A Formal Practice-Oriented Model for the Analysis of Side-Channel Attacks

François-Xavier Standaert[1,2,⋆], Tal G. Malkin[1], Moti Yung[1,3]

[1] Dept. of Computer Science, Columbia University.
[2] UCL Crypto Group, Université Catholique de Louvain.
[3] RSA Laboratories.

e-mails: fstandae@uclouvain.be, tal,moti@cs.columbia.edu

Version 1.3, November 10, 2006.

Abstract. Formal models that allow one to understand side-channel attacks and are also directly meaningful to practice have been an open question. Motivated by this challenge, this work proposes a practice oriented framework for the analysis of cryptographic implementations against such attacks. It is illustratively applied to block ciphers, although it could be used to analyze a larger class of cryptosystems. The model is based on weak and commonly accepted hypotheses about side-channels that computations give rise to. It allows us to quantify the effect of practically relevant leakage functions with a combination of security and information theoretic metrics. From a practical point of view, the model suggests a unified evaluation methodology for side-channel attacks. From a theoretical point of view, it allows discussing the fundamental tradeoffs in such attacks, namely flexibility *vs.* efficiency and information *vs.* computation.

1 Introduction

Traditionally, cryptographic algorithms provide security against an adversary who has only black box access to cryptographic devices. That is, the only thing the adversary can do is to query the cryptographic algorithm on inputs of its choice and analyze the responses, which are always computed according to the correct original secret information. However, such a model does not always correspond to the realities of physical implementations, and actually very rarely does. During the last decade, significant attention has been paid to the physical security evaluation of cryptographic devices. In particular, it has been demonstrated that actual attackers may be much more powerful than what can be captured by the black box model.

In this paper, we investigate the security of cryptographic implementations with respect to side-channel attacks, in which adversaries are enhanced with the possibility to exploit physical leakages such as power consumption or electromagnetic radiation. A large body of experimental work has been created on the subject, *e.g.* [1, 2, 4, 6, 10, 11, 14, 18, 21, 24, 25, 32, 37, 39, 41, 45], and although numerous countermeasures are proposed in the literature, *e.g.* [3, 9, 16, 17, 20, 30, 38, 43, 48, 49], protecting implementations against such attacks is usually difficult and expensive. Moreover, most proposals we are aware of only increase the difficulty of performing the attacks, but do not fundamentally prevent them [13, 26–28, 31, 35, 36, 46]. As a consequence, their cost *vs.* efficiency evaluation is a critical design task for cryptographic designers.

[⋆] François-Xavier Standaert is a post doctoral researcher funded by the FNRS (Funds for National Scientific Research, Belgium).

Perhaps surprisingly (and to the best of our knowledge), there have been only a few attempts to model such physical attacks properly, and to provably address their security. A notable example is the work of Micali and Reyzin who initiated a theoretical analysis of side-channels, taking the modularity of physically observable computations into account. It notably defines the notion of *physical computer* that is basically the combination of an abstract computer (*i.e.* a Turing machine) and a leakage function. The model in [33] is very general, capturing almost any conceivable form of physical leakage. However, arguably because of the great generality of the assumptions, the obtained positive results (*i.e.* leading to useful constructions) are quite restricted in nature, and it is not clear how they apply to practice. This is especially true for primitives such as modern block ciphers (*e.g.* the DES or AES Rijndael) for which even the black box security cannot be proven. Thus, the study of more specialized contexts and specific scenarios which may lead to practical applications was suggested as an open question. Motivated by this challenge, we propose to analyze side-channel attacks in a model of computation that captures the structure and operations of modern block ciphers. Still, the model is general and can be used to analyze other cryptosystems.

With many respects, our following results can be viewed as a specialization of the Micali and Reyzin setting with three distinct objectives:

1. To meaningfully *restrict* the most general assumptions of [33] to reasonable (*i.e.* practically relevant) adversaries and leakage functions.
2. To *relate* the abstract (*i.e.* Turing machine-based) computation model of [33] to more intuitive physical notions (*e.g.* circuits, signals and operations).
3. To *quantify* the side-channel information leakages with sound criteria.

Otherwise said, we aim to reduce the gap between the previously introduced theoretical notions of physical security and the actual side-channel attacks. So basically, we would like to trade some theoretical generality for more applicability to various designs.

From a practical point of view, our framework suggests a unified evaluation methodology for side-channel attacks in which we measure the effect of practically relevant leakage functions with a combination of security and information theoretic metrics. The security metric aims to discriminate different adversaries and corresponds to the formal notion of side-channel key recovery (defined in the paper). The information theoretic metric (namely the mutual information) aims to discriminate different implementations independently of the adversary and its computational power. By combining both metrics, the model allows answering the important following questions in the investigation of physical security issues, namely:

1. How to quantify the amount of information provided by a given physical computer?
2. How successfully can an adversary turn this information into a practical attack?

From a theoretical point of view, the introduced model and metrics allow the discussion of the fundamental tradeoffs in side-channel attacks, namely flexibility *vs.* efficiency and information *vs.* computation. It also provides a sound background for the construction of primitives with provable security against such adversaries and a more formal understanding of the underlying mechanisms in physically observable cryptography.

The rest of this paper is structured as follows. Section 2 recalls certain definitions introduced in [33] that are necessary for the understanding of our results. Section 3 gives an intuitive description of our target circuit for physically secure applications and provides details about the class of attacks we want to prevent. It discusses how this intuitive description can be translated into the model of [33]. Section 4 details the block cipher to which security against side-channel attacks is to be analyzed. We note that this target block cipher is described for concreteness, but much of our work is independent of these details. Section 5 specifies the adversarial context and strategy we consider in our analysis. Section 6 and 7 respectively introduce the notion of leakage prediction functions and their classification, the formal definition of a side-channel adversary and the computational restrictions that we suggest to impose to such adversaries. Section 8 defines the notion of security against side-channel key recovery. Section 9 and 10 describe our evaluation criteria for side-channel attacks and the resulting analysis and comparison methodology. Section 11 finally summarizes the tradeoffs actual adversaries have to face in physically observable cryptography, namely flexibility *vs.* efficiency and information *vs.* computation. Our conclusion and list of open problems are in Section 12. In addition, Appendix B discusses the need and relevance of the introduced metrics with respect to previously introduced solutions.

2 Background: Micali-Reyzin computational model

In order to enable the analysis of physically observable cryptography, Micali and Reyzin introduced a model of computation of which we recall certain definitions of interest with respect to our following results. It is based on five informal axioms [33]:

Axiom 1. Computation and only computation leaks information.

That is, we assume that it is possible to store some secret information securely in a cryptographic device. No leakages will compromise this secret as long as it is not used in any computation. As a matter of fact, this implies that probing attacks are out of the scope of our analysis and we rely on physical protections to prevent them.

Axiom 2. The same computation leaks different information on different computers.

In other words, an algorithm is an abstraction: a set of general instructions whose physical implementation may vary. As a result, the same elementary operation may leak different information on different platforms.

Axiom 3. The information leakage depends on the chosen measurement.

The amount of information that is recovered by an adversary during a side-channel attack depends on the measurement process, that possibly introduces some randomness.

Axiom 4. The information leakage is local.

In other words, the maximum amount of information that may be leaked by a physically observable device is the same in any execution of the algorithm with the same inputs, since it relates to the target device's internal configuration.

Axiom 5. All the information leaked through physical observations can be efficiently computed from a target device's internal configuration.

That is, given an algorithm and its physical implementation, the information leakage is a polynomial time computable function of (1) the computer's internal configuration (because of Axiom 4), (2) the chosen measurement (because of Axiom 3), and possibly (3) some randomness outside anybodys control (also because of Axiom 3).

We note that, from the practical point of view, these axioms may not reflect the entire physical phenomenons observed. For example, as far as Axiom 1 is concerned, volatile memories such as RAMs regularly require a small amount of energy to refresh their values and this could be used to mount a side-channel attack. However, such leakages are significantly more difficult to exploit than computational leakages. Our expectation is therefore that these axioms approximates the physical reality to a sufficient degree.

From these axioms, an *abstract computer* is defined as a collection of special Turing machines, which invoke each other as subroutines and share a special common memory. Each member of the collection is denoted as an *abstract virtual-memory Turing machine* (abstract VTM or simply VTM for short). One writes $\alpha := \{\alpha_1, \alpha_2, ..., \alpha_n\}$ to mean that an abstract computer α consists of abstract VTMs $\alpha_1, \alpha_2, ..., \alpha_n$. All VTM inputs and outputs are binary strings always residing in some virtual memory. Abstract computers and VTMs are not physical devices: they only represent logical computation and may have many different physical implementations.

