Physical functions: the common factor of side-channel and fault attacks?

Bruno Robisson, Hélène Le Bouder

▶ To cite this version:

Bruno Robisson, Hélène Le Bouder. Physical functions: the common factor of side-channel and fault attacks?. Journal of Cryptographic Engineering, 2015, 10.1007/s13389-015-0111-4 . emse-01233314

HAL Id: emse-01233314

https://hal-emse.ccsd.cnrs.fr/emse-01233314

Submitted on 24 Nov 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Noname manuscript No.
(will be inserted by the editor)

Physical functions: the common factor of side-channel and fault attacks?

Bruno Robisson · Hélène Le Bouder

the date of receipt and acceptance should be inserted later

Abstract Physical attacks on cryptographic circuits were first identified in the late 1990s. These types of attacks, which are still considered very powerful, are generally classified into two main categories: "fault attacks" and "side-channel attacks." To secure circuits against such attacks, it is crucial to develop appropriate methods and tools that enable accurate estimates of the protection mechanism's effectiveness. Numerous studies have described such methods and tools but, to the best of our knowledge, these previous investigations have considered side-channel attacks or fault attacks but not the combination of the two types. The present article proposes a combined investigation of both main types of attack by describing them with the same terminology and the same algorithm. This approach is made possible by introducing the concept of "physical functions" as an extension of the concept of "leakage functions," which are widely used in the side-channel community. The paper represents a first step toward applying the strong theoretical background developed for side-channel attacks to the investigation of fault attacks. Besides, the proposed approach could potentially make it easier to combine side-channel attacks with fault attacks, which could certainly facilitate the discovery of new attack paths.

Keywords Differential power analysis, differential fault analysis, differential behavioral analysis, template attacks, fault sensitivity analysis, AES_{128}

CEA-TECH / Ecole des Mines de Saint-Etienne, Secure Architecture and Systems Laboratory, Centre Microélectronique de Provence. 880 Route de Mimet, 13541 Gardanne, France
E-mail: bruno.robisson@cea.fr

1 Introduction

Integrated circuits (ICs), which are used in a wide variety of electronic devices, play a key role in securing information and communications technologies. Because certain types of ICs (such as smart cards) may easily fall into the hands of people with malicious intentions, such devices are particularly vulnerable to "physical attacks." In general, the distinctive feature of these attacks is the use of experimental techniques. The first such technique, called "side-channel analysis," is based on observing certain physical characteristics of the targeted device (such as power consumption, electromagnetic radiation, response time, etc.) that vary during the circuit's computation. The second technique, called "fault analysis," is based on disturbing the circuit's behavior through laser beams, voltage or clock glitches, electromagnetic pulses, etc. An attacker can use these techniques to bypass security functions, retrieve design details, or obtain and manipulate data (cryptographic material, PIN codes, personal data, etc.). Techniques in which the attacker targets exclusively cryptographic material are often referred to as "physical cryptanalysis" or "key recovery" algorithms. Various physical cryptanalysis methods have been proposed: for example, differential power analysis [13,27], stochastic attacks [24], template attacks [9,25], collision attacks [5, 19], differential fault attacks [6,4,2,3], and safe-error attacks [29,22,17]. Although these methods may seem initially unrelated because they require different experimental methods and different algorithms, several studies have investigated a common framework to describe side-channel attacks [18,26,10]. In addition, a recent article has proposed another framework to describe all differential fault-attack schemes [12]. By providing a strong theoretical basis, such frameworks aim to enable the development of practical tools that make it possible to estimate the amount of information leaked by circuits, thereby increasing the level of trust in such devices. This paper describes concepts and algorithms that are common to a wide set of key recovery methods, regardless of whether they are based on using fault or side-channel experimental setups. The AES_{128} cryptographic algorithm is used to demonstrate these concepts and algorithms [20].

The paper first describes a general model for the functioning of integrated circuits in the context of physical attacks. "Physical functions" that formalize the relationships between data computed inside the circuit (and considered secret) and certain variables that may be measured externally by the attacker seem to be at the core of such attacks. Several examples of such functions are provided in Section 2. The principles of fault and side-channel attacks that are based on physical functions are described in Section 3. Several illustrations, extracted from the literature, are included.

2 Model of the integrated circuit

2.1 Observables versus internal variables

Physical attacks are based on experiments involving cryptographic operations that act on the circuit with or without a modification of its behavior. During these experiments, the attacker is able to directly measure data (also called "observables") such as

- Electromagnetic emissions (EM), power consumption traces (power), or signals provided by particular sensors called μ-probes.
- Plaintext or ciphertext obtained without or with perturbations (called "faulty" ciphertext).
- The circuit behavior (i.e., normal or abnormal functioning) in case of fault. This behavior may be determined, for example, by comparing correct and faulty ciphering, by detecting an alarm signal, or by detecting a more or less premature stop in computation.
- The time needed to execute the cryptographic operations. One way, among others, to measure this computation time is to reduce the clock period step by step (on a specific clock cycle) and analyze the circuit's behavior. The step in which the behavior of the circuit switches from normal to abnormal is considered an indicator of the computation time. This particular step is also referred to as "fault intensity" in [17].

Note that the term "observable" may refer to any combination of observables or any observable obtained by applying a signal processing technique or a mathematical transformation (such as filtering, temporal to frequency transformation, etc.).

In contrast to the observables, the execution of a cryptographic algorithm involves data that cannot be measured directly by the attacker. In the following paragraphs, the terms "internal" or "intermediate" are used interchangeably to refer to this type of data. The attacker looks for certain such internal variables that are called "target" or "secret" data.

2.2 Relationships

Variables are related to each other because they are computed by, or originate from, the same circuit (i.e., internal data and observables). Hardware and/or software descriptions generally define the links that exist between pieces of internal data during the normal functioning of the circuit (i.e., without perturbations created by the attacker). These links are also known as "algorithmic" relationships. The possible connections between internal data and observables are referred to as "physical" relationships.

