Power Analysis of Hardware Implementations Protected with Secret Sharing

Guido Bertoni*, Joan Daemen*, Nicolas Debande†‡, Thanh-Ha Le†, Michaël Peeters§ and Gilles Van Assche*,
*STMicroelectronics, †Morpho, ‡TELECOM ParisTech and §NXP Semiconductors

***Abstract*—We analyze the security of three-share hardware implementations against differential power analysis and advanced variants such as mutual information analysis. We present dedicated distinguishers that allow to recover secret key bits from any cryptographic primitive that is implemented as a sequence of quadratic functions. Starting from the analytical treatment of such distinguishers and information-theoretic arguments, we derive the success probability and required number of traces in the presence of algorithmic noise. We show that attacks on three-share hardware implementation require a number of traces that scales in the third power of the algorithmic noise variance. Finally, we apply and test our model on KECCAK in a keyed mode.**

***Keywords*-power analysis; quadratic functions; secret sharing schemes; mutual information analysis; Keccak**

I. INTRODUCTION

Protection against side channel attacks is essential for any implementation of a cryptographic primitive that needs to process secret keys and to which an adversary has access. Failing to put in place sufficient countermeasures can lead to the recovery of a secret key or otherwise secret information that allows, e.g., to forge an authentication code or to recover decrypted data. An important class of side channel attacks is differential power analysis (DPA) and its variants, which exploits dependencies between the data being processed and the power consumption or electromagnetic radiation [1], [2].

In a hardware circuit, the implementer cannot always control the exact sequencing of gate switching and a proper countermeasure must take glitches into account. To achieve protection under realistic assumptions, including the presence of glitches, Nikova et al. propose techniques based on secret sharing [3], [4]. In their paper, the authors analyze a three-share implementation of the Noekeon block cipher [5], which provides security against first-order DPA. However, the authors noticed that an advanced variant of DPA, called mutual information analysis (MIA) [6], can reveal the correct key bits if the power consumption is not affected by noise.

In this paper, we analyze the security of three-share hardware implementations against DPA (including MIA) under the presence of noise induced by all the bits being processed. The analysis can be applied to any cryptographic primitive whose round function is a Boolean vector function of algebraic degree two, called *quadratic* in the sequel or is implemented as sequence of quadratic operations. We use the KECCAK function as an example, for which the designers defined protected and unprotected hardware architectures that we can explicitly compare using the techniques presented in this paper [7]. Nevertheless, the treatment actually covers a variety of primitives. First, there are those whose round function is explicitly described in terms of quadratic functions. For instance, the stream cipher Trivium is built as three cyclically-connected shift registers with quadratic feedback [8]. Then, there are primitives making use of small s-boxes. For instance, the s-boxes of Present [9] or Noekeon [5] have algebraic degree 3 (*cubic*) and decomposable into two quadratic functions. Applying secret sharing techniques on each quadratic function requires less shares than on the cubic function [3], [4]. The AES can also be protected using secret sharing, as presented by Moradi et al. [10].

The techniques of this paper share some similarity with those in [11]. Their Zero-Offset 2DPA distinguisher can be applied in the case of hardware implementations using two shares. We adapt it to three shares in Section IV-C, show that three shares concretely provide a masking scheme that provably offers protection for functions implemented with quadratic operations, analytically model the attack success probability, and compare with simulations. In our analysis and simulations we have adopted the Hamming-Distance model [2], applied to the registers. A conclusion of the analysis of this paper is that a three-share implementation does provide security against DPA and MIA. We analytically show that in our model the number of traces needed to distinguish the correct key bits grows with the third power of the noise variance induced by the bits being computed. This suggests that masking with three shares on quadratic functions efficiently provides security that scales similarly as third-order DPA [12], [13], [14]. An analysis taking into account the power consumption of the combinatorial logic, with its occurrence of glitches, is future work.

The paper is organized as follows. Section II gives a model of the power consumption in hardware both for an unprotected and for a protected hardware implementation. Section III proposes a general form of selection function suitable for any quadratic function and different leakage models. Distinguishers against unprotected and protected implementations are presented in Section IV. Finally, Section V applies the selection function and distinguishers on the KECCAK hash function and shows simulations.

II. FROM BITS TO POWER CONSUMPTION

In power (or electromagnetic) analysis, an attacker attempts to retrieve key information by analyzing the power consump-

This work is partially funded by the ANR Saphir 2 project

tion (or electromagnetic radiation) of the device performing the cryptographic primitive. The general set-up is that the attacker has one or more traces of the measured power consumption. DPA exploits the dependence, however weak, of the power consumption on the processed variables by taking the power traces of many executions of the cryptographic primitive with different input values and processing them with statistical methods to retrieve the values of the processed variables [1].

During the last decade, researchers have improved the treatment of traces and one of these enhancement is the so-called correlation power analysis (CPA) [15]. CPA exploits the correlation between the power consumption and processed variables. Later, more advanced ways to measure the distance between distributions were introduced. In particular, MIA does not require a non-zero correlation, but can in principle exploit any dependence of the distribution of the power consumption on the value of variables [6].

There are different types of countermeasure for protecting the cryptographic computation. They range from the transistor level (such as dual rail logic) up to the protocol level (via a careful use of different keys). In this paper we study the effectiveness of a countermeasure at algorithmic level, i.e., that of a secret sharing scheme.

A. Modeling leakage

A hardware circuit consumes power through the activity of its registers, combinatorial logic, wires and auxiliary logic such as the clock. In synchronous CMOS circuits, the data-dependent part of the power consumption is dominated by the dynamic power consumption, particularly the switching activity of the registers and the combinatorial logic [2, Chapter 3]. In this paper we adopt the Hamming distance model limited to registers, with the convention that a register contributes -1 if it flips and $+1$ otherwise. We make abstraction of the dynamic power consumption of the combinatorial logic, and more in particular the effect of *glitches*. Investigation of the latter requires a model that is closer to the implementation than the Hamming distance model but the analysis of the resulting power consumption functions is of similar nature. The same goes for models in which the switching activity of different elements does not have the same weight.

For a given primitive, let K be the fixed secret key under attack and M the input message controlled by the attacker. The activity in each register bit is given by some binary functions $d_i(M,K) \in \mathrm{GF}(2)$, where $d_i(M,K) = 1$ if the bit flips and 0 otherwise. Then, the contribution of all the registers to the power consumption is expressed as:

$$P(M,K) = \sum_i (-1)^{d_i(M,K)}. \qquad (1)$$

Adopting a more complex model will result in a similar expression than Eq. (1), with terms $a_i(-1)^{d_i(M,K)}$, where i ranges over all elements (gates, registers, ...) contributing to the power consumption and the factors a_i are weighing factors.