Then, to model the physical leakage of any particular implementation, the notion of *physical VTM* is introduced. A physical VTM is a pair $(\mathcal{L}_i, \alpha_i)$, where α_i is an abstract VTM and $\mathcal{L}_i$ is a leakage function. If $\alpha := \{\alpha_1, \alpha_2, ..., \alpha_n\}$ is an abstract computer then $\varphi_i = (\mathcal{L}_i, \alpha_i)$ represents one physical implementation of α_i and $\varphi := \{\varphi_1, \varphi_2, ..., \varphi_n\}$ is defined as a physical implementation of the abstract computer α. It can also be denoted as the combination $\varphi = (\alpha, \mathcal{L})$ with $\mathcal{L} := \{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n\}$.

In these definitions, the relation between an abstract computing machine and a physical implementation is only determined by the leakage function that is qualitatively defined as a function of three inputs, $\mathcal{L}(C_\alpha, M, R)$:

- The first input is the current internal configuration C_α of an abstract computer α, which incorporates anything that is in principle measurable.
- The second input M is the setting of the measuring apparatus (in essence, a specification of what the adversary chooses to measure).
- The third input R is a random string to model the randomness of the measurement process, *e.g.* typically, R models the noise that affect the useful leakage signal.

In practice, one can also give a more quantitative view of a leakage function as follows. Let us imagine that a leaking device contains a secret k-bit value S. That is, S is a part of the computer's internal configuration C_α. Before any side-channel information has been leaked, any adversary would see S distributed according to a uniform distribution: $S \overset{R}{\longleftarrow} \{0,1\}^k$. By opposition, once a leakage has been obtained, the conditional distribution $\mathbf{P}[S|\mathcal{L}(S)]$ is not uniform anymore, meaning that all the secrets are not equally likely anymore. Otherwise said, the effect of a leakage function is to turn a uniform a-priori probability distribution into a non-uniform conditional probability distribution for some target secret signal S contained in the computer's internal configuration.

3 Target circuit

Our target cryptographic implementation is schematized in Figure 1.

It is defined as a combination of signals and operations. First, the set of all signals in the circuit is denoted as:

$$\Sigma := \{\sigma_1, \sigma_2, \sigma_3, ..., \sigma_s\},$$

where s is the total number of signals in the device. As physical signals are usually binary coded, we generally have $\sigma_i \in \mathbb{Z}_2$. In certain contexts, it may also be interesting to consider subsets of signals $\Theta_j := \{\sigma_l, \sigma_m, \sigma_n, ...\} \subset \Sigma$. In practice, the signal values are time-dependent and we have:

$$\Sigma(t) := \{\sigma_1(t), \sigma_2(t), \sigma_3(t), ..., \sigma_s(t)\}$$

Second, the cryptographic device can apply operations to the signals. A number of operations are actually included in the black box model. For example, if we consider block ciphers, a black box attacker could perform queries and obtain plaintext/ciphertext pairs. As a circuit could contain several such operations, we define the set of black box oracles B as the set of operations that one can query in the black box model:

$$B := \{\Omega_1, \Omega_2, \Omega_3, ..., \Omega_o\}$$

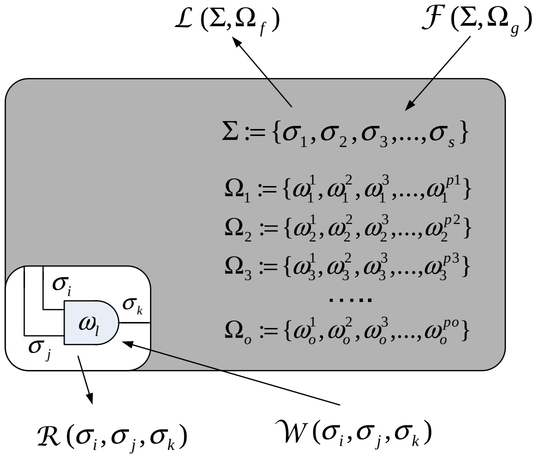
$\mathcal{L}(\Sigma,\Omega_f)$
$\mathcal{F}(\Sigma,\Omega_g)$
$\Sigma := \{\sigma_1,\sigma_2,\sigma_3,...,\sigma_s\}$
$\Omega_1 := \{\omega_1^1,\omega_1^2,\omega_1^3,...,\omega_1^{p1}\}$
$\Omega_2 := \{\omega_2^1,\omega_2^2,\omega_2^3,...,\omega_2^{p2}\}$
$\Omega_3 := \{\omega_3^1,\omega_3^2,\omega_3^3,...,\omega_3^{p3}\}$
.....
$\Omega_o := \{\omega_o^1,\omega_o^2,\omega_o^3,...,\omega_o^{po}\}$
σ_i
σ_k
ω_l
σ_j
$\mathcal{R}(\sigma_i,\sigma_j,\sigma_k)$
$\mathcal{W}(\sigma_i,\sigma_j,\sigma_k)$

Fig. 1. Circuit model including physical threats.

Then, in actual implementations, these oracles are made of several elementary operations that cannot be queried by the black box attacker (but possibly by the side-channel one) because they apply to the circuit inner signals. For every oracle Ω_i, we have:

$$\Omega_i := \{\omega_i^1, \omega_i^2, \omega_i^3, ..., \omega_i^{p_i}\}$$

We mention that we make no hypothesis about the actual form of the elementary operations ω_i^j's, although Figure 1 suggests that they represent logic gates. For clarity purposes, we represented our cryptographic implementation as a hardware circuit where every ω_i^j is physically implemented. However, in practice, different operations could be performed by the same hardware resource. This is typically the case in software-programmed processors and in this latter context, the ω_i^j's represent instructions applied sequentially to the signals rather than physical resources.

Based on these definitions, we can consider different types of physical opponents. For example, an invasive probing attack gives read/write access to a limited subset of signals in the device (*i.e.* the functions $\mathcal{R}$ and $\mathcal{W}$ in the figure) [4]. A fault attack applies some probabilistic function $\mathcal{F}$ to the signals or operations (it is probabilistic in the sense that a signal or operation is affected by the fault function with a certain probability) [7]. Finally, side-channel attacks enhance the opponent with a leakage function $\mathcal{L}$, *e.g.* [1, 24, 25]. In the following, we only consider these side-channel opponents.

It is important to observe that such a description can be efficiently translated into the formalism of [33]. Basically, our oracles Ω_i's can be simulated with abstract computers and the elementary operations ω_i^j's with VTMs. Also, our signals are simply the inputs and outputs of the VTMs. In the following, we will denote *abstract computers* as *cryptographic primitives* and *physical computers* as *implementations.* Note that in the cryptographic literature, a cryptographic primitive frequently denotes an idealized notion, *e.g.* a block cipher and the actual algorithms like the AES Rijndael [15] are rather considered as cryptographic primitives *instantiations.* Since in our physical context, only these practical instances are relevant, we denote them as primitives for short.

4 Target block cipher

A block cipher transforms a plaintext block P of a fixed bit length n_b into a ciphertext block C of the same length, under the influence of a cipher key K, of bit length n_k. We denote the forward operation of a block cipher as the encryption: $C = E_K(P)$ and the reverse operation as the decryption: $P = D_K(C)$.

In practice, modern block ciphers are usually composed of several identical transforms, denoted as the encryption (*resp.* decryption) rounds. If such a product cipher applies the same round function r times to the cipher state, it is necessary to expand the cipher key K into different round keys k_i. This is done by means of a key round. The round and key round functions are respectively denoted as:

$$\begin{aligned} p_{i+1} &= \mathrm{R}(p_i, k_i), \\ k_{j+1} &= \mathrm{KR}(k_j), \end{aligned}$$

where the p_i's represent the cipher state, with $p_0 = P$, $p_{r+1} = C$ and $k_0 = K$.

Finally, we model our round and key round functions as made of 3 different operations: a non-linear substitution layer, a linear diffusion layer and a bitwise XOR layer. Those are usual components of present block ciphers, *e.g.* the AES Rijndael.

More specifically, the substitution layer S consists of the parallel application of substitution boxes s to the b-bit blocks of the state:

$$\mathrm{S} : (\mathbb{Z}_{2^b})^{\frac{n_b}{b}} \rightarrow (\mathbb{Z}_{2^b})^{\frac{n_b}{b}} : x \rightarrow y = \mathrm{S}(x) \Leftrightarrow y^i = s\big(x^i\big), \quad 0 \leq i \leq \frac{n_b}{b} - 1,$$

where x^i is the ith b-bit block of the state vector x. The small S-boxes s are assumed to have good non-linearity, differential profile, non-linear order *etc.*

The linear diffusion layer D applies to the whole state and is assumed to have good diffusion properties (*e.g.* avalanche effect, high branch number, *etc.*):

$$\mathrm{D} : \mathbb{Z}_{2^{n_b}} \rightarrow \mathbb{Z}_{2^{n_b}} : x \rightarrow y = \mathrm{D}(x)$$

Finally, the bitwise XOR layer $\oplus$ is denoted as:

$$\oplus : \mathbb{Z}_2^{n_b} \times \mathbb{Z}_2^{n_b} \rightarrow \mathbb{Z}_2^{n_b} : x, y \rightarrow z \Leftrightarrow z(i) = x(i) \oplus y(i), \quad 0 \leq i \leq n_b - 1$$

where $x(i)$ is the ith bit of the state vector x. For example, a 3-round block cipher, is represented in Figure 2. With respect to the model of Section 3, the complete block cipher is an oracle E_K and a possible division in elementary operations would be $E_K := \{\mathrm{R}_1, \mathrm{R}_2, \mathrm{R}_3, \mathrm{KR}_1, \mathrm{KR}_2, \mathrm{KR}_3\}$. Another division (with smaller operations) is $E_K := \{\oplus_1, ..., \oplus_4, \mathrm{S}_\mathrm{A}, ..., \mathrm{S}_\mathrm{F}, \mathrm{D}_1, ..., \mathrm{D}_6\}$. As already mentioned, the choice of elementary operations is let open in our model and they can be as small as logic gates.