2.2.1 Algorithmic relationships

The present investigation used the 128-bit version of the AES standard (denoted by AES_{128}), which is a specification for symmetric key cryptography established by the U.S. NIST [20]. It ciphers a 128-bit piece of data called "plaintext" (or "input") by using a 128-bit key, producing a 128-bit piece of data called "ciphertext" (or "output"). The encryption process consists in first transforming the input data into a two-dimensional array of bytes, known as a "state." Then, following a preliminary bitwise XOR operation between the input and the key, AES_{128} executes 10 instances (also called "rounds") of a function that operates on the state. The following operations are used during these rounds:

- SubBytes (SB) is a non-linear transformation that works independently of the individual bytes of a state. The result of this operation for round i is denoted by $rnd[i].s\_box$.
- ShiftRows (SR) is a rotation operation acting on each row of the state. The result of this operation for round i is denoted by $rnd[i].s\_row$.
- MixColumns (MC) is a linear matrix multiplication that acts on each column of the state. The result of this operation for round i is denoted by $rnd[i].m\_col$.
- AddRoundKey (ARK) is a bitwise XOR operation between the state and $k\_sch[i]$. $k\_sch[i]$ is computed according to a transformation referred to as a "key expansion," which is not detailed here. The result of this operation is the start of the next round denoted by $rnd[i+1].start$.

AES rounds are identical, except for the final round, which skips the MixColumns operation. At the end of the ciphering operations, the state is copied to the output.

2.2.2 Physical relations

For side-channel attacks, the connection between internal data and observables is referred to as a"leakage function"[26]. To extend this notion to fault attacks, the present investigation also considered "error functions," which are used to model the effect of a fault injection on a piece of internal data. Leakage and error functions are referred to as "physical functions" because they are both at the heart of any physical attack. These functions are detailed in the following section.

3 Models of physical functions

Models of physical functions proposed in the literature can be divided into 1) "real" functions (i.e., each input corresponds to exactly one output) and 2) functions in which inputs and outputs are considered random variables; these two categories are respectively known as "deterministic" and "probabilistic" physical functions. A nonexhaustive list of typical physical functions is proposed in the following.

3.1 Preliminary definitions

3.1.1 Data representation

A byte B is a set of 8 bits denoted by $\{b_7, \ldots, b_0\}$ and B is matched with the binary value $B_b = b_7b_6b_5b_4b_3b_2b_1b_0$; to simplify, B and B_b are used interchangeably. The decimal value associated with B is denoted by B_d (for this conversion, b_0 is the less significant bit).

3.1.2 Bit level functions

The logical operators AND (symbol $\wedge$), OR (symbol $\vee$), and XOR (symbol $\oplus$) defined for a byte are considered bitwise.

3.1.3 Subsets of bits

Let B be a set of N bits $\{b_{N-1}, \ldots, b_0\}$. Let Ω be a set of N bits $\{\omega_{N-1}, \ldots, \omega_0\}$. The set of bits b_i in B, such that ω_i in Ω is equal to 1, is denoted by $R_{\Omega_d}(B)$. Note that, if B is composed of N values, there are $2^N - 1$ possible subsets. Example with $N = 8$: let $\Omega = \{0, 0, 1, 0, 0, 1, 0, 0\}$ and $B = \{b_7, \ldots, b_0\}$ be two bytes. In this case, the bits ω_2 and ω_5 are equal to 1 and the decimal value of Ω is $\Omega_d = 2^5 + 2^2 = 36$. Thus, the subset $R_{36}(B)$ is the set of bits $\{b_5, b_2\}$ in B. In the following, a subset is referred to as *monobit* when it contains only 1 bit. The 8 monobit subsets of a byte B are denoted by $R_1, R_2, R_4, R_8, R_{16}, R_{32}, R_{64}, R_{128}$.

3.1.4 Subset of bits equal to

Let Ω and B be two sets of N bits. Let M be the number of bits of the subset $R_\Omega(B)$ and α be a set of M bits. The function denoted by $R_\Omega(B) == \alpha_d$ returns 1 when, for all bits $i \in \{0, \ldots, M-1\}$ in $R_\Omega(B)$ and in α, we have $R_\Omega(B)_i == \alpha_i$ and returns 0 otherwise. For example, as explained above, $R_{36}(B)$ contains two bits which may take on four different values. $R_{36}(B) == 0$ is equal to 1 for any B such that $B = \{b_7, b_6, 0, b_4, b_3, 0, b_1, b_0\}$, irrespective of the values of b_7,b_6,b_4,b_3,b_1 and b_0. Similarly, $R_{36}(B) == 1$ is equal to 1 for any B such that $B = \{b_7, b_6, 0, b_4, b_3, 1, b_1, b_0\}$, irrespective of the values of b_7,b_6,b_4,b_3,b_1 and b_0.

3.2 Deterministic physical functions

3.2.1 Deterministic leakage functions

The simplest model proposed for power consumption is based on considering the value of a single bit. In this case, the leakage function is R_i with $i \in \{1, 2, \ldots, 128\}$. In general, side-channel functions are typically chosen as a combination of subsets and weighted sums. For example, the number of 1's in a binary number B is called the *Hamming weight* and is denoted by $HW(B)$. A variation considers the number of transitions between two binary values B and C. This function, called the *Hamming distance*, is denoted by $HD(B, C) = HW(B \oplus C)$.

3.2.2 Deterministic error functions

Bit flip A fault may invert a set of bits. The function that models the bit flip is the bitwise XOR between B and Ω with $\Omega_d \in [0, \ldots, 255]$. If all bits in Ω are equal to 1, then all bits are inverted; if all bits in Ω are equal to 0, then all bits remain unchanged.

Set and reset A fault that forces bits to take on the value 0 is generally called a *reset*. The function that models such a reset is the bitwise AND between B and Ω with $\Omega_d \in [0, \ldots, 255]$. Conversely, a fault that forces bits to take on the value 1 is generally called a *set*. The function that models such a set is the bitwise OR between B and Ω with $\Omega_d \in [0, \ldots, 255]$. A reset with $\Omega_d = 0$ returns a set of 0. A set with $\Omega_d = 255$ returns a set of 1.