B. Protection with secret sharing

The secret sharing scheme technique proposed in [3], [4] can be seen as an evolved version of the duplication method that was introduced in [16]. These two methods consist in representing the internal variables x of a primitive (i.e., its state) in a number of shares $a, b, c \ldots$ The sum (typically in $\mathrm{GF}(2)$) of the shares gives the native value of the state: $x = a \oplus b \oplus c \ldots$. Performing computations or storing a single share has a power consumption that is independent of the native value as each share is independent of the native value.

The linear steps of the primitive can be computed on the shares in separate computations as for a linear function f it yields $f(x) = f(a \oplus b \oplus c \ldots) = f(a) \oplus f(b) \oplus f(c) \ldots$ Information-theoretically, the variables processed in each of these calls to f provide no information on the native value of the state. For the non-linear steps, one cannot avoid computations that mix elements from more than one share of the state. However, if one can ensure that no intermediate results are correlated to native state bits, this rules out any form of DPA that requires a correlation between the power consumption and native state bits, such as correlation power analysis (CPA).

In dedicated hardware, however, it is not easy to ensure the absence of such correlations, which can arise from transient states due to glitches in the combinatorial logic [17]. The secret sharing scheme technique proposed in [3], [4] solves this problem essentially by representing the state in n shares such that any computation involves at most $n-1$ shares. The required number of shares depends on the algebraic degree of the nonlinear step to be computed, and for quadratic functions 3 shares are usually sufficient (see the exact conditions in [3], [4]).

C. Modeling leakage with three shares

The implementation of quadratic functions can be protected using three shares. Each bit of the state is therefore stored as the parity of three registers, a so-called *register triplet*, i.e., the native value 0 can be represented as 000, 011, 101 or 110, and 1 as 111, 100, 010 or 001. The value of two of the three registers is generated randomly each time the primitive is evaluated, hence for each measurement, and the power consumption becomes a stochastic variable to the adversary.

Considering the power consumed by the changes in registers, we define $d_i(M,K)$ as in Section II-A, where $d_i(M,K) = 1$ if the native value of the bit i changes and 0 otherwise. The impact on the power consumption of the register triplets is as follows.

- $d_i(M,K) = 0$: either no register flips, contributing $+3$, or 2 of the 3 registers flip, contributing -1. The contribution is a stochastic variable that has value 3 with probability $1/4$ and -1 with probability $3/4$. We denote this variable by T_0 and its distribution by $T_0(t)$.
- $d_i(M,K) = 1$: either all 3 registers flip or 1 of the 3 registers flip. The contribution is a stochastic variable that has value -3 with probability $1/4$ and 1 with probability $3/4$. We denote it by T_1 and its distribution by $T_1(t)$.

Both $T_0(t)$ and $T_1(t)$ have mean 0 and variance 3.

The power consumption is the sum of the stochastic variables determined by the bits of d:

$$P(M,K) = \sum_i T_{d_i(M,K)}. \qquad (2)$$

As opposed to an unprotected implementation, the total power consumption is not a deterministic function of d, but a stochastic variable with a probability density function that is determined by d. Its probability distribution is obtained by convoluting the distributions $T_{d_i(M,K)}(t)$ over all indices i.

Interestingly, the value of a bit of $d_i(M,K)$ does not affect the mean of the probability density function of $P(M,K)$, hence precluding CPA. Moreover, as T_0 and T_1 have the same variance, it does not even affect the variance of $P(M,K)$.

III. Selection function for quadratic functions

We describe a general way to build a suitable selection function for the implementation of a quadratic function. We assume that the primitive is implemented as a sequence of quadratic operations, although a slightly more general set of functions is covered. This applies not only to primitives making use of a quadratic round function but also for round functions that can suitably be implemented as the composition of quadratic functions. Let K be the fixed secret key under attack and M the input message controlled by the attacker. Let the activity we target be denoted as d, gathering b binary functions $d_i(M,K)$, one for each output bit i. The function $d_i(M,K)$ has the following form in $\mathrm{GF}(2)$:

$$d_i(M,K) = \alpha_i(M) + \beta_i(K) + K^{\mathrm{T}}\Gamma_i M. \tag{3}$$

Here, $\alpha_i(M)$ and $\beta_i(K)$ are any functions of M and K only, respectively, while Γ_i is a binary matrix connecting the bits of K and M in the bilinear form $K^{\mathrm{T}}\Gamma_i M$. Note that Eq. (3) encompasses not only all quadratic functions but also some more general functions, as the degree of $\alpha_i(M)$ and of $\beta_i(K)$ is not limited.

The function d can model the switching activity of a register with a quadratic updating function (Hamming distance model) but also consumption imbalance in (quadratic) combinatorial logic (Hamming weight model).

We will consider in this paper attacks based on the consumption of a single register at a time. We do not expect attacks addressing multiple registers to give better success probabilities. It may result in a marginal decrease of algorithmic noise but at the same time it increases the number of hypothesis exponentially.

The bit i we use to partition the traces at a given time is called the *focus point*. Our goal is to build a selection function for the focus point that splits measurements into two sets M_0 and M_1, whose distributions can be set apart when the hypothesis on key bits is correct. The key point is that the distributions of M_0 and M_1 differ iff the correct hypothesis is set.

For the activity function in Eq. (3), the selection function is defined as

$$s_i(M,K) = \alpha_i(M) + K^{\mathrm{T}}\Gamma_i M.$$

Compared to d, we remove in s any fixed (unknown) contribution of the key. The influence of the key goes only through $K^{\mathrm{T}}\Gamma_i M$. The hypothesis on K need to be set only within a subset of dimension $\operatorname{rank}\Gamma_i$. For instance, let U_i be an invertible matrix such that $\mathrm{U}_i^{\mathrm{T}}\Gamma_i$ has only $\operatorname{rank}\Gamma_i$ non-zero rows, and let $K = \mathrm{U}_i\kappa$. Then, the hypothesis is only on the bits κ_j of the non-zero rows of $\mathrm{U}_i^{\mathrm{T}}\Gamma_i$.

