary according to the context (amount of noise, specificity of the information leakage, nature of the side channel, etc.). For this reason we choose to split our comparisons into four scenarios. Each scenario has a single varying parameter that, depending on the attacker context, may wish to be minimized. Hereafter the definition of the four scenario. In the following only results of the two first is reported, the interested reader might refer to AppendixA for results of in the two other scenarios.

Scenario 1 varying parameter: number of attack traces N_a,

Scenario 2 varying parameter: number of profiling traces N_p,

Scenario 3 varying parameter: number of projecting components selected C,

Scenario 4 varying parameter: number of original time samples implied into the trace preprocessing computation $\sharp$PoI .

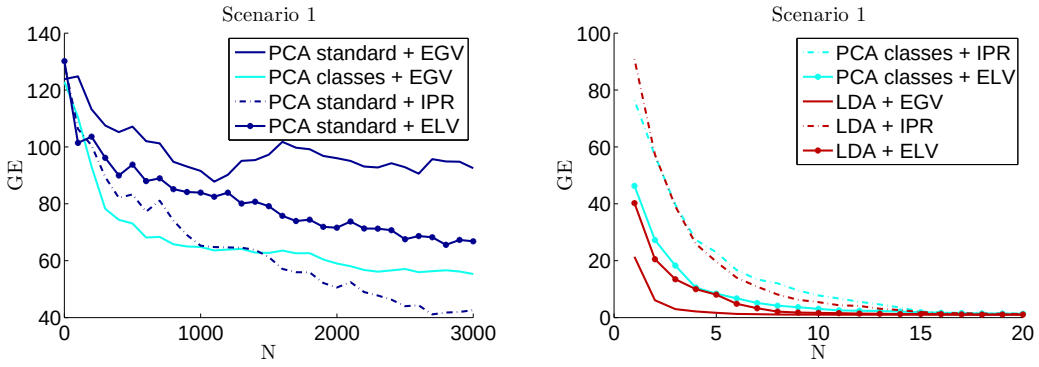
For scenarios in which N_p is fixed, the value of N_p is chosen high enough to avoid the SSS problem, and the extensions of LDA presented in Sec. **??** are not evaluated. This choice of N_p will imply that the LDA is always performed in a favourable situation, which makes expect the LDA to be particularly efficient for these experiments. Consequentely, for the scenarios in which N_p is high, our goal is to study whether the PCA can be made almost as efficient as the LDA thanks to the component selection methods discussed in Sec. 6.3.2.

This part will maybe be useless: somewhere I will have described all trace sets

The testing adversary.

Scenario 1.

cos'e N_z To analyse the dependence between the extraction methods presented in Sections **??** and **??** and the number of attack traces N_a needed to achieve a given GE, we fixed the other parameters as follows: $N_z = 50$ ($N_p = 50 \times 256$), $C = 3$ and $\sharp\text{PoI} = 3996$ (all points are allowed to participate in the building of the PCs and of the LDCs). The experimental results, depicted in Fig. 6.5(a)-(b), show that the PCA standard method has very bad performances in SCA, while the LDA outperforms the others. Concerning the class-oriented PCA, we observe that its performance is close to that of LDA when combined with the selection methods ELV (which performs best) or IPR.



(a) (b)

FIGURE 6.5: Guessing Entropy as function of the number of attack traces for different extraction methods. All Guessing Entropies are estimated as the average rank of the right key over 100 independent experiments.

Scenario 2.

Now we test the behaviour of the extraction methods as function of the number N_z of available profiling traces per class. The number of components C is still fixed to 3, $\sharp\text{PoI} = 3996$ again and the number of attack traces is $N_a = 100$. This scenario has to be divided into two parts: if $N_z \leq 15$, then $N_p < D$ and the SSS problem occurs. Thus, in this case we test the four extensions of LDA presented in Sec. **??**, associated to either the standard selection, to which we abusively refer as EGV,[1] or to the IPR selection. We compare them to the class-oriented PCA associated to EGV, IPR or ELV. The ELV selection is not performed for the techniques extending LDA, since for some of them the projecting LDCs are not associated to some eigenvalues in a meaningful way. On the contrary, if $N_z \geq 16$ there is no need to approximate the LDA technique, so the classical one is performed. Results for this scenario are shown in Fig. 6.6. It may be noticed that the combinations class-oriented PCA + ELV/IPR select exactly the same components, for our data, see Fig. 6.6(e) and do not suffer from the lack of profiling traces. They are slightly outperformed by the $\mathbf{S_W}$ Null Space method associated with the EGV, see Fig.6.6(d). The Direct LDA (Fig. 6.6(b)) method also provides a good alternative, while the other tested methods do not show a stable behaviour. The results in absence of the SSS problem (Fig.6.6(f)) confirm that the standard PCA is not adapted to SCA, even when provided with more profiling traces. It also shows that among class-oriented PCA and LDA, the class-oriented PCA converges faster.