Behavior In a generalization of the *set* and *reset* models, a fault may force a set of bits $R_\Omega(B)$ in B to take on some value α. In such a case, the circuit behavior is normal only when the perturbed result is expected to be equal to α and is abnormal otherwise. This behavior can be described by using the function $R_\Omega(B) == \alpha$.

Running time (or 'fault intensity') Faults that are produced by violating the latch setup times may either affect only one bit or affect initially bytes with a higher Hamming weight [1,17]. In the first case, the chip's behavior can be described according to the function that selects one bit R_i with $i \in \{1, 2, 4, \ldots, 128\}$; in the second case it is described according to the Hamming weight.

3.3 Probabilistic physical functions

In the probabilistic case, the input X and output Y of the physical functions are considered discrete random variables. X and Y have sample spaces of $S_X = \{0; 2^M - 1\}$ and $S_Y = \{0; 2^N - 1\}$, respectively. In accordance with [19,15], the probabilistic relationship is modeled with a joint probability mass function (joint *pmf*). The joint *pmf* of the discrete variables X and Y (used to model a physical function f with domain X and co-domain Y) is denoted by $f_{X,Y}(x, y) = Pr(X = x, Y = y)$, irrespective of $x \in S_X$ and $y \in S_Y$. Examples of joint *pmf* are described in the following sections.

3.3.1 Probabilistic leakage functions

Consider a physical relationship that links the integer values of an 8-bit piece of data to the measured current consumed by the circuit handling these values. A classic probabilistic leakage function considers that the current consumption of the data x is the sum of the Hamming weight of x and a Gaussian noise [9]. This assumption holds, for example, when the data switches from the all-zero value to the value x. This switch occurs, for example, during the transfer of data between the memory and the pipeline of a microprocessor on a pre-charged bus. In such a case, we have $f(x) = a * HW(x) + e$, where e models the "noise" as a normal random variable with a mean 0 and a given standard deviation. Figure 1 (left) describes the corresponding *pmf* (the values of $a = 6$ and a noise with a standard deviation of 2 were chosen arbitrarily to illustrate the example). Integer values for 8-bit data are reported in abscissa; the corresponding current consumption is given in ordinates. The grayscale illustration (Fig.1, left) represents probabilities: the darker the shade, the higher the probability that the circuit consumes a current of value y when the manipulated data has the value x (shorter x "consumes" y). Figure 1 (right) illustrates the same *pmf* measured for a real component. Measurements were obtained with a 32-bit software implementation of AES_{128} running on a 32-bit unsecured up-to-date microcontroller. This particular case shows a remarkably good fit of the theoretical and practical relationships.

3.3.2 Probabilistic error functions

Consider a physical relationship that, for an 8-bit register, links the integer values to the values that were modified through some type of fault injection. One widely used model is based on the assumption that any one of the register's 8 bits will flip, with equal probabilities for each of these bit flips (i.e., equal likelihood hypothesis). Here, this model, denoted by $f(x) = x \oplus \Omega$ with $\Omega \in \{1, 2, 4, \ldots, 128\}$ in decimal value and $(Pr(\Omega) = 1/8)\forall\Omega$, is referred to as the "random monobit fault model." This type of theoretical model was, for example, used in [11]. Figure 2 (left) illustrates the corresponding *pmf.* The integer values of the 8-bit register are reported in abscissa; the corresponding faulted value is given in ordinates. The grayscale illustration (Fig.2, left) represents probabilities: the darker the shade, the higher the probability that x is changed into y. Figure 2 (right) shows the same *pmf* measured experimentally, obtained with a hardware implementation of the AES_{128}, which was faulted by reducing the clock period that feeds the circuit (in accordance with [1]). To avoid creating faults on several bits instead of on only one bit, this clock period was deliberately reduced by only a small amount. The faulted bit was dependent on the plaintext value.

In this example, there is a clear distinction between the theoretical and the experimental *pmf.* In fact, although monobit faults are identifiable in the experimental setup, some bits seem to be more easily modifiable than others. In this case, the equal likelihood hypothesis does not hold.

4 Physical functions for physical cryptanalysis

The scenario of a physical cryptanalysis (i.e., targeting cryptographic material only) is exemplified by numerous attacks that have been proposed in the literature: μ-probing attacks, differential power analysis (with or without a profiling phase), differential fault analysis, and safe-error attacks (e.g., differential behavioral analysis and fault sensitivity analysis). The key retrieval algorithm described in the following applies to all of these physical attacks.

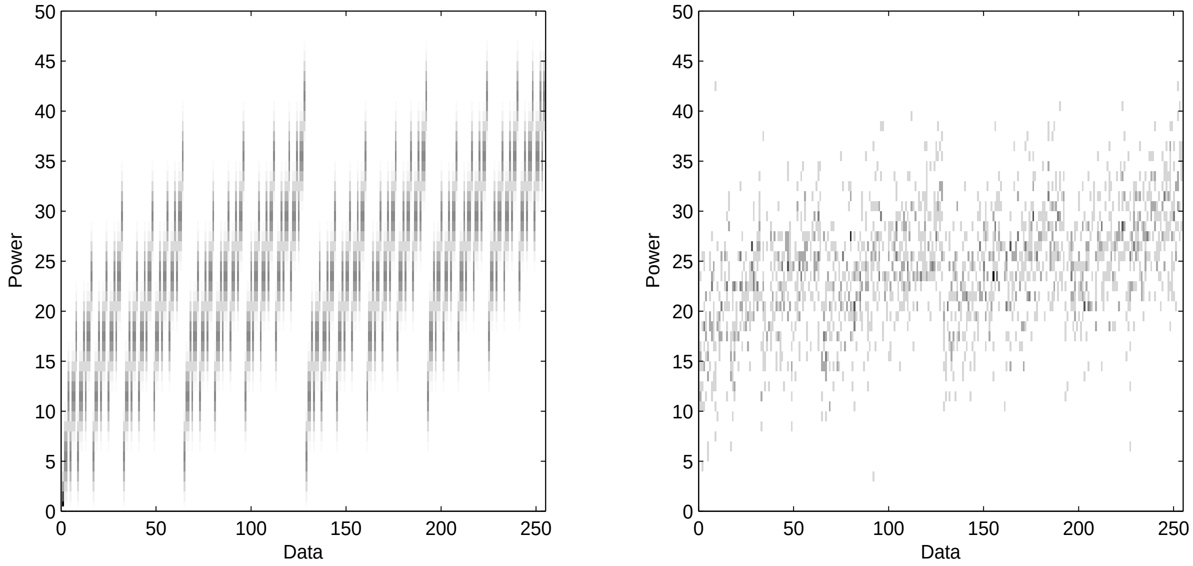

Fig. 1 Joint *pmf* of the leakage functions: theoretical (left) and experimental (right)