Let the hypothesis on the key K^* be related to the correct key K as $K = K^* + \epsilon$. Then, $s_i(M, K+\epsilon) = s_i(M,K) + \epsilon^{\mathrm{T}}\Gamma_i M$. When $\epsilon \in \ker\Gamma_i^{\mathrm{T}}$, then $s_i(M,K^*)$ differs from $d_i(M,K)$ only by a constant (unknown) term. Otherwise, if $\epsilon \notin \ker\Gamma_i^{\mathrm{T}}$ and at the condition that $\epsilon^{\mathrm{T}}\Gamma_i M$ is balanced over $\mathrm{GF}(2)$ when M is seen as a random variable, then $s_i(M,K^*)$ behaves as a random variable that is independent from $d_i(M,K)$. This is the case, for instance, if M is uniformly distributed over its underlying set.

IV. Distinguishers

We start with using the Kullback-Leibler divergence to determine a lower bound on the number of measurements needed to distinguish two distributions. Then, we see how to distinguish the distributions in the cases of the unprotected and three-share implementations.

A. Kullback-Leibler divergence as a reference

The number of samples that are needed to distinguish between one distribution f over another g with some given success probability is inversely proportional to the Kullback-Leibler divergence $D(f\|g)$ between the two distributions [18], with

$$D(f\|g) = \int f(t)(\log(f(t)) - \log(g(t))dt.$$

This measure allows estimating the required number of samples for successfully distinguishing the correct hypothesis over the wrong ones. We model the distributions that correspond to the different hypotheses as a two-dimensional function of t and s, with t the value of the power consumption and s the value of $s_i(M,K^*)$. Let us denote them by $f^\star(t,s)$ for the correct hypothesis and $f^\diamond(t,s)$ for the incorrect hypotheses. We assume the distributions of all incorrect hypotheses are equal. We then have:

$$D(f^\star(t,s)\|f^\diamond(t,s)) = \sum_s \int f^\star(t,s)\log\left(\frac{f^\star(t,s)}{f^\diamond(t,s)}\right)dt. \tag{4}$$

If we assume that $\Pr(s_i(M,K)=0|M) = \Pr(s_i(M,K)=1|M) = \frac{1}{2}$, we can express $f^\star(t,s)$ as $f^\star(t,s) = \frac{1}{2}f^\star_s(t)$ with $f^\star_0(t)$ and $f^\star_1(t)$ the distributions of the power consumption for $s=0$ and $s=1$ respectively. If we further assume for the incorrect hypothesis that the distribution is independent from s, we have $f^\diamond(t,s) = \frac{1}{2}f^\diamond(t)$. This allows us to simplify Eq. (4):

$$\begin{aligned} D(f^\star\|f^\diamond) =& \frac{1}{2}\int f^\star_0(t)\log\left(\frac{f^\star_0(t)}{f^\diamond(t)}\right) \\ &+ \frac{1}{2}\int f^\star_1(t)\log\left(\frac{f^\star_1(t)}{f^\diamond(t)}\right)dt. \end{aligned}$$

Additionally, if we have a symmetry condition such that $f^\star_1(t) = f^\star_0(-t)$ and $f^\diamond(t) = f^\diamond(-t)$, we can further simplify this into:

$$D(f^\star\|f^\diamond) = \int f^\star_0(t)\log\left(\frac{f^\star_0(t)}{f^\diamond(t)}\right)dt. \tag{5}$$

B. *Distinguishing for unprotected implementations*

We show how to distinguish the measurements in the case of a correct hypothesis from those in the case of an incorrect one. For this, we assume that the power consumption follows the model in Section II-A and in particular Eq. (1). Depending on the exact form of the functions d_i, there are ways to select the messages M such that $d_{i'}(M, K)$ is perfectly balanced. However, to simplify the discussion, we assume that the messages M are randomly selected.

For a given focus point i and hypothesis K^*, we gather in sets M_0 and M_1 the power measurements $P(M, K)$ for which $s_i(M, K^*)$ is 0 or 1, respectively. In each of these sets, the noise is only algorithmic and its amplitude depends on the number b of bits computed simultaneously. We study the influence of this noise on the success probability of an attack on an unprotected implementation.

If $s_i(M, K^*)$ is independent from $d_{i'}(M, K)$ with $i' \neq i$, the distribution of the power consumption is the convolution of distributions with peaks at -1 and $+1$. This leads to a binomial distribution that is close to a normal distribution with variance b thanks to the central limit theorem. For the distribution within the sets M_0 and M_1, we distinguish two cases:

- For the correct hypothesis, $P(M, K)$ knowing $s_i(M, K)$ has mean ± 1 and variance $b - 1$. So $f_s^\star(t) \approx \mathcal{N}_{((-1)^{(s+\delta)};b-1)}(t)$, where δ is fixed due to the (unknown) difference between s_i and d_i.
- For the incorrect hypotheses, $P(M, K)$ is independent of $s_i(M, K^*)$ and $f^\diamond(t) \approx \mathcal{N}_{(0;b)}(t)$.

The goal of our attack on the plain core is to build an understanding on the influence of noise on the success probability and to compare later to the three-share core. Therefore, we will consider an attack where a (passive) attacker is provided traces for randomly chosen message blocks. This random process introduces noise that is easy to model. We are aware that in our model of the plain core one can conduct a noiseless attack but this would set the wrong target for comparing with the three-share core, where noise is unavoidable.

If the bits $q_{(x',y',z')}$ are independent from each other, each one contributes to the distribution by convoluting it with a distribution with a impulse at -1 and one at 1. This leads to a binomial distribution that is very close to a normal distribution with variance $b - 1$ thanks to the central limit theorem.

The distributions $f^\star$ and $f^\diamond$ satisfy the assumptions of Section IV-A, so Eq. (5) yields:

$$D(f^\star \| f^\diamond) = \int \mathcal{N}_{(1;b-1)}(t) \log\left(\frac{\mathcal{N}_{(1;b-1)}(t)}{\mathcal{N}_{(0;b)}(t)}\right) dt.$$

If we approximate $b - 1$ by b, this can be solved analytically resulting in $D(f^\star \| f^\diamond) = 1/2b$. So the required number of samples for a given success rate is proportional to b.