6.7 Misaligning Effects

give parameters: 6 4 citare Choudary, Template Attacks over different devices In this section we experimentally show how the approach based on linear dimensionality reduction described in this chapter is affected by traces misalignment. To this aim, we simply take the same data and parameters exploited for Scenario 1 in Sec. 6.6, and artificially misalign them through the technique proposed in Appendix A.2 with parameters **parameters here**. Then we tried to pre-process attack them through the 9 methodology tested in Scenario 1. It may be noticed in Fig. 6.7

[1] It consists in keeping the C first LDCs (the C last for the Direct LDA)

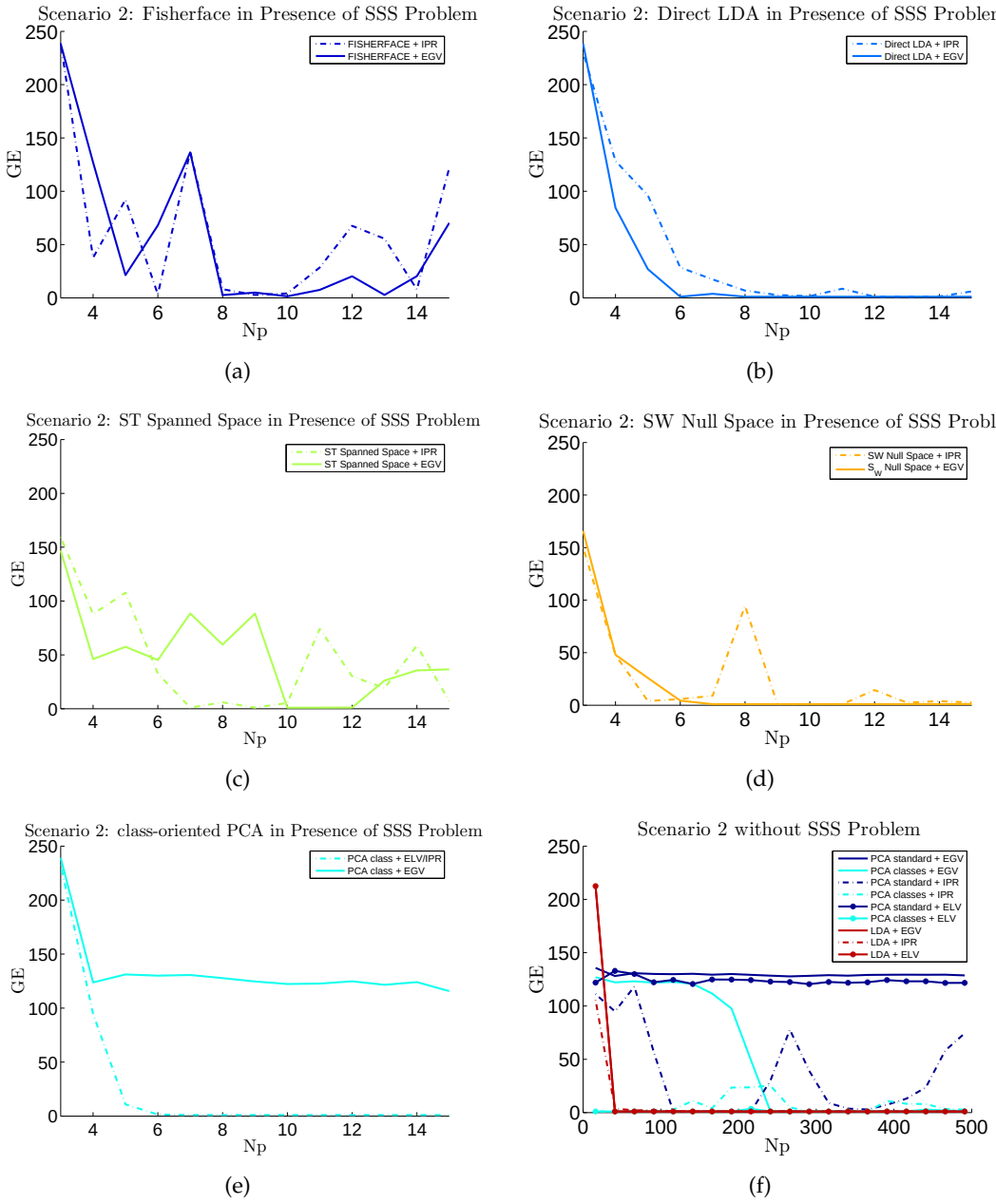


FIGURE 6.6: Guessing Entropy as function of the number of profiling traces. Figures (a)-(d): methods extending the LDA in presence of SSS problem; Figure (e): class-oriented PCA in presence of the SSS problem; Figure (f): number of profiling traces high enough to avoid the SSS problem.