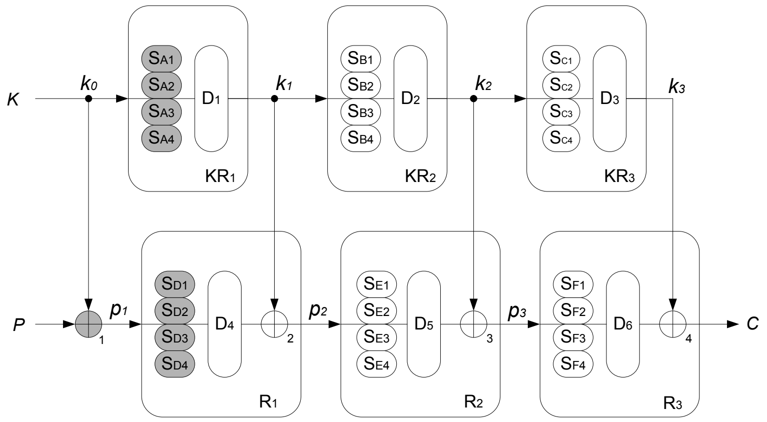

Fig. 2. A 3-round block cipher

5 Adversarial context and strategy

Before any formal security evaluation of a cryptographic primitive, it is important to clearly determine the adversarial context investigated. Similarly to black box attacks, side-channel attacks can consider the following adversaries:

1. non-adaptive, known plaintext side-channel adversaries (*na-kp-sca*),
2. non-adaptive, known ciphertext side-channel adversaries (*na-kc-sca*),
3. non-adaptive, chosen plaintext side-channel adversaries (*na-cp-sca*),
4. non-adaptive, chosen ciphertext side-channel adversaries (*na-cc-sca*),
5. adaptive chosen plaintext side-channel adversaries (*a-cp-sca*),
6. adaptive chosen ciphertext side-channel adversaries (*a-cc-sca*).

Importantly, a non-adaptive adversary is the one that can query its target cryptographic primitive (*e.g.* the block cipher of Figure 2) with an arbitrary number of plaintexts q and obtain the corresponding physical observations, but cannot choose its queries in function of the previously obtained observations. As a matter of fact, most of the presently investigated side-channel attacks are non-adaptive, *e.g.* the Differential Power Analysis (DPA for short) [25].

In addition to the adversarial context, we will consider the following adversarial strategies: "*given some physical observations and a resulting classification of key candidates, select the h best classified key(s)*[1]". That is, we have to chose between a hard decision (select only one key) or a soft decision (select a weighted list of h key candidates).

As will be detailed in Section 8, this adversarial strategy corresponds to the more formal notion of a key recovery attack. In the following of the paper, we aim to analyze the security of cryptographic implementations against side-channel key recovery.

[1] In the following of the paper, we will frequently refer to the secret signals in an implementation as the secret keys, or keys for short. However, any part of a computer's internal configuration could be the target of a side-channel attack.

6 Leakages predictions

In practice, most actual side-channel attacks perform in three steps:

1. According to some approximation of the leakage function, an adversary *predicts* (a part/function of) the key dependent leakages emanated from a target device.
2. It then *measures* the actual leakages from the target physical implementation.
3. It finally *compares* the actual leakages with the key dependent predictions. If the attack is successful, it is expected that the correct key candidate gives rise to the best leakage prediction which can be detected with a side-channel distinguisher.

In order to include these practical adversaries in our framework, it is therefore important to distinguish between the leakages, modelled by Micali and Reyzin as the measured side-channels of an actual device (*i.e.* a physical object) from their predictions with some approximation of the leakage function (*i.e.* a mathematical object). In the following, we will consequently use the leakage function $\mathcal{L}$ to describe the actual observations of a physical implementation and the prediction function $\mathcal{P}$ to describe the adversary's key dependent estimations. In this section, we first introduce univariate and multivariate leakage predictions. Then, we present a division of these predictions into three different categories (or adversarial contexts), namely non profiled, device profiled and key profiled leakage predictions. Finally, we formally define a side-channel adversary as the combination of a prediction function $\mathcal{P}$ and a distinguisher $\mathbf{C}$.

6.1 Univariate leakage predictions

Historically, the first side-channel attacks like the DPA were typically based on simple models, *e.g.* assuming some dependency between the Hamming weight of a value S computed in a physical device and its actual leakages. Such a context is illustratively depicted in the upper part of Figure 3. The figure shows a leakage trace corresponding to the serial execution of the block cipher's different operations: $\oplus$, S, D. In an univariate model, the adversary selects a number of points of interest (crossed) in the curve and tries to recover some information about the target secret signal from each of these points *independently.* It is interesting to observe that univariate predictions are frequently derived from the theoretical understanding of the target devices. For example, assuming power consumption dependencies related to the charge and discharge of certain capacitances in CMOS devices can explain Hamming weight data dependencies [40]. By opposition, multivariate models are best exploited with a statistical analysis of the target devices (that can possibly be combined with a-priori knowledge).

6.2 Multivariate leakage predictions

Multivariate leakage predictions correspond to more powerful attacks generally denoted as template attacks [10] and are illustrated in the lower part of Figure 3. The figure shows a leakage trace corresponding to the parallel (*i.e.* pipeline) execution of the block cipher's different operations. Because of the pipelined structure, different plaintexts are encrypted concurrently. As a matter of fact, a serious limitation of the univariate approach is that it neglects the dependencies between different leakage values and arbitrarily (or heuristically) selects the points of interest in the curve. The idea of multivariate predictions is to take these dependencies into account and build a leakage model that captures the correlations between different time instants.

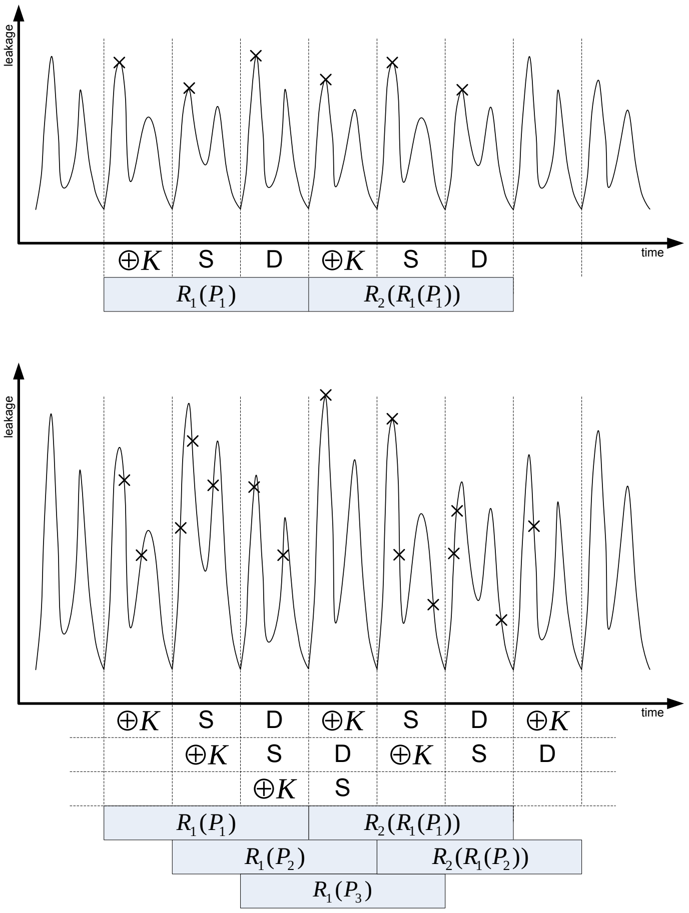

Fig. 3. Exemplary leakage traces of the 3-round cipher of Figure 2: (up) serial implementation, univariate leakage function, (down) pipeline implementation, multivariate leakage function.

For example, in Figure 3, one could use the 15 crosses in the pipeline implementation trace to characterize a target secret signal[2]. It is important to observe that exploiting multivariate predictions generally require a strong adversarial context. Indeed, since the leakage dependencies are particularly difficult to capture theoretically, they are usually exploited via a statistically estimated model which may be a limitation for practical adversaries. However, one can also take advantage of multivariate statistics without such statistical profiling, *e.g.* in the context of higher-order side-channel attacks [31, 50]. Note finally that, similarly to template attacks, stochastic models can be used for the construction of multivariate leakage predictions [41].