Looking at $f^\star$ and $f^\diamond$, it appears clearly that we can distinguish the correct hypothesis from the incorrect one(s) based on the distance between the averages of the measured power consumptions in M_0 and M_1. We choose as correct hypothesis the one that maximizes this distance (in absolute value). This is known as *difference of means* (DoM), where $\mathbb{E}$ denotes the expected value:

$$\begin{aligned}\Delta_{\text{DoM}}(K^*) = &|\mathbb{E}[P(M, K)|s_i(M, K^*) = 0] \\ &- \mathbb{E}[P(M, K)|s_i(M, K^*) = 1]|.\end{aligned}$$

If we denote the number of traces by $|M|$ and assume M_0 and M_1 have the same size, the average power consumption over M_0 and over M_1 have an expected variance of $2(b - 1)/|M|$ and $2b/|M|$ for the correct and incorrect hypotheses, respectively. The difference of the means is $2(-1)^\delta$, and the variance is the sum of the variances. Taking the absolute value folds the distribution around the y-axis and discards the unknown sign $(-1)^\delta$. This results in the following distributions for $t \geq 0$ (and 0 otherwise):

$$\begin{aligned}f_{\text{DoM}}^\star(t) &= \mathcal{N}_{(2;4(b-1)/|M|)}(t) + \mathcal{N}_{(-2;4(b-1)/|M|)}(t) \\ f_{\text{DoM}}^\diamond(t) &= 2\mathcal{N}_{(0;4b/|M|)}(t).\end{aligned}$$

These distributions (assuming $b - 1 \approx b$) are illustrated in Figure 1.

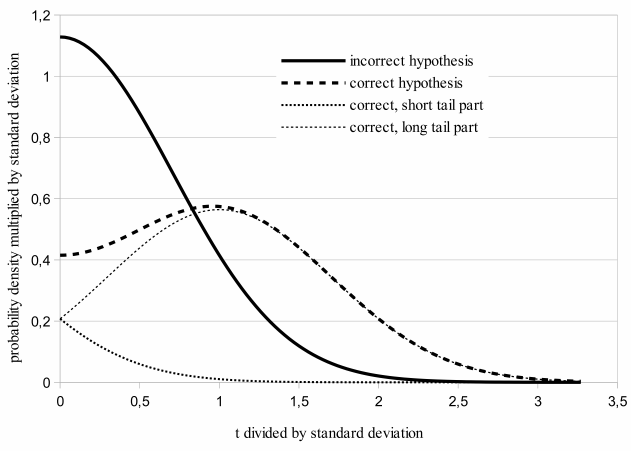

Fig. 1. Distributions $f_{\text{DoM}}^\star(t)$ and $f_{\text{DoM}}^\diamond(t)$ (or $f_{\text{DoA}}^\star(t)$ and $f_{\text{DoA}}^\diamond(t)$).

The probability of success can now readily be computed from these distributions. It is the probability that a variable chosen according to $f_{\text{DoM}}^\star(x)$ is larger than $h - 1$ values chosen according to $f_{\text{DoM}}^\diamond(x)$, with $h = 2^{\operatorname{rank}\Gamma_i}$ the number of hypotheses considered. This is given by:

$$P_{\text{success}} = \int_0^\infty \left(\int_0^t f_{\text{DoM}}^\diamond(y)dy\right)^{h-1} f_{\text{DoM}}^\star(t)dt.$$

Let G_h be the function defined as

$$G_h(\sigma^2) = \int_0^\infty \left(\operatorname{erf}\left(\frac{t}{\sqrt{2}\sigma}\right)\right)^{h-1} \left(\mathcal{N}_{(-1;\sigma^2)}(t) + \mathcal{N}_{(1;\sigma^2)}(t)\right) dt.$$

The expression $\int_0^t f_{\text{DoM}}^\diamond(y)dy = 2\int_0^t \mathcal{N}_{(0;4b/|M|)}(y)dy$ can be expressed in terms of the error function as follows: $\operatorname{erf}(t/\sqrt{8b/|M|})$. If we approximate $b - 1$ by b, we see the success probability is fully determined by the ratio $b/|M|$, and $P_{\text{success}} = G_h(b/|M|)$. Using $D(f^\star \| f^\diamond) = 1/2b$ we can express it as a function of the Kullback-Leibler divergence, yielding:

$$P_{\text{success}} = G_h\left(\frac{1}{2D(f^\star \| f^\diamond)|M|}\right) \tag{6}$$

C. Distinguishing for three-share implementations

We assume a power consumption given by Eq. (2) in Section II-C. Hence we have:

$$\begin{aligned} f_s^\star(t) &= T_{s+\delta}(t) \otimes \mathcal{N}_{(0;3(b-1))}(t) \\ &= \frac{1}{4}\mathcal{N}_{(-3(-1)^\delta;3(b-1))}(t) + \frac{3}{4}\mathcal{N}_{((-1)^\delta;3(b-1))}(t), \text{ and} \\ f^\diamond(t) &= \mathcal{N}_{(0;3b)}(t), \end{aligned}$$

with δ the (unknown) fixed difference between s_i and d_i. We can fill this in Eq. (5) to compute the Kullback-Leibler distance. We evaluated this expression numerically and for large values of b it converges to $D(f^\star \| f^\diamond) = 1/(9b^3)$. So the required number of traces is proportional to the 3rd power of the register size.

T_0 and T_1 start to differ only from their third statistical moment, called the *coefficient of asymmetry* (or skewness) [19]. We call the distinguisher based on this the Difference of coefficient of Asymmetry (DoA):

$$\begin{aligned} \Delta_{\text{DoA}}(K^*) = &|\mathbb{E}[P(M,K)^3 | s_i(M,K^*) = 0] \\ &- \mathbb{E}[P(M,K)^3 | s_i(M,K^*) = 1]|. \end{aligned}$$

Note that the DoA can also be seen as the Zero-Offset 2DPA distinguisher of [11] adapted to the third order.

For a normal distribution $\mathcal{N}(\mu;\sigma^2)$, its third moment is $\mu^3 + 3\mu\sigma^2$ [20]. So, for the correct hypothesis, the power measurements in M_s have $\mathbb{E}[P^3] = 6(-1)^{s+\delta}$ and taking the difference gives $\mathbb{E}[\Delta_{\text{DoA}}] = 12(-1)^\delta$. For the incorrect hypotheses, the third moment is zero.

The variance on the skewness $\mathbb{E}[P^3]$ for $|M|$ samples of a normal distribution with variance σ^2 is given by $6\sigma^6/|M|$ [19]. So assuming $b \gg 1$, we have $\sigma^2(\Delta_{\text{DoA}}) \approx 24(3b)^3/|M|$. Using the central limit theorem and then taking the absolute value, we write for $t \geq 0$:

$$\begin{aligned} f_{\text{DoA}}^\star(t) &= \mathcal{N}_{(12;24(3b)^3/|M|)}(t) + \mathcal{N}_{(-12;24(3b)^3/|M|)}(t) \\ f_{\text{DoA}}^\diamond(t) &= 2\mathcal{N}_{(0;24(3b)^3/|M|)}(t). \end{aligned}$$

Figure 1 illustrates also these distributions. Following the same reasoning as in Section IV-B yields $P_{\text{success}} = G_h(9b^3/2|M|)$. Using $D(f^\star \| f^\diamond) = 1/(9b^3)$, we can again express the success probability as a function of the Kullback-Leibler divergence. Remarkably, this leads to the same expression as in the case of DoM, i.e., Equation (6).