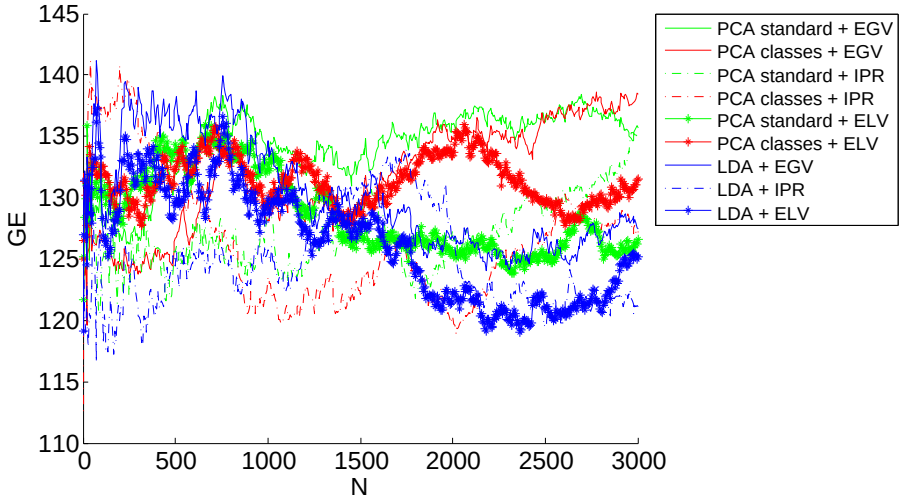


FIGURE 6.7: Degradation of linear-reduction-based template attacks in presence of misalignment.

that none of the 9 techniques is still efficient, included the optimal LDA+EGV that lead to null guessing entropy the synchronized traces using less 7 attack traces. In this case it cannot lead to successful attack in less than 3000 traces.

Chapter 7

Kernel Dimensionality Reduction

7.1 Motivation

7.1.1 Higher-Order Attacks

Higher-Order Version of Projection Pursuits

7.2 Kernel Function and Kernel Trick

7.2.1 Local Kernel Functions as Similarity Metrics

7.3 Kernel Discriminant Analysis

7.4 Experiments over Atmega328P

7.4.1 The Regularization Problem

7.4.2 The Multi-Class Trade-Off

7.4.3 Multi-Class vs 2-class Approach

7.4.4 Asymmetric Preprocessing/Attack Approach

Comparison with Projection Pursuits

7.5 Drawbacks of Kernel Methods

Misalignment Effects

Memory Complexity and Actual Number of Parameters

Two-Phases Approach: Preprocessing-Templates

Chapter 8

Convolutional Neural Networks against Jitter-Based Countermeasures

Chapter 9

KDA vs Neural Networks Approach for HO-Attacks

9.1 Simulated Experiment for Profiled HO-Attacks

9.1.1 The Simulations

9.1.2 Comparison between KDA and MLP

9.2 Real-Case Experiments over ARM Cortex-M4

Chapter 10

Siamese Neural Networks for Collision Attacks

Chapter 11

Conclusions and Perspectives

11.1 Summary

11.2 Strengthen Embedded Security: the Main Challenge for Machine Learning Applications

Appendix A

Scenario 3 and 4 of CARDIS '15 paper

A.1 Scenario 3.

Let C be now variable and let the other parameters be fixed as follows: $N_a = 100, N_z = 200, \sharp\text{PoI} = 3996$. Looking at Fig. **??**, we might observe that the standard PCA might actually well perform in SCA context if provided with a larger number of kept components; on the contrary, a little number of components suffices to the LDA. Finally, keeping more of the necessary components does not worsen the efficiency of the attack, which allows the attacker to choose C as the maximum value supported by his computational means.

Remark. In our experiments the ELV selection method only slightly outperforms the IPR. Nevertheless, since it relies on more sound and more general observations, *i.e.* the maximization of explained variance concentrated into few points, it is likely to be more robust and less case-specific. For example, in Fig. 6.6(f) it can be remarked that while the class-oriented PCA + ELV line keeps constant on the value 0 of guessing entropy, the class-oriented PCA + IPR is sometimes higher than 0.

Is the table with results overview interesting?

A.2 Scenario 4.