6.3 Categories of leakage predictions

The previous examples illustrate that univariate and multivariate leakage predictions can be seen as different approaches to take advantage of the same physical reality. Both generally try to extract some secret information from data-dependent physical observations. From an adversarial point of view, it is therefore important to include their specificity in the adversary definition. In particular, we distinguish:

1. Non profiling adversaries: do not require any statistical evaluation of the target device. They typically use simple univariate leakage predictions.
2. Unsupervised profiling adversaries: increase the quality of their leakage predictions due to some statistical evaluation of the target device's physical features, but the profiling step does not use secret key information.
3. Supervised profiling adversaries: additionally use secret key information during the profiling of the leakage prediction.

For short, we respectively denote the corresponding leakage predictions as non profiled, device profiled and key profiled. These contexts will be added to the adversary descriptions of Section 5 to allow a fair evaluation of side-channel attacks in Section 10.

6.4 Definition of a side-channel adversary

Let f_K be a cryptographic primitive (or abstract computer) embedding a secret key K with security parameter G, *i.e.* $K \in \{0, 1, 2, \ldots, G-1\}$[3]. Let $(f_K, \mathcal{L})$ be the physical computer (or implementation) corresponding to the association of the primitive f_K with a leakage function $\mathcal{L}$. We define a side-channel adversary $A^C_{f_K,\mathcal{L}}(\tau, q)$ with time complexity $\tau = \tau_p + \tau_c$ and $q = q_p + q_c$ queries to the implementation $(f_K, \mathcal{L})$ in a certain adversarial context C (that has to be selected from the lists in Sections 5, 6.3) as a combination of a prediction function $\mathcal{P}$ and a distinguisher $\mathbf{C}$.

The prediction function $\mathcal{P}$ of the side-channel adversary is a polynomial time function of the computer's internal configuration C_α that is used by the distinguisher to estimate the actual leakages of the target implementation. It may be profiled with time complexity τ_p and q_p queries to the implementation. The distinguisher of a side-channel adversary $\mathbf{C}$ is a polynomial time algorithm that performs an attack from the side-channel leakages modeled by the function $\mathcal{L}$ and the predictions modeled by the function $\mathcal{P}$, with time complexity τ_c and q_c queries to the target device.

[2] We note that the optimal selection of the points of interest in the curve is as challenging in multivariate models as it is for univariate ones and remains an open question [5].

[3] Typically, G is the guessing parameter of Section 7.1.

7 Computational restrictions in side-channel attacks

As mentioned in Section 2, a leakage function can be defined quantitatively as follows. Let S be a secret value contained in a computer's internal configuration. Before any side-channel information has been leaked, any adversary would see S distributed according to a uniform distribution: $S \xleftarrow{R} \{0,1\}^k$. A leakage function is any function such that the conditional distribution $\mathbf{P}[S|\mathcal{L}(S)]$ is not uniform anymore. From this definition, a general and well known observation is that, as far as block ciphers are concerned, obtaining a plaintext/ciphertext pair is already a very powerful leakage. Indeed, since for a given ciphertext, there is generally a one-to-one correspondence between the plaintext and the key, the knowledge of a plaintext/ciphertext pair is generally equivalent to the knowledge of the key, *from an information theoretic point of view* [44]. This simple example already suggests that, similarly to black box security, computational restrictions have to be imposed to side-channel adversaries in order to capture the reality of physical implementations.

7.1 Limitations of the memory complexity

As most cryptanalytic techniques, side-channel attacks are based on a divide-and-conquer strategy in which different (computationally tractable) parts of a secret key are recovered separately. Looking at the generic attack description of Section 6, it involves that the leakage predictions should only be dependent on a reasonable number of key bits. Otherwise said, the only signals for which the side-channel information is exploitable are those for which the corresponding key-dependent predictions can be stored in memory. For this purpose, we introduce the notion of M-limited adversaries, *i.e.* adversaries that are able to store the predictions of up to M candidates for the target secret signal in a leaking implementation.

For example, looking at the block cipher in Figure 2, assuming 16-bit S-boxes, a 64-bit diffusion layer, and a known plaintext adversary, it is clear that the signals before the first diffusion layer (*i.e.* corresponding to the grey boxes) can be enumerated by a 2^{16}-limited adversary. By opposition, the signals after the diffusion layer cannot be enumerated anymore, excepted by a (potentially unrealistic) 2^{64} or more limited adversary [45]. Importantly, this does *not* mean that the leakages corresponding to operations after the diffusion layer are not useful. Indeed, there can be dependencies between operations corresponding to these leakages and enumerable signals.

While the previous computational assumption is important to determine the limits of what an adversary can achieve, it is not sufficient to compare side-channel attacks. For example, an adversary could be 2^{16}-limited, but decide (for some practical reasons) to target only 8 bits of a secret signal. For this purpose, we additionally define a G-guessing adversary as an adversary that uses its side-channel leakages to identify a secret signal that was a priori contained in a set of G uniform candidates.

7.2 Limitations of the time and data complexity

The time and data complexity are integrated in the adversary definition of Section 6.4 and in the following definition of security against side-channel key recovery.

8 Definition of security against side-channel attacks

As mentioned in Section 5, the adversarial strategy that we consider in this paper corresponds to a side-channel key recovery attack that can be formalized as follows.

Let $A^{C}_{f_K,\mathcal{L}}(\tau, q)$ be the previously defined side-channel adversary against an implementation $(f_K, \mathcal{L})$. We consider the following experiment:

```
Experiment $\mathbf{Exp}^{\text{sc-kr}}_{f_K,\mathcal{L}}$
$K \xleftarrow{R} \{0, 1, 2, \dots, G-1\}$;
$K^* \leftarrow A^{C}_{f_K,\mathcal{L}}(\tau, q)$;
if $K = K^*$ then return 1;
               else return 0;
```

The key recovery advantage of the adversary $A^{C}_{f_K,\mathcal{L}}(t, q)$ is defined as:

$$\mathbf{Adv}^{\text{sc-kr}}_{A}(\tau, q) = \mathbf{P}\ [\mathbf{Exp}^{\text{sc-kr}}_{f_K,\mathcal{L}} = 1] \tag{1}$$

For any τ, q, we finally define the key recovery advantage of a cryptographic implementation $(f_K, \mathcal{L})$ against side-channel adversaries in a given adversarial context as:

$$\mathbf{Adv}^{\text{sc-kr}}_{f_K,\mathcal{L}}(\tau, q) = \max_{A}\{\mathbf{Adv}^{\text{sc-kr}}_{A}(\tau, q)\} \tag{2}$$

Note that the key recovery can be straightforwardly obtained with the hard strategy of Section 5 (*i.e.* select only the best classified key) and therefore require no additional time complexity than the one of the distinguisher. Similarly, a soft strategy can lead to key recovery by combining the side-channel attack with some additional offline black box computations (*e.g.* testing the remaining candidates by executing the algorithm).

9 Evaluation criteria for side-channel attacks

In our theoretical framework, a side-channel attack and its evaluation would typically take place as as illustrated in Figure 4. On the measurement side, a target device containing a secret S_g and running a cryptographic primitive (*e.g.* a block cipher) is queried with q inputs $P_1, P_2, \dots, P_q$ and monitored, depending on an adversarial context that has to be clearly defined. On the evaluation side, an adversarial strategy has turned the leakages into a selection for the target secret signal[4]. From this description, there are two natural questions to face in order to quantify the power and efficiency of such a physical adversary, namely:

1. What is the amount of information provided by the given leakage function?
2. How successfully has the adversary turned this information into a practical attack?

For this purpose, this section describes how to evaluate the effect of physical leakages with a combination of security and information theoretic measurements.

[4] Note that the useful leakages do not mandatorily directly depend on S_g. For example, in block ciphers implementations, it is usual to exploit the leakages at the output of the S-box layer for different plaintexts, *e.g.* $\mathcal{L}(Y_i = \mathrm{S}(P_i \oplus S_g))$.

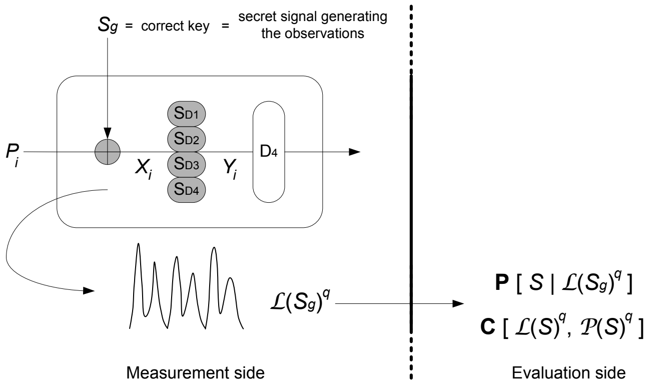

Fig. 4. Evaluation of a side-channel attack.