D. Information theoretic distinguishers

The mutual information of two discrete variables X and Y is $I(X;Y) = H(X) - H(X|Y)$, with H denoting the Shannon entropy [18]. Informally, the mutual information $I(X;Y)$ is the amount of information (in bits) the variables X and Y have in common. When two variables are independent, the mutual information is zero.

We can compute the mutual information between the variables $P(M,K)$ and $s_i(M,K^*)$, for a given hypothesis K^*. For the incorrect hypothesis, $f^\diamond(t,0) = f^\diamond(t,1)$, therefore the two variables are independent and $I(P,s_i) = 0$. For the correct hypothesis, $f^\star(t,0) \neq f^\star(t,1)$ and the mutual information is strictly positive.

This makes the mutual information a candidate distinguisher. One needs to estimate or approximate the joint distribution of $P(M,K)$ and $s_i(M,K^*)$ using the measurements. This leads to a number of variants (see also, e.g., [21]). Most of MIA estimators are in the probability density function (pdf) estimators family. Another family groups methods that use statistics. In the pdf estimators family, the methods are either non-parametric methods or parametric methods. Histogram MIA is a non-parametric method and does not need any preliminary knowledge on the distributions. Parametric methods assume particular distributions, e.g., the Gaussian parametric method assumes the distributions are normal. MIA statistics methods use either statistical tests such as computing a distance or statistical tools such as computing statistical moments. An example of the former is the Kolmogorov-Smirnov distance and an example of the latter is cumulant MIA. There are also extensions of MIA from univariate to multivariate distributions [22].

The method of *cumulant MIA* aims at estimating mutual information by using high-order cumulants. This method was described in [23], where an Edgeworth expansion [20] was used to estimate mutual information. Applied to $I(P(M,K);Z)$ with $Z = (-1)^{s_i(M,K^*)}$, this becomes

$$\begin{aligned} I(P;Z) \approx &\frac{1}{4}\mathbb{E}[P.Z]^2 + \frac{1}{12}(\mathbb{E}[P^2.Z]^2 + \mathbb{E}[P.Z^2]^2) \\ &+ \frac{1}{48}(4(\mathbb{E}[P^3.Z] - 3\mathbb{E}[P^2]\mathbb{E}[P.Z])^2 \\ &+ 6(\mathbb{E}[P^2.Z^2] - \mathbb{E}[P^2]\mathbb{E}[Z^2] - 2\mathbb{E}[P.Z]^2)^2 \\ &+ 4(\mathbb{E}[P.Z^3] - 3\mathbb{E}[P.Z]\mathbb{E}[Z^2])^2). \end{aligned} \tag{7}$$

The attacker can evaluate this expression for the different hypotheses by computing the different terms based on the measurements. This estimation allows to analyze the different dependencies separately, which is not the case with other approaches. Indeed, an attacker can determine which components of the mutual information estimation give the most useful information.

In the case of the three-share implementation, we see that most terms in Eq. (7) disappear. First, $\mathbb{E}[P.Z^i] = 0$ for any i as the mean of P is zero over both M_1 and M_0. Then, assuming $|M_0| = |M_1|$, the term $\mathbb{E}[P^2.Z]$ is also zero, as the variance of P is the same over both M_0 and M_1. Finally, as $Z^2 = 1$, the terms $\mathbb{E}[P^2.Z^2]$ and $\mathbb{E}[P^2]\mathbb{E}[Z^2]$ are equal and cancel each other out. This results in $I(P;Z) \approx \frac{1}{12}\mathbb{E}[P^3.Z] = \frac{1}{6}\Delta_{\text{DoA}}$.

V. KECCAK

KECCAK is a function with variable-length input and arbitrary-length output based on the sponge construction [24]. In this construction, a permutation f with b input/output bits is iterated. First, the input padded and its blocks are *absorbed* sequentially into the state, with a simple XOR operation. Then, the output is *squeezed* from the state block by block. The size of the blocks is denoted by r and called the *bitrate*. The remaining number of bits $c = b - r$ is called the *capacity* and determines the security level of the function.

The simplest use case of a sponge function is to use it as a hash function by simply truncating the output. A MAC function can be built by taking as input the concatenation of a secret key and a message. For using it as a stream cipher, it suffices to input the secret key and a nonce and using the output as a key stream. More modes of use are described in [25] and [24].

Seven permutations, denoted KECCAK-$f[b]$, are defined with width $b = 25w$ ranging from 25 to 1600 bits, increasing in powers of two. The state of KECCAK-$f[b]$ is organized as a set of $5 \times 5 \times w$ bits with (x, y, z) coordinates. The coordinates are always considered modulo 5 for x and y and modulo w for z. Coordinates are taken modulo 5 for x and y and modulo w for z. A *row* is a set of 5 bits with given (y, z) coordinates, a *column* is a set of 5 bits with given (x, z) coordinates and a *slice* is a set of 25 bits with given z coordinate. A *row* is a set of 5 bits with given (y, z) coordinates and a *column* is a set of 5 bits with given (x, z) coordinates.

The round function of KECCAK-$f[b]$ consists of the following steps, which are only briefly summarized here. For more details, we refer to the specifications [25].

- θ is a linear mixing layer that adds a pattern depending solely on the parity of the columns of the state.
- ρ and π displace bits without altering their value. Jointly, their effect is denoted by $(x', y', z') \xrightarrow{\pi\circ\rho} (x, y, z)$, with (x', y', z') a bit position before ρ and π and (x, y, z) its coordinates afterward.
- χ is a degree-2 non-linear mapping that processes each row independently. It can be seen as the application of a translation-invariant 5-bit quadratic S-box:

$$a_{(x,y,z)} \leftarrow a_{(x,y,z)} + (a_{(x+1,y,z)} + 1)a_{(x+2,y,z)}.$$

- ι adds a round constant.

A. The plain core

The *plain core* architecture instantiates the KECCAK-f round function using combinatorial logic and keeps track of the state in a register [25]. At each clock cycle, the state value in the register is updated by applying the round function to it. Applying the permutation simply consists in performing as many clock cycles as there are rounds in KECCAK-f. The absorbing of message blocks is implemented by an XOR stage at the input of the round function logic, which takes its input from a buffer. The round constants are handled by a simple finite state machine.