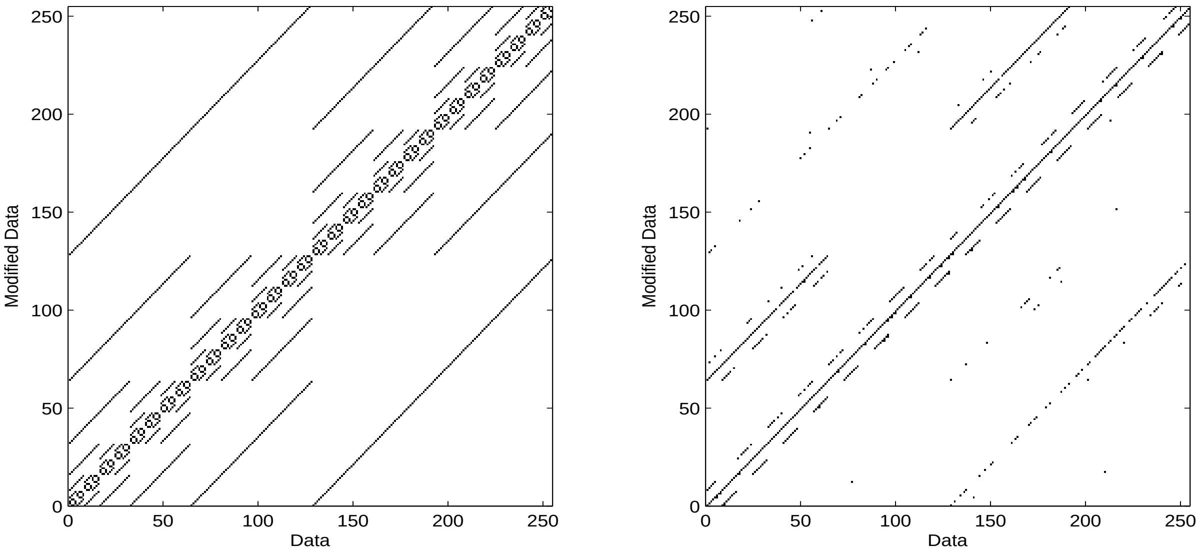

Fig. 2 Joint *pmf* of the error functions: theoretical (left) and experimental (right)

4.1 Attack path

4.1.1 Definition

An attack path is a relationship between observables that involves the secret (or only part of the secret) and that enables the recovery of information about it. In the following paragraphs, $P = REL(C, F, O)$ denotes the attack path that links the observables O to P according to the (unknown) value of the internal variables C (among them, the secret) and according to a set of (unknown) physical functions F. Although the relationship between P and O is not necessarily causal, the observables O are also known as "stimuli" and the observables P are referred to as "reactions" or "responses".

4.1.2 Examples of attack paths

The attack path used in [13] and denoted by REL_{power} links the j^{th} plaintext byte (denoted by $plain^j$) to the j^{th} byte of the first round key (denoted by $k\_sch[0]^j$) and to the power consumption that is measured during the computation of the j^{th} SubBytes of the first round. The algorithmic functions involved in this relationship are the SubBytes and the bitwise XOR of the AES_{128}. The physical functions involved in this relationship create a link between the value of the output of the j^{th} SubByte in the first round and the power consumption of the circuit when this value is computed. The associated physical function, denoted by f_{power}, has no analytical expression. The secret variable here is $\{k\_sch[0]^j\}$. With

- $O = \{plain^j\}$,
- $P = \{power\}$,
- $C = \{k\_sch[0]^j\}$, and
- $F = \{f_{power}\}$, the relationship REL_{power} may be written as
- $power = f_{power}(SB(ARK(plain^j, k\_sch[0]^j)).$

Several variants of this relationship may be used interchangeably for either side-channel or fault attacks. In the first case, the attacker might take measurements of the electromagnetic radiation or of a signal recovered by a μ-probe. In the second case, the observables, such as the behavior or the running time, are obtained by modifying the circuit's functioning. In either case, to obtain the corresponding relationship, it is sufficient to replace the physical function f_{power} with either f_{EM}, $f_{\mu-probe}$, $f_{behaviour}$, or f_{time}.

This relation and other examples of relations that are commonly used to break the AES_{128} with physical attacks are reported in Table 1. In these relations, k denotes the index of a byte of the AES_{128} state (thus, $0 \leq k < 16$) before the $ShiftRow$ operation, and j is the index of the same byte after this operation (thus, $0 \leq j < 16$). According to the notation used in the standard, $k = r + 4 * c$, with $0 \leq r < 4$, $0 \leq c < 4$, and $j = r + 4 * c'$, where $c' = (r - c)Modulo(4)$. In these relations, the functions f, g, and h are leakage or error functions.

In Table 1, REL_2 is identical to REL_1 (described above) except it pertains to the end of the AES execution. The relation links two measured observables to perform, for example, a differential behavioral analysis [22]: the first observable (denoted by O) is the value of the ciphertext obtained without error; the second observable (denoted by P) is the behavior of the chip measured when the state is disturbed by means of a fault injection at the beginning of the final round. In this case, the relation involves the value of the key of the final round as well as an error function that models the impact of the fault injection on the state at the beginning of the final round. Several variants of this relation have been used. For example, one variant measures the sensitivity of the fault instead of the behavior of the circuit to perform a fault sensitivity analysis [17].