9.1 Security measurement: the adversary's average success rate

The most natural way to assess the performances of a side-channel attack is to evaluate the probability of success of the adversary specified in Section 5. This metric is in fact directly inspired from the "zero/one loss" metric that is generally used in statistical machine learning to characterize Bayesian classification schemes [22]. As a matter of fact, the side-channel key recovery defined in Section 8 fundamentally corresponds to a classification problem. More specifically, if we consider an adversary of which the strategy is to select the best classified key *only* (*i.e.* with $h = 1$, see Section 5), the success rate can be written as follows.

Let S be a random variable in the discrete domain $\mathcal{S}$ denoting the target secret signals in a side-channel attack. Let $L^q_{S_g} = \mathcal{L}(S_g)^q$ and $P^q_S = \mathcal{P}(S)^q$ be two random variables respectively denoting the side-channel observations generated by a secret signal S_g with q queries to the target device and the adversary's leakage predictions. Let finally $\mathbf{C}(L^q_{S_g}, P^q_S)$ be the distinguisher used by the adversary to compare the actual side-channel observations with its guessed predictions. This distinguisher could be a difference of mean test [25], a correlation test [6], a Bayesian classification [10], or any other tool, possibly inspired from classical cryptanalysis, *e.g.* [8, 34, 42]. For each observation $L^q_{S_g}$, we define the set of keys selected by the adversary as:

$$M^q_{S_g} = \{\hat{s} \mid \hat{s} = \underset{S}{argmax}\, \mathbf{C}(L^q_{S_g}, P^q_S)\},$$

and the result of the attack with the index matrix:

$$\mathrm{I}^q_{S_g,S} = \qquad \tfrac{1}{|M^q_{S_g}|} \textbf{ if } S \in M^q_{S_g}, \qquad \textbf{else } 0$$

Then, we define the success rate of the adversary after q queries for a secret signal S_g:

$$\mathbf{S_R}(S_g, q) = \underset{L^q_{S_g}}{\mathbf{E}} \mathrm{I}^q_{S_g,S_g}, \tag{3}$$

and the average success rate of the adversary after q queries:

$$\overline{\mathbf{S_R}}(q) = \underset{S_g}{\mathbf{E}} \; \underset{L^q_{S_g}}{\mathbf{E}} \mathrm{I}^q_{S_g,S_g} \tag{4}$$

Note that the average success rate obviously not only depends on the number of queries q but also on the adversary's time complexity τ that is omitted in the formulas for clarity reasons. As a matter of fact, we have: $\overline{\mathbf{S_R}}(\tau, q) = \mathbf{Adv}_A^{\text{sc-kr}}(\tau, q)$.

Finally, we remark that one can use the complete index matrix to build a confusion matrix $\Gamma^q_{S_g,S} = \mathbf{E}_{L^q_{S_g}} \ \mathrm{I}^q_{S_g,S}$. The previously defined average success rate simply corresponds to the averaged diagonal of this matrix.

9.2 Information theoretic measurement: the conditional entropy

The previous security metric gives an indication about how successfully a side-channel adversary with a simple "select the most likely signal" strategy can guess a secret key. It could be easily extended to soft strategies (where a list of candidates are selected) by modifying the index matrix. In this section, we define an information theoretic metric (namely the conditional entropy) to combine with the average success rate for the analysis of side-channel attacks. It can be written as follows.

Let $\mathbf{P}[S|L^q_{S_g}]$ be the probability vector of the different key candidates S given a leakage $L^q_{S_g}$ generated by a correct key S_g after q queries to the target device. Similarly to the confusion matrix of the previous section, we define a conditional entropy matrix $\mathrm{H}^q_{S_g,S} = \mathbf{E}_{L^q_{S_g}} - \log_2 \mathbf{P}[S|L^q_{S_g}]$. Then, we define the conditional entropy:

$$\mathbf{H}[S_g|L^q_{S_g}] = \mathop{\mathbf{E}}_{S_g} \ \mathrm{H}^q_{S_g,S_g} \tag{5}$$

We note that this definition is equivalent to Shannon's definition [12][5]. We used the previous notation because it is convenient to consider the conditional entropy matrix.

Finally, we define the leakage matrix corresponding to the entropy reductions: $\Lambda^q_{S_g,S} = \mathbf{H}[S_g] - \mathrm{H}^q_{S_g,S}$, where $\mathbf{H}[S_g]$ is the entropy of the secret values before any side-channel attack has been performed. It directly yields the mutual information:

$$\mathbf{I}(S_g; L^q_{S_g}) = \mathbf{H}[S_g] - \mathbf{H}[S_g|L^q_{S_g}] = \mathop{\mathbf{E}}_{S_g} \Lambda^q_{S_g,S_g} \tag{6}$$

Note that because of its physical nature, a leakage function cannot be easily expressed as an analytical expression and therefore has to be approximated with the adversary's predictions. In practice, a good evaluation of the information leakages is provided by the use of templates or stochastic models, although it still relies on some hypotheses. The question of how to best extract the information of a leaking device, *i.e.* how to best approximate a leakage function is still open [5, 10, 41].

[5] Since:
$$\begin{aligned}\mathbf{H}[S_g|L^q_S] &= \textstyle\sum_{L^q_S} \mathbf{P}[L^q_S] \sum_{S_g} \mathbf{P}[S_g|L^q_S] \cdot - \log_2(\mathbf{P}[S_g|L^q_S]) \\ &= \textstyle\sum_{L^q_S} \mathbf{P}[L^q_S] \sum_{S_g} \frac{\mathbf{P}[L^q_S|S_g] \cdot \mathbf{P}[S_g]}{\mathbf{P}[L^q_S]} \cdot - \log_2(\mathbf{P}[S_g|L^q_S]) \\ &= \textstyle\sum_{L^q_S} \sum_{S_g} \mathbf{P}[L^q_S|S_g] \cdot \mathbf{P}[S_g] \cdot - \log_2(\mathbf{P}[S_g|L^q_S]) \\ &= \textstyle\sum_{S_g} \sum_{L^q_S} \mathbf{P}[L^q_S|S_g] \cdot \mathbf{P}[S_g] \cdot - \log_2(\mathbf{P}[S_g|L^q_S]) \\ &= \textstyle\sum_{S_g} \mathbf{P}[S_g] \sum_{L^q_S} \mathbf{P}[L^q_S|S_g] \cdot - \log_2(\mathbf{P}[S_g|L^q_S]) = \mathbf{E}_{S_g} \ \mathrm{H}^q_{S_g,S_g}\end{aligned}$$

9.3 Combining security and information theoretic measurements

From the previous definitions, it is important to observe that the average success rate fundamentally describes an adversary and generally has to be computed for different number of queries q in order to determine how much observations are required to perform a successful key recovery. By contrast, the information theoretic metric characterizes a physical computer and is generally evaluated only once, for an arbitrarily chosen number of queries, in order to determine if there is *enough* information to mount an attack and to quantify this information. With this respect, the mutual information is particularly interesting since it allows to easily detect *sound leakage functions*, *i.e.* leakage functions such that the maximum leakage corresponds to the correct key.

Definition: A leakage function is sound if and only if $\max_S \Lambda^q_{S_g,S} = \Lambda^q_{S_g,S_g}$, $\forall\ S_g, q$.

If enhanced with a sound leakage function, an adversary allowed to perform unlimited queries to a target device will recover the correct key with a Bayesian classification, since the product of all probabilities $\mathbf{P}[S|L^q_{S_g}]$ will be maximum for the correct key.

The intuition behind the proposed evaluation criteria for side-channel attacks is summarized in Figure 5. As already mentioned, security and information theoretic measurements provide different aspects of a side-channel attack's efficiency. Namely, the mutual information measures the average amount of information that is available in the observations while the average success rate measures how efficiently an actual adversarial strategy can turn this information into a successful key recovery.

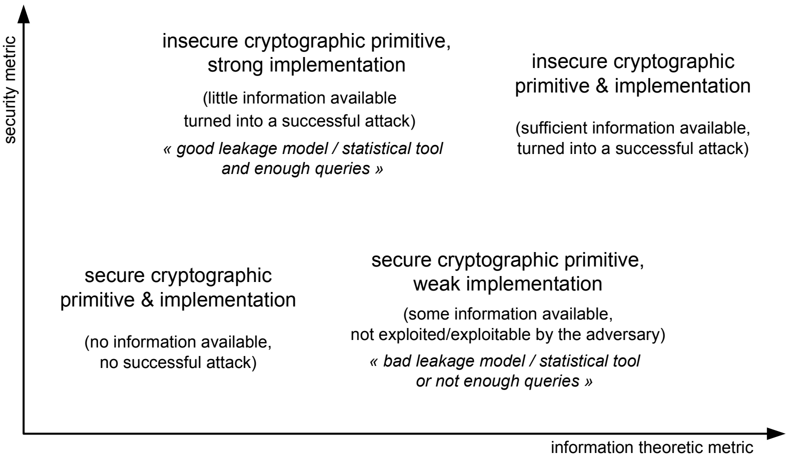
security metric
insecure cryptographic primitive, strong implementation
(little information available turned into a successful attack)
« good leakage model / statistical tool and enough queries »
insecure cryptographic primitive & implementation
(sufficient information available, turned into a successful attack)
secure cryptographic primitive & implementation
(no information available, no successful attack)
secure cryptographic primitive, weak implementation
(some information available, not exploited/exploitable by the adversary)
« bad leakage model / statistical tool or not enough queries »
information theoretic metric

Fig. 5. Summary of side-channel evaluation criteria.