B. The three-share core

The *three-share core* architecture implements the secret sharing scheme technique presented in Section II-B to offer protection against CPA. The designers of KECCAK show that three shares are sufficient thanks to the low degree of the nonlinear step in the KECCAK-f round function χ [7]. Their architecture is a rather straightforward application of the secret sharing scheme technique to the plain core: it keeps the three shares of the state in separate registers and instantiates the three-share version of the KECCAK-f round function in combinatorial logic. The linear layer λ is instantiated three times, operating on each share separately. It implements the nonlinear step χ by three separate logic blocks each implementing a function $a = \chi'(b, c)$ defined by:

$$\begin{aligned} a_{(x,y,z)} \leftarrow & b_{(x,y,z)} + (b_{(x+1,y,z)} + 1)b_{(x+2,y,z)} \\ & + b_{(x+1,y,z)}c_{(x+2,y,z)} + c_{(x+1,y,z)}b_{(x+2,y,z)} \ . \end{aligned}$$

Each block computes χ' for a share taking as input the two other shares. The message blocks are applied to a single share at the input of the round logic and the round constants are applied to a single share at the output of the round logic.

The three-share core is illustrated in Figure 2.

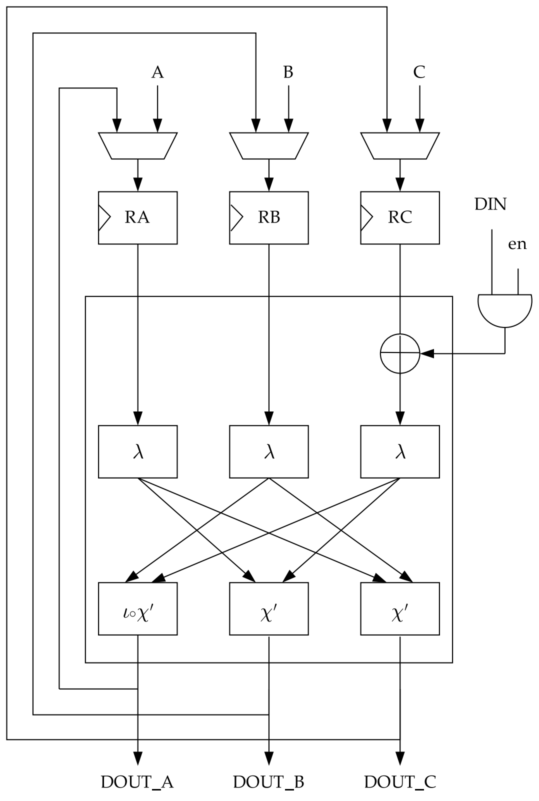

Fig. 2. The three-share core

Before processing, the three shares are generated from a random source. As the initial state of KECCAK is equal to zero, two of the three shares can be generated randomly and the third is computed as the sum (in GF(2)) of the other two. Before processing, the three shares are initialized consistent with the zero value: two of the three shares are generated randomly and the third is computed as their XOR.

C. The attack point

A DPA attack on a KECCAK core consists in making it run repeatedly, taking each time the same secret key K

and a chosen (or known) input block M, and recording the power computation in a so-called *trace*. The traces can be identified by the corresponding value of M and we can use the measured power consumption $P(M, K)$ to determine the unknown secret value K.

In typical cases where a sponge function is used in combination with a secret key, the input is prefixed with the key. The input of the KECCAK-f permutation under attack is $K + M$, with K the state of the sponge function after absorbing the key. The knowledge of K is sufficient to, e.g., forge MACs or produce key streams with arbitrary nonces. This setting corresponds with the case that the key consists of (or is padded as) exactly one block and the attacker targets $K = \text{KECCAK-}f(\text{Key}||0^c)$ with $\text{Key} \in \text{GF}(2)^r$. After retrieving K, the attacker can recover the absorbed key by inverting KECCAK-f. The second block brings message bits and $K + M$ is the input to KECCAK-f, with M spanning the first r bits.

A key that is shorter than the rate would give rise to a slightly different setting, in which the key bits and message bits sit together in a first block $\text{Key}||M||0^c$, with M spanning $r - |\text{Key}|$ bits. We do not consider this setting in the remainder of this section, but its analysis is very similar to the one we present.

Both in the plain core and in the three-share core, the output of the round function is stored in registers. The round function of KECCAK-f is quadratic, hence we can apply the definitions of Section III and target the power consumed when storing the result in the registers. Subsequent rounds would make the dependencies between key and message bits more difficult to deal with.

Let B be the output of the first round, $\kappa = \pi(\rho(\theta(K)))$ and $\mu = \pi(\rho(\theta(M)))$. Then the output bit at coordinates (x, y, z) is

$$B_{(x,y,z)} = \text{RC}_{(x,y,z)} + \kappa_{(x,y,z)} + \mu_{(x,y,z)} + (\kappa_{(x+1,y,z)} + \mu_{(x+1,y,z)} + 1)(\kappa_{(x+2,y,z)} + \mu_{(x+2,y,z)}).$$

The function describing the register activity is $d = K + B$ and the selection function is derived as in Section III:

$$s_{(x,y,z)}(M, \kappa^*{}_{(x+1,y,z)}, \kappa^*{}_{(x+2,y,z)}) = \quad (8)$$

$$\mu_{(x,y,z)} + \mu_{(x+2,y,z)} + \mu_{(x+1,y,z)}\mu_{(x+2,y,z)} \quad (9)$$

$$+ \kappa^*{}_{(x+1,y,z)}\mu_{(x+2,y,z)} + \kappa^*{}_{(x+2,y,z)}\mu_{(x+1,y,z)}. \quad (10)$$

If the hypothesis on $\kappa^*{}_{(x+1,y,z)}$ and $\kappa^*{}_{(x+2,y,z)}$ is correct, then $s_{(x,y,z)}(M, \kappa^*{}_{(x+1,y,z)}, \kappa^*{}_{(x+2,y,z)})$ differs from $d_{(x,y,z)}(M, K)$ by a fixed (unknown) term. However, if the hypothesis is incorrect, $(\epsilon_{(x+1,y,z)}, \epsilon_{(x+2,y,z)}) \neq (0, 0)$, then $s_{(x,y,z)}(M, \kappa^*{}_{(x+1,y,z)}, \kappa^*{}_{(x+2,y,z)})$ is independent from $d_{(x,y,z)}(M, K)$.