REL_3 links two observables measured with a differential fault analysis [11]: the first observable (denoted by O) is the value of the ciphertext obtained without any perturbation; the second observable (denoted by P) is the value of the ciphertext obtained with perturbations. This relation involves the value of the key of the final round as well as an error function. This error function models the impact of the fault setup on the state at the beginning of the final round.

REL_4 links two observables measured to mount, for example, a combined attack, as proposed by Roche et al. in [23]. The first observable (denoted by O) is the ciphertext value obtained without any perturbation; the second observable (denoted by P) is the power consumption of the ciphertext value that contains an error. This relation involves the value of the key of the final round, two error functions, and a leakage function. The first error function models the impact of the fault setup on the last round key. The second error function models the impact of the fault setup on the second to last round key. The leakage function models the power consumption of the ciphertext containing an error. Some of the attacks that use these relations are detailed in Section 4.3.

4.2 Key retrieval algorithm

Step 0: Notations Let A be a set of n variables denoted by $\{a_1, a_2, \ldots, a_n\}$. Assume that each variable a_i is discrete and has its own finite set of definitions, denoted by A_i with cardinal $|A_i|$. If a variable a_i takes a particular value, denoted by $a_i(j)$, with $0 \leq j < |A_i|$, then this variable is considered "instantiated." By extension, $\tilde{A}$ denotes the set of all possible values of A. Thus, there are $|\tilde{A}| = |A_1| * |A_2| * \ldots * |A_n|$ distinct possible instances. Each of these instances is denoted by $\tilde{A}(j)$.

Step 1: Choosing the attack path Among the different relationships described in Table 1 and depending on the experimental setup, the attacker chooses a relation $P = REL(C, F, O)$, for which C and F are unknown and O and P are known to the attacker.

Step 2: Defining the search space The attacker chooses hypotheses for the values of C and F. The set of hypotheses for the physical functions is denoted by $\tilde{F}$; the set of hypotheses for the internal variables is denoted by $\tilde{C}$. The set $\tilde{C} * \tilde{F}$ is called the search space and has $|\tilde{C}| * |\tilde{F}|$ elements. These sets are chosen such that $\tilde{C}$ contains C and such that at least one physical function of $\tilde{F}$ "approximates" the function F. Because it is impossible to test all possible physical functions (the set

REL_0	$O = \{plain\}$ $P = \{cipher\}$ $C = \{k\_sch[0]\}$ $cipher = AES(plain, f(k\_sch[0]))$
REL_1	$O = \{plain^j\}$ $P = \{power\}$ $C = \{k\_sch[0]^j\}$ $power = f(rnd[1].start^j)$ with $rnd[1].start^j = SB(ARK(plain^j, k\_sch[0]^j))$
Var.	$P = \{EM\}$ or $P = \{\mu\_probe\}$ or $P = \{behaviour\}$ or $P = \{time\}$
REL_2	$O = \{cipher^j\}$ $P = \{power\}$ $C = \{k\_sch[10]^j\}$ $power = f(rnd[10].start^k)$ with $rnd[10].start^k =$ $SB^{-1}(SR^{-1}(ARK(cipher^j, k\_sch[10]^j)))$
Var.	$P = \{EM\}$ or $P = \{\mu\_probe\}$ or $P = \{behaviour\}$ or $P = \{time\}$
REL_3	$O = \{cipher^j\}$ $P = \{faulted^j\}$ $C = \{k\_sch[10]^j\}$ $faulted^j =$ $ARK(SR(SB(f(rnd[10].start^k))), k\_sch[10]^j)$ with $rnd[10].start^k =$ $SB^{-1}(SR^{-1}(ARK(cipher^j, k\_sch[10]^j)))$
REL_4	$O = \{cipher^j\}$ $P = \{faulted^j\}$ $C = \{k\_sch[10]^j, rnd[9].m\_col^j\}$ $faulted^j = h(ARK(rnd[10].s\_row^{j'}, f(k\_sch[10]^j)))$ with $rnd[10].s\_row^{j'} =$ $SR(SB(ARK(k\_sch[9]^{k'}, rnd[9].m\_col^k)))$ $k\_sch[9]^{k'} = g(ARK(rnd[10].start^k, rnd[9].m\_col^k))$ $rnd[10].start^k =$ $SB^{-1}(SR^{-1}(ARK(cipher^j, k\_sch[10]^j)))$
Var.	$P = \{power\}$

Table 1 Relationships used to break the AES_{128} with physical cryptanalysis

of leakage functions is infinite, and the set of all error functions for an octet is composed of $256^{256} = 2^{2048}$ elements), heuristics must be used. These heuristics are based on the attacker's knowledge of the circuit and on the chosen experimental setup. However, in some scenarios, the attacker may take advantage of a "profiling" phase on a circuit identical to the circuit under attack. During such an optional phase, the attacker is then able to estimate the leakage functions (and thus to refine the hypotheses). In all other cases, the attacker models the physical functions with the functions described in Section 3.

Step 3: Experiments The attacker performs a set of experiments to measure and record the circuit's reactions P to different stimuli O. The stimulus used for the experiment e is denoted by $O(e)$; the measured reaction for this experiment is denoted by $P(e)$. Based on the set of all measurements, the attacker is able to compute $Pr(P, O)$.

Step 4: Predictions The attacker builds a set of relationship hypotheses by replacing all elements of C in REL with the values $\tilde{C}(i) \in \tilde{C}$ of the internal variables and replacing all elements of F with the physical functions $\tilde{F}(j) \in \tilde{F}$. Each element in this set of relationships is called a "model" parametrized by $\tilde{C}(i)$ and $\tilde{F}(j)$ and denoted by $Mod(i, j)$. Thus, depending on the type of physical function chosen (deterministic or probabilistic), two cases are possible:

- The deterministic case: For all models and for each experiment e performed during the previous step, the attacker predicts the reactions denoted by $P'_{Mod(i,j)}(e) = REL(\tilde{C}(i), \tilde{F}(j), O(e))$.
- The probabilistic case: For all models, the attacker estimates the reaction probability and the stimuli $Pr(P'_{Mod(i,j)}, O)$ from the set of all experiments.