This is the single scenario in which we allow the ELV selection method to not only select the components to keep but also to modify them, keeping only some coefficients within each component, setting the other ones to zero. We select pairs *(component, time sample)* in decreasing order of the ELV values, allowing the presence of only $C = 3$ components and $\sharp\text{PoI}$ different times samples: *i.e.*, we impose that the matrix A defining the extractor (see (**??**)) has $C = 3$ rows (storing the 3 chosen components, transposed) and exactly $\sharp\text{PoI}$ non-zero columns. Looking at Fig. **??** we might observe that the LDA allows to achieve the maximal guessing entropy with only 1 PoI in each of the 3 selected components. Actually, adding PoIs worsen its performances, which is coherent with the assumption that the vulnerable information leaks in only a few points. Such points are excellently detected by the LDA. Adding contribution from other points raises the noise, which is then compensated by the contributions of further noisy points, in a very delicate balance. Such a behaviour is clearly visible in standard PCA case: the first 10 points considered raise the level of noise, that is then balanced by the last 1000 points.

Artificially Simulate Jitter

Appendix B

Geometrical Definition of PCA

Bibliography

[BHW12] Lejla Batina, Jip Hogenboom, and Jasper G.J. van Woudenberg. "Getting More from PCA: First Results of Using Principal Component Analysis for Extensive Power Analysis". English. In: *Topics in Cryptology CT-RSA 2012*. Ed. by Orr Dunkelman. Vol. 7178. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 383–397. ISBN: 978-3-642-27953-9. DOI: 10.1007/978-3-642-27954-6_24. URL: http://dx.doi.org/10.1007/978-3-642-27954-6_24 (cit. on p. 30).

[CK14] Omar Choudary and Markus G Kuhn. "Efficient template attacks". In: *Smart Card Research and Advanced Applications*. Springer, 2014, pp. 253–270 (cit. on p. 30).

[Gen+15] Daniel Genkin et al. "Stealing keys from PCs using a radio: Cheap electromagnetic attacks on windowed exponentiation". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 2015, pp. 207–228 (cit. on p. 6).

[Gen+16] Daniel Genkin et al. "ECDH key-extraction via low-bandwidth electromagnetic attacks on PCs". In: *Cryptographers' Track at the RSA Conference*. Springer. 2016, pp. 219–235 (cit. on p. 6).

[GMO01] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. "Electromagnetic analysis: Concrete results". In: *Cryptographic Hardware and Embedded Systems - CHES 2001*. Springer. 2001, pp. 251–261 (cit. on p. 6).

[GPT15] Daniel Genkin, Itamar Pipman, and Eran Tromer. "Get your hands off my laptop: Physical side-channel key-extraction attacks on PCs". In: *Journal of Cryptographic Engineering* 5.2 (2015), pp. 95–112 (cit. on p. 6).

[GST14] Daniel Genkin, Adi Shamir, and Eran Tromer. "RSA key extraction via low-bandwidth acoustic cryptanalysis". In: *International Cryptology Conference*. Springer. 2014, pp. 444–461 (cit. on p. 6).

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. "Differential power analysis". In: *Annual International Cryptology Conference*. Springer. 1999, pp. 388–397 (cit. on p. 6).

[Koc96] 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 (cit. on p. 6).

[Lib13] Joint Interpretation Library. *Application of Attack Potential to SmartSmart, Version 2.9*. Tech. rep. Common Criteria, 2013 (cit. on p. 12).

[Mav+12] Dimitrios Mavroeidis et al. "PCA, Eigenvector Localization and Clustering for Side-Channel Attacks on Cryptographic Hardware Devices". English. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by PeterA. Flach, Tijl De Bie, and Nello Cristianini. Vol. 7523. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 253–268. ISBN: 978-3-642-33459-7. DOI: 10.1007/978-3-642-33460-

3_22. URL: http://dx.doi.org/10.1007/978-3-642-33460-3_22 (cit. on p. 31).

[NIS] FIPS PUB NIST. *197," Advanced Encryption Standard (AES)," November 2001* (cit. on pp. 4, 5).

[Par] TELECOM ParisTech. "DPA Contest 4". In: (). http://www.DPAcontest.org/v4/. URL: http://www.DPAcontest.org/v4/ (cit. on p. 30).

[QS01] Jean-Jacques Quisquater and David Samyde. "Electromagnetic analysis (ema): Measures and counter-measures for smart cards". In: *Smart Card Programming and Security* (2001), pp. 200–210 (cit. on p. 6).

[Spe+15] Robert Specht et al. "Improving Non-Profiled Attacks on Exponentiations Based on Clustering and Extracting Leakage from Multi-Channel High-Resolution EM Measurements". In: *Sixth International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2015)*. 2015 (cit. on p. 30).