Regarding the independence of the activity function between different coordinates, their sum $s_{(x_1,y_1,z_1)}(M, \kappa) + s_{(x_2,y_2,z_2)}(M, \kappa)$ is not necessarily balanced for all pairs $(x_1, y_1, z_1) \neq (x_2, y_2, z_2)$ because M is zero in the inner part of the state. However, we can prove the following theorem.

Theorem Let $(x'_i, y'_i, z'_i) \xrightarrow{\pi\circ\rho} (x_i, y_i, z_i)$ for $i \in \{1, 2\}$. Then $s_{(x_1,y_1,z_1)}(M, \kappa)$ and $s_{(x_2,y_2,z_2)}(M, \kappa)$ are independent unless $(x'_1, z'_1) = (x'_2, z'_2)$ and $(5y_i + x_i)w + z_i \geq r$ for $i \in \{1, 2\}$ (i.e., the bits come from the same column and both from the last c bits).

Proof: Two Boolean functions are independent if their bitwise sum is balanced, so we need to prove that the function $s_{(x_1,y_1,z_1)}(M, \kappa) + s_{(x_2,y_2,z_2)}(M, \kappa)$ is balanced if the conditions of the theorem are satisfied. Let us trace the bits of M to those of μ throught $\lambda = \pi \circ \rho \circ \theta$. The bits of M are balanced in the outer part of the state ($(5y'_i + x'_i)w + z'_i < r$) and zero in the inner part of the state. Let $\mu' = \theta(M)$. If the rate is not below $5w$, all bits of μ' are balanced. However, in the outer part of the state, bits in the same column are equal due to the fact that θ treats the bits of a column in the same way. So the bitwise sum of two such bits will be zero. The bit transposition $\pi \circ \rho$ just moves bits to other positions, realizing the mapping from (x'_i, y'_i, z'_i) to (x_i, y_i, z_i) specified in the theorem. If (x'_1, y'_1, z'_1) and (x'_2, y'_2, z'_2) are in different columns or not both in the outer part, the function $s_{(x_1,y_1,z_1)}(M, \kappa) + s_{(x_2,y_2,z_2)}(M, \kappa)$ will exhibit two balanced terms that do not cancel out. It may be that (x_1, y_1, z_1) is equal to $(x_2 + 1, y_2, z_2)$ or $(x_2 + 2, y_2, z_2)$, possibly canceling out terms. In that case the balanced term $\mu_{(x_2,y_2,z_2)}$ remains, as (x_2, y_2, z_2) can then not be equal to $(x_1 + 1, y_1, z_1)$ or $(x_1 + 2, y_1, z_1)$. In a similar way, equality of (x_2, y_2, z_2) to $(x_1 + 1, y_1, z_1)$ or $(x_1 + 2, y_1, z_1)$ leaves $\mu_{(x_1,y_1,z_1)}$ as a balanced term. ■

For a focus point not covered by this theorem, there may be other activity bits correlated with the selection function. The contribution of these bits will affect the DoM values for M_0 and M_1, both for the correct and the incorrect key hypotheses and hence impact the success probability.

For focus points covered by this theorem, there may be pairs of bit positions (x_1, y_1, z_1) and (x_2, y_2, z_2) with activity functions that are correlated and their contributions to the variance is not independent. It is easy to see that if the correlation between the bits is c, their sum contributes $2(1+c)$ to the variance instead of 2. Making the plausible assumption that the sign is (near)-balanced over all non-zero correlation, allows us to predict the total variance by $b - 1$ for the correct and b for the incorrect hypotheses.

D. *Experimental results for the plain core*

We performed experiments for the DoM distinguisher on KECCAK variants based on all seven KECCAK-f versions, namely $\text{KECCAK}[r = 16 \times 2^\ell, c = 9 \times 2^\ell]$. The rate ranges from 1024 bits for KECCAK-f[1600] down to 16 bits for KECCAK-f[25]. The secret value to recover is as in Section V-C.

The experiment assumes that the power consumed is equal to the number of bits that flip in the b-bit register that initially contains the output of the previous call to KECCAK-f and then the state after the first round. For each width of KECCAK-f, we conducted distinguishing experiments for 1000 different secret key values and for each key value we took traces for a large number of r-bit message blocks. The focus point in this experiment is $(0, 0, 0)$, which is not in the last c bits (see Theorem in Section V-C).

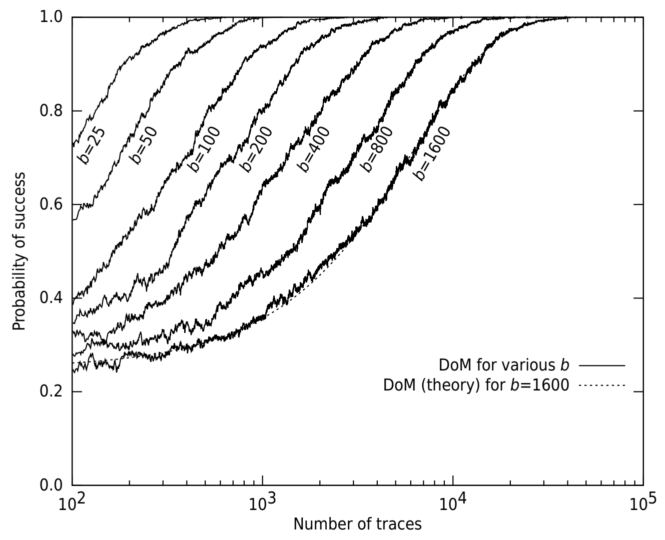

Fig. 3. Success rate for the experiments on the plain core with DoM.

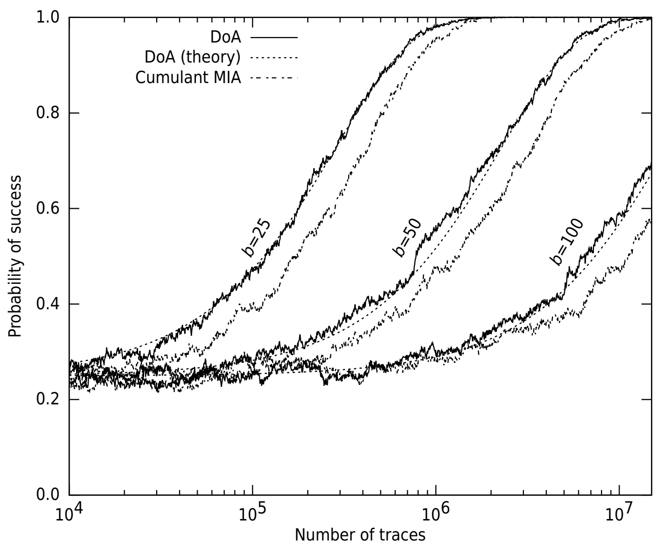

Fig. 4. Success rate for the experiments on the three-share core with cumulant MIA and DoA.