Note that the joint probability $Pr(P'_{Mod(i,j)}, O)$ may also be calculated in the first case (i.e., when the physical functions are deterministic). Further note that in the second case, the stimuli used to compute the joint probability do not have to be strictly identical to those used during the measurement step. They are merely required to have the same probability distribution.

Step 5: Comparing predictions with measurements The attacker then compares the predictions with the measurements. Depending on the values of the hypothesis, two cases are possible if the relation REL has certain "good" statistical properties (which is the case for the relations reported in Table 1).

1. Correct hypothesis: If a fair approximation of the physical function F can be obtained through a function in $\tilde{F}$, then there exists an index j_0 such that $\tilde{F}(j_0) \simeq F$. In this case, for $i = i_0$, where i_0 is the index of C in $\tilde{C}$, the predicted reactions are "quasi-identical" to the measured reactions for all experiments. Thus, we have $P'_{Mod(i_0,j_0)}(e) \simeq P(e) \forall e$ for deterministic physical functions and $Pr(P'_{Mod(i_0,j_0)}, O) \simeq Pr(P, O)$ for probabilistic physical functions. In this case, the model parametrized by the key $\tilde{C}(i_0)$ and the function $\tilde{F}(j_0)$ "explains the data".
2. Incorrect hypothesis: In contrast, if either $i \neq i_0$ or $j \neq j_0$, then $P(e)$ and $P'_{Mod(i,j)}(e)$ are not identical for all experiments in the deterministic case, and $Pr(P'_{Mod(i_0,j_0)}, O)$ and $Pr(P, O)$ are distinct in the probabilistic case.

Other algorithms may be used to implement the comparator $\simeq$ cited above (also called a "distinguisher"). For the deterministic case, ad-hoc algorithms, such as the following three examples, have been proposed:

- The "sieve" algorithm returns a binary value equal to 1 if and only if $P'_{Mod(i,j)}(e) = P(e)$ for all experiments. The model explains the data if and only if the return value is 1.
- The "count" algorithm returns the number of times that $P'_{Mod(i,j)}(e) = P(e)$. The higher this number, the better the model $Mod(i,j)$ explains the data.
- For predictions whose possible values are restricted to the set $\{0,1\}$, Kocher et al. calculated the difference of means (DoM) for the set of measurements corresponding to a prediction of 1 and the mean of the set of measurements corresponding to a prediction of 0 [13].
  The higher the difference of means, the better the model $Mod(i,j)$ explains the data.

Additionally, more general algorithms based on information theory or on statistical operators have been proposed. For deterministic cases, testing the statement $P'_{Mod(i_0,j_0)}(e) \simeq P(e) \forall e$ is equivalent to comparing the *pmf* of $Pr(P'_{Mod(i_0,j_0)})$ and $Pr(P)$. For probabilistic cases, determining whether the joint *pmf* $Pr(P'_{Mod(i_0,j_0)}, O)$ and $Pr(P,O)$ are identical is equivalent to comparing the *pmf* of $P'_{Mod(i_0,j_0)}$ and $Pr(P)$. Thus, the two cases (i.e., probabilistic and deterministic) collapse. For comparison, a survey of operators is available in [8]. Two of these operators are described below.

- Based on the expected similarity of predictions and measurements, the Pearson coefficient is used to test the linearity between $P'_{Mod(i,j)}$ and P. This coefficient returns a value between 1 and -1. The higher this absolute value, the better the model explains the data.
- Because predictions and measurements are expected to have information in common, a method based on mutual information (MI) between the random variables $P'_{Mod(i,j)}$ and P has also been proposed. The MI returns a value between 0 and 1. The higher the value, the better the model explains the data.

Note that, when cases of correct and incorrect hypotheses are distinguishable, the attacker obtains information concerning internal data (value of C) as well as concerning the physical functions associated with the circuit and with the experimental setup. Examples of key retrieval algorithms, extracted from the literature, are described in the following section.

4.3 Illustrations

Table 2 presents the various parameters defined in the framework proposed above for some well-known attacks on the AES. For the 128-bit version of this algorithm, the attacker must perform an elementary attack 16 times (enabling the attacker to retrieve the octet j of the key).

Semi-exhaustive This rather theoretical attack is based on using any means of fault injection to force 15 bytes of the key to 0 while leaving the value of the byte j unchanged. The attacker records a plaintext and a ciphertext obtained under such conditions. In the prediction step, the attacker generates the set of 256 keys such that all the bytes of these keys are equal to 0, except the j^{th} byte, which takes on a value of i in $\{0,\dots,255\}$. Using the AES algorithm, the attacker computes the ciphertext obtained for each value of this set of key hypotheses. Only one of these values, for example i_0, explains the ciphertext through the plaintext. The attacker concludes that the byte j of the key is equal to i_0.

μ-probing Suppose, for example, that the attacker targets a software implementation of the AES on an 8-bit microcontroller, knowing exactly when the S-Box output of the first round for one byte of the state is moved from the processor to memory. Further suppose that the attacker is able to measure the signal of one (and only one) wire of the data bus, but that the index of the corresponding bit is unknown. As a result, the attacker must test 8 different physical functions, each of them a monobit subset that corresponds to an index of the bit in the byte. Various kinds of distinguishers could be used to retrieve the value of the key hypothesis.

DPA In DPA (differential power analysis), the physical function used to model the power consumption is a monobit subset. Thus, the attacker must test 8 possible physical functions (one for each bit) for each of the 256 values of the key hypothesis. The attacker could use either the difference of means or the Pearson correlation coefficient to discriminate between the correct key and the false key.

CPA In CPA (correlation power analysis), the attacker assumes that the power consumption depends on the SubByte output and on an additional value (which could be the previous value in the register that stores the SubByte outputs). Because this value is unknown, it may be considered a parameter of the leakage function. Thus, the attacker must test all 256 values of the secret key for each of the 256 leakage functions. In CPA, the Pearson correlation coefficient is the distinguisher.