By combining both measurements, one can analyze the security of a cryptographic primitive as well as the quality of its implementation. For illustration purposes, we divided the possible results of a side-channel attack in four intuitive categories (illustrated in Figure 5). In practice, however, most of the presently investigated implementations fall into the "insecure cryptographic primitive and implementation" category and their respective efficiency has to be quantified numerically. Note that actual implementations may differ by their leakage functions, *e.g.* smart cards and FPGAs have different leakage models. They may also differ by the way the abstract computer is divided into different elementary operations or VTMs, *e.g.* within the same circuit technology, one could implement different AES Rijndael cores: loop, unrolled, masked,...

Importantly, these evaluation criteria can be applied to any leakage prediction function/distinguisher, including univariate and multivariate ones.

9.4 Comparison with black box security

In this section, we briefly illustrate the need of combined metrics for the evaluation of side-channel attacks by comparing them with black box attacks.

Let us for example consider side-channel attacks and linear cryptanalysis [29]. Looking at their similarities, one can first observe that both attacks basically include the same steps. That is, both adversaries are provided with some information (*e.g.* the black box or side-channel queries to the primitive and its implementation) and then try to exploit this information with some distinguisher. Secondly, in both contexts different distinguishers give rise to different efficiencies. For example, in linear cryptanalysis, one can use Matsui's simple counter strategy or more optimal distinguishers (*e.g.* [23]). Similarly, side-channel attacks can exploit the leakages with difference of mean tests, correlation analysis, Bayesian classification, ...

As a consequence, one could evaluate both the linear cryptanalysis and side-channel attacks with a combination of security and information theoretic metrics. The difference is that, as far as black box attacks are concerned, the number of queries is already a satisfactory measurement of the amount of information obtained by the adversary. Otherwise said, the relevant information about a black box attack is mainly computational (measured with the adversary's advantage). By contrast, in side-channel attacks, for a similar number of queries q, the amount of information obtained by the adversary can vary for different implementations. Therefore, both the security and the information leakages of an implementation have to be measured carefully in this context.

10 Evaluation methodology

Figure 6 summarizes our evaluation methodology for side-channel attacks. It holds in five steps that we detail carefully in this section.

1. We define the target implementation as modeled by Micali and Reyzin. That is, we define the combination of an abstract computer and a leakage function. In practice, the target implementation is a physical object, *e.g.* a smart card, FPGA or ASIC running some cryptographic primitive.
2. We define the target secret S_g for the side-channel attack. It involves a specification of the guessing parameter G, defined in Section 7.1.
3. Once the target has been specified, we answer the first question in our evaluation, namely: "*What is the amount of information contained in the physical observations obtained from a leaking device?*". For this purpose, we use the mutual information.

As already mentioned in Section 9.2, we are theoretically interested in the information content of the leakage function but in practice, it can only be approximated with a good prediction function, *e.g.* based on template-like techniques.

4. Before the security evaluation of the cryptographic primitive, we define the adversary, including the adversarial context and strategy as well as a side-channel distinguisher and a prediction function.
5. We finally answer the second question in our evaluation, namely: "*How successfully can an adversary turn this information into a practical attack?*". For this purpose, we use the advantage of the side-channel key recovery adversary of Section 9.1.

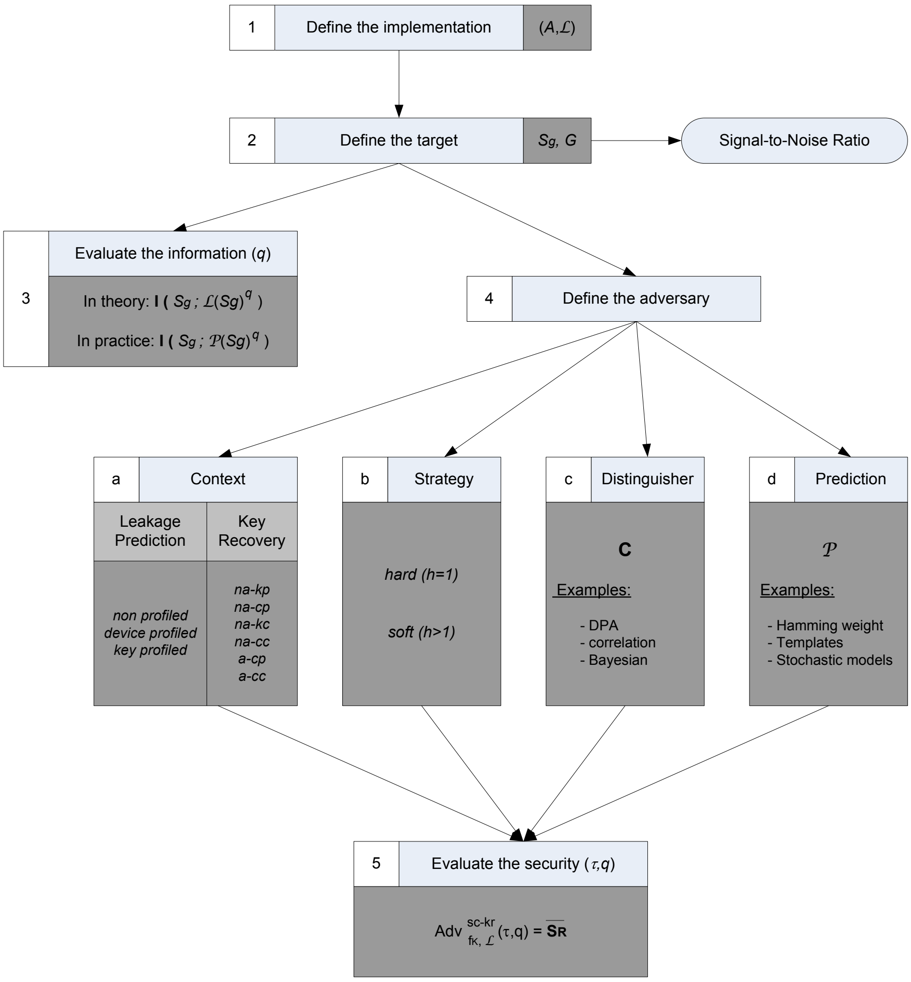

Fig. 6. Evaluation methodology for side-channel attacks.

Figure 6 typically indicates that the information theoretic metric can be used to discriminate different implementations, while the security metric is rather useful to discriminate different adversaries, for a given implementation.

Additionally to these evaluation criteria, it is often interesting to define a Signal-to-Noise Ratio (SNR) in order to determine the *fraction of useful signal in the side-channel observations*. For example, an SNR was defined in [26] as the ratio between the leakage (*e.g.* the power consumption) caused by the attacked intermediate result S in an implementation and the additive noise. Although such an SNR is independent of the previously discussed information and security issues (see the example in Appendix A), it can be used to plot the information theoretic and security evaluation metrics with its respect. Note that the useful signal may be difficult to define, *e.g.* for multivariate leakage predictions and in the latter case, it can simply be normalized so that the SNR remains useful to determine the amount of noise in the observations.

Another important remark is that, as any statistical evaluation of security, the relevance of the previous investigations depends on the variance of the estimated parameters. High variances over the leakages typically indicate a possibility to take advantage of an adaptive context, by forcing worst case leakages. High variances over the secret signals rather indicate that certain keys are more difficult to identify than others.

Some previous metrics for analyzing side-channel attacks are discussed in appendix B.

11 Side-channel attacks tradeoffs

From a theoretical point of view, the previously introduced model finally allows the discussion of the fundamental tradeoffs a side-channel adversary has to face, namely "*flexibility vs. efficiency*" and "*information vs. computation*".

The flexibility *vs.* efficiency tradeoff typically relates to the adversarial context considered. As a matter of fact, an adaptive adversary using a key-profiled leakage prediction function will generally recover (much) more information from side-channel measurements than a non-adaptive one, using a non profiled leakage prediction. However, simpler models do not only involve a sub-optimal information extraction from side-channel traces. They may also be more easily reproducible to different devices. As a typical example, a DPA only assumes that somewhere in a physical observation, the leakage will depend on a single bit value. The simplicity of this assumption made it straightforwardly applicable to a wide range of devices, without any adaptation. Correlation attacks [6], template attacks [10] multi-channel attacks [2] or stochastic models [41] are trading some of this flexibility for a more efficient information extraction.