Attack	Relationships	Physical function	Type of physical function	Similarity and distance tools
Semi-exhaustive (on octet j)	R_0 $O = \{plain\}$ $P = \{cipher\}$ $C = \{k\_sch[0]\}$	$f(x) = x$ if x is the j^{th} octet else $f(x) = 0$	Determ.	Sieve
μ-probing	R_1 $O = \{plain^j\}$ $P = \{\mu - probe\}$ $C = \{k\_sch[0]^j\}$	$f(x) = R_\Omega(x)$ with $\Omega \in \{1, 2, 4, \ldots, 128\}$	Determ.	All
DPA [13]	R_2 $O = \{cipher^j\}$ $P = \{Power\}$ $C = \{k\_sch[10]^j\}$	$f(x) = R_\Omega(x)$ with $\Omega \in \{1, 2, 4, \ldots, 128\}$	Determ.	DoM or Pearson correlation
CPA [7]	R_1 $O = \{plain^j\}$ $P = \{power\}$ $C = \{k\_sch[0]^j\}$	$f(x) = HW(x \oplus \Omega)$ with $\Omega \in [\![1, 255]\!]$	Determ.	Pearson correlation
MIA [28]	R_1 $O = \{plain^j\}$ $P = \{power\}$ $C = \{k\_sch[0]^j\}$	$f(x) = HW(x) + N$ with N a Gaussian noise	Probab.	Mutual information
DFA1 [11]	R_3 $O = \{cipher^j\}$ $P = \{faulted^j\}$ $C = \{k\_sch[10]^j\}$	$f(x) = x \oplus \Omega$ with $\Omega \in \{1, 2, 4, \ldots, 128\}$ and $(Pr(\Omega) = 1/8)\ \forall \Omega$	Probab.	Sieve
DFA2 [23]	R_4 $O = \{cipher^j\}$ $P = \{faulted^j\}$ $C = \{k\_sch[10]^j,$ $rnd[9].m\_col^j\}$	$h(x) = x$ and $g(x, \Omega) = x \oplus \Omega$ with $\Omega \in [\![1, 255]\!]$ $f(y, \Gamma) = y \oplus \Gamma$ with $\Gamma \in [\![1, 255]\!]$	Determ.	Count
DFA3 [23]	R_4 $O = \{cipher^j\}$ $P = \{power\}$ $C = \{k\_sch[10]^j,$ $rnd[9].m\_col^j\}$	$h(x) = HW(x)$ f and g as above	Determ.	Pearson correlation
DBA [22]	R_1 $O = \{plain^j\}$ $P = \{behavior\}$ $C = \{k\_sch[0]^j\}$	$f(x) = (R_\Omega(x) == 0)$ with $\Omega \in [\![1, 255]\!]$	Determ.	Pearson correlation
FSA [17]	R_2 $O = \{cipher^j\}$ $P = \{intensity^j\}$ $C = \{k\_sch[10]^j\}$	$f(x) = HW(x)$ or $f(x) = R_\Omega(x)$ with $\Omega \in \{1, 2, 4, \ldots, 128\}$	Determ.	Pearson correlation

Table 2 Examples of physical attacks and their corresponding parameters

MIA In MIA (mutual information analysis), the attacker assumes that the power consumption depends on the SubByte output and on an additive Gaussian noise (the model is thus probabilistic). In MIA, mutual information is the distinguisher.

DFA1 The DFA (differential fault analysis) proposed in [11] used the monobit fault model described in Section 3.3.2. The distinguisher is a sieve algorithm.

DFA2 The DFA proposed in [23] used constant errors (or at least errors that offer "good repeatability") distributed across the key schedule. These errors create a fault on the last round key and a fault on the second to last round key; the values of these faults are unknown to the attacker. The attacker must test all possible values of these faults (in addition to testing all possible values of the key) in order to find the model that explains the data. The distinguisher used is a count algorithm.

DFA3 The third DFA is based on the DFA2 but takes advantage of side-channel leakage. In such an attack, three physical functions are used: the two error functions of DFA2 and the leakage function of the faulted ciphertext. The chosen distinguisher is the Pearson coefficient.

DBA DBA (differential behavioral analysis) explains a behavior (i.e., the circuit either functions correctly or malfunctions) based on a model that is parametrized by the value of the key and a constant (but unknown) fault model (e.g., the set or the reset of some bits of the SubByte outputs). The distinguisher is a Pearson correlation. This attack is particularly effective for circuits with embedded protections against DFA.

FSA FSA (fault sensitivity analysis) explains a fault sensitivity (i.e., the reaction of the circuit to a fault injection whose intensity is variable) based on a model parametrized by the value of the key and a constant (but unknown) fault model (e.g., a Hamming weight or set/reset of a particular bit of the SubByte input). The distinguisher is a Pearson coefficient.

4.4 Discussion

The use of physical functions for physical cryptography raises multiple questions:

- How can the error between the model and the actual physical functions be estimated? This question has been investigated for side-channel attacks in several articles such as in [10]. However, previous findings will likely have to be updated to take into account the *pmf* of the error functions, which, in fact, have very different statistical properties than the *pmf* of the leakage functions.
- Do some physical functions leak more information than others? As suggested in [14,16,12], fewer risks certainly exist if the *pmf* of a physical function is a uniformly random distribution than if the *pmf* of the function corresponds to one of those described in Section 3.3.2. It may be of interest to find mathematical criteria (likely based on the measure of randomness of the *pmf*) to evaluate the intrinsic leakage of physical functions.
- Following from the first two questions, how can the physical functions that leak the most be identified? The present investigation considered that the physical functions take only one argument. Yet, theoretically, each observable is dependent on all other variables. For example, an error function could be dependent either on X, on the previous value of X, or on other observables such as the voltage, the temperature, and even (perhaps, due to cross-talk) on many other internal variables. Thus, there seems to be a tremendous number of possible physical functions. It is impossible to enumerate all of them, even for a very small circuit. It will be up to the community to define heuristics to find the most critical physical functions. However, searching for answers in such a nonexhaustive manner could result in potential undetected leakage.

Conclusion

In this article, we have shown that most side-channel and fault attacks share the same principles. This merging of shared characteristics was accomplished by extending the concept of leakage functions to the concept of error functions and by using probability mass functions to model both functions. Numerous possibilities for further research exist: First, we plan to use the presented framework to describe a wider set of attacks. In particular, it could be of interest to extend the proposed approach to asymmetrical cryptography and to include other attacks such as those described in [19,2,3,21]. Second, for a given set of models of physical functions, the attack paths could be studied formally, enabling the automatic generation of new attack paths. Third, this work may be regarded as a basis to aggregate the advantages of attack-specific protections, thereby providing a more generic set of countermeasures.

Acknowledgements The authors would like to thank the members of the Secure Architectures and Systems Laboratory for their contributions to this work. We would also like to thank the anonymous reviewers and Ph. Maurine and K. Adbellatif for their valuable comments and suggestions. This research was supported in part by the French government through the HOMERE+ and the PANDORE (ANR-14-CE28-0027) projects.

References

1. Michel Agoyan, Jean-Max Dutertre, David Naccache, Bruno Robisson, and Assia Tria. When clocks fail: On critical paths and clock faults. In Dieter Gollmann, Jean-Louis Lanet, and Julien Iguchi-Cartigny, editors, *CARDIS*, volume 6035 of *Lecture Notes in Computer Science*, pages 182–193. Springer, 2010.
2. SkSubidh Ali, Debdeep Mukhopadhyay, and Michael Tunstall. Differential fault analysis of aes: towards reaching its limits. *Journal of Cryptographic Engineering*, 3(2):73–97, 2013.
3. Gilles Barthe, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Jean-Christophe Zapalowicz. Synthesis of fault attacks on cryptographic implementations. *IACR Cryptology ePrint Archive*, 2014:436, 2014.
4. Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In B.S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer, 1997.
5. Johannes Blömer and Volker Krummel. Fault based collision attacks on AES. In Luca Breveglieri, Israel Koren, David Naccache, and Jean-Pierre Seifert, editors, *FDTC*, volume 4236 of *Lecture Notes in Computer Science*, pages 106–120. Springer, 2006.
6. Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults. In W. Fumy, editor, *Advances in Cryptology - EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer, 1997.

7. Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.
8. Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007.
9. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES '02: Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28, London, UK, 2002. Springer-Verlag.
10. François Durvaux, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. How to certify the leakage of a chip? In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 459–476. Springer, 2014.
11. Christophe Giraud. DFA on AES. In H. Dobbertin, V. Rijmen, and A. Sowa, editors, *Advanced Encryption Standard - AES*, volume 3373 of *Lecture Notes in Computer Science*, pages 27–41. Springer, 2005.
12. Xiaofei Guo, Debdeep Mukhopadhyay, Chenglu Jin, and Ramesh Karri. Security analysis of concurrent error detection against differential fault analysis. *Journal of Cryptographic Engineering*, 5(3):153–169, 2015.
13. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.
14. Ronan Lashermes, Guillaume Reymond, Jean-Max Dutertre, Jacques Fournier, Bruno Robisson, and Assia Tria. A DFA on AES based on the entropy of error distributions. In Guido Bertoni and Benedikt Gierlichs, editors, *FDTC*, pages 34–43. IEEE, 2012.
15. Yang Li, Sho Endo, Nicolas Debande, Naofumi Homma, Takafumi Aoki, Thanh-Ha Le, Jean-Luc Danger, Kazuo Ohta, and Kazuo Sakiyama. Exploring the relations between fault sensitivity and power consumption. In Emmanuel Prouff, editor, *COSADE*, volume 7864 of *Lecture Notes in Computer Science*, pages 137–153. Springer, 2013.
16. Yang Li, Yu ichi Hayashi, Arisa Matsubara, Naofumi Homma, Takafumi Aoki, Kazuo Ohta, and Kazuo Sakiyama. Yet another fault-based leakage in non-uniform faulty ciphertexts. In Jean Luc Danger, Mourad Debbabi, Jean-Yves Marion, Joaquín García-Alfaro, and A. Nur Zincir-Heywood, editors, *FPS*, volume 8352 of *Lecture Notes in Computer Science*, pages 272–287. Springer, 2013.
17. Yang Li, Kazuo Sakiyama, Shigeto Gomisawa, Toshinori Fukunaga, Junko Takahashi, and Kazuo Ohta. Fault sensitivity analysis. In Stefan Mangard and Franois-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 320–334. Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-15031-9_22.
18. Silvio Micali and Leonid Reyzin. Physically observable cryptography. *IACR Cryptology ePrint Archive*, 2003:120, 2003.
19. Amir Moradi. Statistical tools flavor side-channel collision attacks. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 428–445. Springer Berlin Heidelberg, 2012.
20. NIST. Announcing the Advanced Encryption Standard (AES). Federal Information Processing Standards Publication, n. 197, November 26, 2001.
21. Guilherme Perin, Laurent Imbert, Lionel Torres, and Philippe Maurine. Attacking randomized exponentiations using unsupervised learning. In Emmanuel Prouff, editor, *Constructive Side-Channel Analysis and Secure Design*, Lecture Notes in Computer Science, pages 144–160. Springer International Publishing, 2014.
22. Bruno Robisson and Pascal Manet. Differential behavioral analysis. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems (CHES)*, volume 4727 of *Lecture Notes in Computer Science*, pages 413–426. Springer, 10-13 September 2007.
23. Thomas Roche, Victor Lomné, and Karim Khalfallah. Combined fault and side-channel attack on protected implementations of AES, 2011.
24. Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Josyula R. Rao and Berk Sunar, editors, *CHES*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.
25. François-Xavier Standaert and Cédric Archambeau. Using subspace-based template attacks to compare and combine power and electromagnetic information leakages. In Elisabeth Oswald and Pankaj Rohatgi, editors, *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 411–425. Springer, 2008.
26. François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.
27. Elisabeth Oswald Stefan Mangard and Thomas Popp. *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer Verlag, 2007.
28. Nicolas Veyrat-Charvillon and François-Xavier Standaert. Mutual information analysis: How, When and Why? In Christophe Clavier and Kris Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 429–443. Springer, 2009.
29. Sung-Ming Yen and Marc Joye. Checking before output may not be enough against fault-based cryptanalysis. *IEEE Transactions on Computers*, 49(9):967–970, 2000.