The information *vs.* computation tradeoff rather relates to the adversarial strategy considered. As a matter of fact, for comparable amounts of side-channel queries q, a soft strategy trying to extract a list of key candidates including the correct one will generally have a higher success rate than a hard strategy, trying to extract the correct key value only. However, if this list of candidates can be tested with some computational power, it can be turned into a successful key recovery. That is, a lack of information can be overcome by an more computationally intensive adversarial strategy.

12 Conclusions and open problems

A formal practice-oriented model for the analysis of cryptographic primitives against side-channel attacks is introduced as a specialization of Micali and Reyzin's "physically observable cryptography" paradigm [33]. It is based on a theoretical framework in which the effect of practically relevant leakage functions is evaluated with a combination of security and information theoretic measurements. The model allows, both, the practical comparison of actual side-channel attacks and the analysis and understanding of the underlying mechanisms in physically observable cryptography.

Open problems include the evaluation of actual side-channel attacks and countermeasures within the model in different implementation contexts, in particular those for which the security evaluation remains an open question, *e.g.* dual rail pre-charge logic styles. The design of cryptographic primitives with provable security against side-channel key recovery is another scope for research. Importantly, proving the security of an implementation would require to consider the side-channel advantage of this implementation over all possible adversaries (including the side-channel distinguisher and the prediction function). It leads to the additional following practical open questions: "*What is the best way to approximate a leakage function?*" and "*how to best exploit it*?" [5, 10, 19, 41]. Finally, the study of stronger security notions than side-channel key recovery (*e.g.* indistinguishability) and the extension to other physical adversaries (*e.g.* fault-based) are a third direction for further investigations.

Acknowledgements: The authors would like to acknowledge Leonid Reyzin for his valuable contributions to the development and understanding of the issues discussed in the paper. We also would like to thank Cédric Archambeau, Nenad Dedic, Marc Joye, François Koeune, Stefan Mangard, Elisabeth Oswald, Eric Peeters, Christophe Petit, Gilles Piret, Emmanuel Prouff and our anonymous reviewers for helpful comments and discussions about preliminary versions of this work.

References

1. D. Agrawal, B. Archambeault, J. Rao, P. Rohatgi, *The EM Side-Channel(s)*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 29-45, Redwood City, California, USA, August 2002.
2. D. Agrawal, J. Rao, P. Rohatgi, *Multi-channel Attacks*, in the proceedings of CHES 2003, Lecture Notes in Computer Science, vol 2779, pp 2-16, Cologne, Germany, Sept. 2003.
3. M.L. Akkar, C. Giraud, *An Implementation of DES and AES Secure againts Some Attacks*, in the proceedings of CHES 2001, Lecture Notes in Computer Science, vol 2162, pp 309-318, Paris, France, May 2001.
4. R. Anderson, M. Kuhn, *Tamper Resistance - a Cautionary Note*, in the proceedings of the USENIX Workshop on Electronic Commerce, pp 1-11, Oakland, USA, Nov. 1996.
5. C. Archambeau, E. Peeters, F.-X. Standaert, J.-J. Quisquater, *Template Attacks in Principal Subspaces*, in the proceedings of CHES 2006, Lecture Notes in Computer Science, vol 4249, pp. 1–14, Yokohama, Japan, October 2006.
6. E. Brier, C. Clavier, F. Olivier, *Correlation Power Analysis with a Leakage Model*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 16-29, Boston, Massachusetts, USA, August 2004.
7. D. Boneh, R. DeMillo, R. Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*, proceedings of Eurocrypt 1997, Lecture Notes in Computer Science, vol 1233, pp 37-51, Konstanz, Germany, May 1997.
8. V. Carlier, H. Chabanne, E. Dottax, H. Pelletier, *Electromagnetic Side Channels of an FPGA Implementation of AES*, Cryptology ePrint Archive, Report 2004/145, 2004, http://eprint.iacr.org/.
9. S. Chari C. Jutla, J. Rao, P. Rohatgi, *Towards Sound Approaches to Counteract Power-Analysis Attacks*, in the proceedings of Crypto 1999, Lecture Notes in Computer Science, vol 1666, pp 398-412, Santa Barbara, California, USA, August 1999.
10. S. Chari, J. Rao, P. Rohatgi, *Template Attacks*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 13-28, Redwood City, CA, USA, August 2002.
11. J.S. Coron, P. Kocher, D. Naccache, *Statistics and Secret Leakage*, in the proceedings of Financial Crypto 2000, Lecture Notes in Computer Science, vol 1972, pp 157-173, Anguilla, British West Indies, February 2000.
12. T.M. Cover, J.A. Thomas, *Information Theory*, Wiley and Sons, New York, 1991.
13. C. Clavier, J.S. Coron, N. Dabbous, *Differential Power Analysis in the Presence of Hardware Countermeasures*, in the proceedings of CHES 2000, Lecture Notes in Computer Science, vol 1965, pp 252-263, Worcester, Massachusetts, USA, August 2000.
14. J.-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestre, J.-J. Quisquater, *A Practical Implementation of the Timing Attack*, in the proceedings of CARDIS 1998, Lecture Notes in Computer Science, vol 1820, pp 167-182, Louvain-la-Neuve, Belgium, 1998.
15. FIPS 197, *"Advanced Encryption Standard,"* Federal Information Processing Standard, NIST, U.S. Dept. of Commerce, November 26, 2001.
16. W. Fischer, B.M. Gammel, *Masking at Gate Level in the Presence of Glitches*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 187-200, Edinburgh, Scotland, September 2005.
17. J.A. Fournier, S. Moore, H. Li, R.D. Mullins, G.S. Taylor, *Security Evaluation of Asynchronous Circuits*, in the proceedings of CHES 2003, Lecture Notes in Computer Science, vol 2779, pp 137-151, Cologne, Germany, September 2003.
18. K. Gandolfi, C. Mourtel, F. Olivier, *Electromagnetic Analysis: Concrete Results*, in the proceedings of CHES 2001, Lecture Notes in Computer Science, vol 2162, pp 251-261, Paris, France, May 2001.
19. B. Gierlichs, K. Lemke, C. Paar, *Templates vs. Stochastic Methods*, in the proceedings of CHES 2006, Lecture Notes in Computer Science, vol 4249, pp 15-29, Yokohama, Japan, October 2006.

20. L. Goubin, J. Patarin, *DES and Differential Power Analysis*, in the proceedings of CHES 1999, Lecture Notes in Computer Science, vol 1717, pp 158-172, Worcester, Massachussets, USA, August 1999.
21. S. Guilley, P. Hoogvorst, R. Pacalet, *Differential Power Analysis Model and Some Results*, in the proceedings of CARDIS 2004, pp 127-142, Toulouse, France, Kluwer 2004.
22. T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, second edition, Springer Verlag, New York, 2001.
23. P. Junod, *On the Optimality of Linear, Differential, and Sequential Distinguishers*, in the proceedings of Eurocrypt 2003, Lecture Notes in Computer Science, vol 2656, pp 17-32, Warsaw, Poland, May 2003.
24. P. Kocher, *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems*, in the proceedings of Crypto 1996, Lecture Notes in Computer Science, vol 1109, pp 104-113, Santa Barbara, California, USA, August 1996.
25. P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*, in the proceedings of Crypto 1999, Lecture Notes in Computer Science, vol 1666, pp 398-412, Santa-Barbara, California, USA, August 1999.
26. S. Mangard, *Hardware Countermeasures against DPA - A Statistical Analysis of Their Effectiveness*, in the proceedings of CT-RSA 2004, Lecture Notes in Computer Science, vol 2964, pp 222-235, San Francisco, California, USA, February 2004.
27. S. Mangard, T. Popp, B.M. Gammel, *Side-Channel Leakage of Masked CMOS Gates*, in the proceedings of CT-RSA 2005, Lecture Notes in Computer Science, vol 3376, pp 351-365, San Francisco, California, USA, February 2004.
28. S. Mangard, N. Pramstaller, E. Oswald, *Successfully Attacking Masked AES Hardware Implementations*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 157-171, Edinburgh, Scotland, September 2005.
29. M. Matsui, Linear Cryptoanalysis Method for DES Cipher, in the proceedings of Eurocrypt 1992, Lecture Notes in Computer Science, vol 658, pp 81-91, Balatonfured, Hungary, May 1992.
30. D. May, H. Muller, N. Smart, *Randomized Register Renaming to Foil DPA*, in the proceedings of CHES 2001, Lecture Notes in Computer Science, vol 2162, pp 28-38, Paris, France, May 2001, Springer-Verlag.
31. T.S. Messerges, *Using Second-Order Power Analysis to Attack DPA Resistant Software*, in the proceedings of CHES 2000, Lecture Notes in Computer Science, vol 1965, pp 71-77, Worcester, Massachusetts, USA, August 2000.
32. T.S. Messerges, E.A. Dabbish, R.H. Sloan, *Examining Smart-Card Security under the Threat of Power Analysis Attacks*, IEEE Transactions on Computers, vol 51, num 5, pp 541-552, May 2002.
33. S. Micali, L. Reyzin, *Physically Observable Cryptography*, in the proceedings of TCC 2004, Lecture Notes in Computer Science, vol 2951, pp 278-296, Cambridge, Massachusetts, USA, February 2004.
34. F. Muller, *Analyse d'Algorithmes en Cryptographie Symétrique*, PhD Thesis, Sept. 2005.
35. E. Oswald, S. Mangard, N. Pramstaller, V. Rijmen, *A Side-Channel Analysis Resistant Description of the AES S-Box*, in the proceedings of FSE 2005, Lecture Notes in Computer Science, vol 3557, pp 413-423, Paris, France, February 2005.
36. E. Peeters, F.-X. Standaert, N. Donckers, J.-J. Quisquater, *Improved Higher-Order Side-Channel Attacks With FPGA Experiments*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 309-323, Edinburgh, Scotland, September 2005.
37. E. Peeters, F.-X. Standaert, J.-J. Quisquater, *Power and Electromagnetic Analysis: Improved Model, Consequences and Comparisons*, in Integration, the VLSI Journal, vol 40, pp 52-60, Spring 2007, Elsevier.
38. T. Popp, S. Mangard, *Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 172-186, Edinburgh, Scotland, September 2005.

39. J.-J. Quisquater, D. Samyde, *ElectroMagnetic Analysis (EMA): Measures and Couter-Measures for Smard Cards*, in the proceedings of E-smart 2001, Lecture Notes in Computer Science, vol 2140, pp 200-210, Cannes, France, September 2001.
40. J.M. Rabaey, *Digital Integrated Circuits*, Prentice Hall International, 1996.
41. W. Schindler, K. Lemke, C. Paar, *A Stochastic Model for Differential Side-Channel Cryptanalysis*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 30-46, Edinburgh, Scotland, September 2005.
42. K. Schramm, G. Leander, P. Felke, C. Paar, *A Collision Attack on AES: Combining Side Channel and Differential Attack*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 163-175, Boston, USA, August 2004.
43. A. Shamir, *Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies*, in the proceedings of CHES 2000, Lecture Notes in Computer Science, vol 1965, pp 238-251, Worcester, Massachusetts, USA, August 2000.
44. C. E. Shannon, *Communication theory of secrecy systems*, Bell System Technical Journal, vol 28, pp 656-715, October 1949.
45. F.-X. Standaert, S.B. Ors, B. Preneel, *Power Analysis of an FPGA Implementation of Rijndael: is Pipelining a DPA Countermeasure?*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 30-44, Boston, USA, August 2004.
46. F.-X. Standaert, E. Peeters, G. Rouvroy, J.-J. Quisquater, *Power Analysis Attacks Against Field Programmable Gate Arrays*, in the Proceedings of the IEEE, vol. 94, num. 2, pp 383-394, February 2006.
47. F.-X. Standaert, E. Peeters, C. Archambeau, J.-J. Quisquater, *Towards Security Limits in Side-Channel Attacks*, in the proceedings of CHES 2006, Lecture Notes in Computer Science, vol 4249, pp. 30–45, Yokohama, Japan, October 2006.
48. K. Tiri, M. Akmal, I. Verbauwhede, *A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards*, in the proceedings of ESSCIRC 2003.
49. K. Tiri, I. Verbauwhede, *Design Method for Constant Power Consumption of Differential Logic Circuits*, in the proceedings of DATE 2005, IEEE Computer Society , pp 628-633, Munich, Germany, March 2005.
50. J. Waddle, D. Wagner, *Towards Efficient Second-Order Power Analysis*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 1-15, Boston, Massachusetts, USA, August 2004.

A Limitations of the Signal to Noise Ratio

The aim of the signal to noise ratio is to determine the fraction of useful signal in an implementation, no matter if it contains information. For example, an SNR was defined in [26] as the ratio between the leakage (*e.g.* the power consumption) caused by the attacked intermediate result S in an implementation and the additive noise N. It was initially introduced to measure the efficiency of side-channel attacks using the correlation coefficient. Since DC components are not relevant for the computation of this coefficient, only the variance of the signals were considered in the definition:

$$\mathrm{SNR} = \frac{\sigma^2(\mathcal{L}(S))}{\sigma^2(N)} \tag{7}$$

We illustrate this definition with the left implementation of Figure 7, in a Hamming weight leakage model. For simplicity, we assume that only the values outside the grey box are leaking. The figure illustrates a context where an adversary targets a b-bit S-box that is affected by $3b$ random bits of noise. It typically corresponds to a side-channel attack against a block cipher where the adversary targets one S-box out of four, *e.g.* as in [46]. Consequently, the outputs of the un-targeted S-boxes produce what is usually referred to as algorithmic noise, approximated by the random bits r_1, r_2, r_3. Since we consider a Hamming weight model, the variances of the leakages are easily calculated. Namely the mean Hamming weigh of an n-bit random value is $n/2$ and its variance $n/4$. Therefore, the SNR of the example in Figure 7 is worth $\frac{b/4}{3b/4} = \frac{1}{3}$.

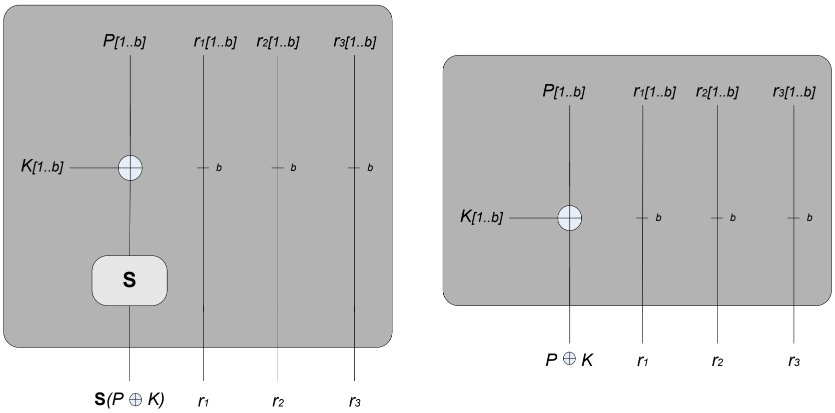

Fig. 7. Illustrative implementation with SNR=1/3.

We can easily observe that this SNR is not an information theoretic metric nor a security metric in itself with the following example. Let us consider the right scheme of Figure 7 in the Hamming distance leakage model. Clearly, the SNR of this example is again $\frac{1}{3}$ while it does not leak any key information. Indeed $W_H(P_1 \oplus K \oplus P_2 \oplus K)$ does not depend on the key. In general, the implementation SNR is independent of the leakage function and statistical tool used by the adversary.

B Discussion of some previous evaluation metrics

This appendix briefly discusses the relevance of our introduced metrics with respect to some commonly accepted solutions for the analysis of side-channel attacks. Looking at the following discussion, an important observation is that these tools (1) generally fail to allow a unifying view of all side-channel attacks; (2) usually neglect the information issues and mainly focus on security. This does not mean that such metrics are not meaningful in the context in which they were introduced but justifies the need of new evaluation criteria, as the following examples underline.

SNR + correlation coefficient: in [26], it is suggested to relate the previously defined SNR with some statistical tool, *e.g.* the correlation coefficient and to determine the relation between them. However, different statistical tools may evolve differently in function of the SNR. For example, the correlation coefficient only depends on the signal variances while different statistical tools (*e.g.* Bayesian classification) take advantage of all the information contained in the leakages probability density functions [11, 36]. This prevents this solution from serving as a good security metric.

Messerges's attack SNR: in [32], Messerges suggested to define an attack SNR in order to characterize a DPA based on a difference of mean test. In general, if we define the random variable $\varDelta_g$ to represent the difference between two mean leakage traces for a good key candidate and the random variable $\varDelta_w$ to represent the same statistic for a wrong key candidate, the attack SNR can be written as:

$$\mathrm{SNR}_{\varDelta} = \frac{\mathbf{E}(\varDelta_g) - \mathbf{E}(\varDelta_w)}{\sigma^2(\varDelta)}$$

From a theoretical point of view, such an attack SNR could be used as a security metric to analyze side-channel attacks since is could be similarly defined for any statistical tool, *e.g.* correlation attacks:

$$\mathrm{SNR}_{\rho} = \frac{\mathbf{E}(\rho_g) - \mathbf{E}(\rho_w)}{\sigma^2(\rho)}$$

It could even be extended to Bayesian classification based attacks:

$$\mathrm{SNR}_{\mathbf{P}[S|O]} = \frac{\mathbf{E}(\mathbf{P}[S_g|O]) - \mathbf{E}(\mathbf{P}[S_w|O])}{\sigma^2(\mathbf{P}[S|O])}$$

Intuitively, an attack SNR determines *how precisely an adversary knows some statistic* while the attack success rate rather determines *how some statistic has turned the available information into a successful key recovery.* We selected the success rate as a security metric because of its clear relation with our formal definition of security.