Figure 3 reports on the outcome of our experiments. It plots the success rate, being the ratio of correctly selected hypotheses among all keys, as a function of the number of traces. It also plots the theoretical success probability $G_h(b/|M|)$, obtained in Section IV-B for KECCAK-f[1600], which the experimental results follow closely. The success probability starts at $1/h = \frac{1}{4}$, as the correct hypothesis could be any value, and then grows very close to 1 when $|M|$ reaches $20b$. It follows the scaling suggested by the Kullback-Leibler distance.

E. Experimental results for the three-share core

We performed experiments for the DoA distinguisher on KECCAK$[r = 16, c = 9]$, KECCAK$[r = 32, c = 18]$ and KECCAK$[r = 64, c = 36]$. These toy primitives already require a high number of measurements when protected by three shares and can give an idea of the adequacy of the model and on the scaling. We again assume that the power consumed is equal to the number of bits that flip in the registers, except that the register now contains $3b$ bits. For each instance, we conducted distinguishing experiments for 1000 different secret key values and for each key value we took traces for large numbers of message blocks. The focus point is again $(0, 0, 0)$.

Figure 4 shows the success rate as a function of the number of traces for KECCAK$[r = 16, c = 9]$, KECCAK$[r = 32, c = 18]$ and KECCAK$[r = 64, c = 36]$, both for DoA and cumulant MIA. Clearly, DoA is more efficient than cumulant MIA, confirming the intuition behind the DoA distinguisher.

The figure also plots the theoretical success probability $G_h(9b^3/(2|M|))$ that is confirmed by the experimentally obtained values for DoA. In this case too it follows the scaling suggested by the Kullback-Leibler distance.

These simulations are for toy versions of KECCAK-f. For instances actually usable in practice, e.g., from lightweight KECCAK-f[200] till KECCAK-f[1600] used in SHA-3, the scaling suggests a tremendous number of traces (from billions to hundreds of billions) needed to recover secret key bits.

VI. CONCLUSIONS

We presented selection functions and distinguishers dedicated to protected and unprotected hardware implementations, for any cryptographic primitive implemented as a sequence of quadratic functions. We analyzed such implementations under the presence of algorithmic noise, in particular in terms of required number of traces and success probabilities.

This analysis shows that a three-share implementation does provide security against DPA and MIA. The number of traces needed to distinguish the correct key bits grows with the third power of the algorithmic noise variance, i.e., induced by the bits being computed. This suggests that masking with a secret-sharing scheme on quadratic functions efficiently provides security that scales similarly as third-order DPA. This applies readily to the three-share architecture proposed in [7].

Our power model does not deal with glitches in the combinatorial logic. They introduce second-order effects [3], [4], although we expect them to be of lower amplitude than the register switching activity. An extension of this work includes the analysis of their effect on the success probability and adapting the results of Section IV to this seems straightforward.

REFERENCES

[1] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology – Crypto '99*, ser. Lecture Notes in Computer Science, M. Wiener, Ed., vol. 1666. Springer, 1999, pp. 388–397.

[2] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks — Revealing the Secrets of Smartcards*. Springer-Verlag, 2007.

[3] S. Nikova, V. Rijmen, and M. Schläffer, "Secure hardware implementation of nonlinear functions in the presence of glitches," in *ICISC*, ser. Lecture Notes in Computer Science, P. J. Lee and J. H. Cheon, Eds., vol. 5461. Springer, 2008, pp. 218–234.

[4] ——, "Secure hardware implementation of nonlinear functions in the presence of glitches," *J. Cryptology*, vol. 24, no. 2, pp. 292–321, 2011.

[5] J. Daemen, M. Peeters, G. V. Assche, and V. Rijmen, "Nessie proposal: the block cipher Noekeon," Nessie submission, 2000.

[6] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel, "Mutual information analysis," in *CHES*, ser. Lecture Notes in Computer Science, E. Oswald and P. Rohatgi, Eds., vol. 5154. Springer, 2008, pp. 426–442.

[7] G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "Building power analysis resistant implementations of Keccak," Second SHA-3 candidate conference, August 2010.

[8] C. D. Cannière, "Trivium: A stream cipher construction inspired by block cipher design principles," in *ISC*, ser. Lecture Notes in Computer Science, S. K. Katsikas, J. Lopez, M. Backes, S. Gritzalis, and B. Preneel, Eds., vol. 4176. Springer, 2006, pp. 171–186.

[9] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," in *CHES*, ser. Lecture Notes in Computer Science, P. Paillier and I. Verbauwhede, Eds., vol. 4727. Springer, 2007, pp. 450–466.

[10] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: A very compact and a threshold implementation of AES," in *Eurocrypt*, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed., vol. 6632. Springer, 2011, pp. 69–88.

[11] J. Waddle and D. Wagner, "Towards efficient second-order power analysis," in *CHES*, ser. Lecture Notes in Computer Science, M. Joye and J.-J. Quisquater, Eds., vol. 3156. Springer, 2004, pp. 1–15.

[12] S. Chari, C. S. Jutla, J. R. Rao, and P. Rohatgi, "Towards sound approaches to counteract power-analysis attacks," in *Advances in Cryptology – Crypto '99*, ser. Lecture Notes in Computer Science, M. Wiener, Ed., vol. 1666. Springer, 1999, pp. 398–412.

[13] E. Prouff, M. Rivain, and R. Bevan, "Statistical analysis of second order differential power analysis," *IEEE Trans. Computers*, vol. 58, no. 6, pp. 799–811, 2009.

[14] F.-X. Standaert, N. Veyrat-Charvillon, E. Oswald, B. Gierlichs, M. Medwed, M. Kasper, and S. Mangard, "The world is not enough: Another look on second-order DPA," in *Asiacrypt*, ser. Lecture Notes in Computer Science, M. Abe, Ed., vol. 6477. Springer, 2010, pp. 112–129.

[15] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *CHES*, ser. Lecture Notes in Computer Science, M. Joye and J.-J. Quisquater, Eds., vol. 3156. Springer, 2004, pp. 16–29.

[16] L. Goubin and J. Patarin, "DES and differential power analysis (the duplication method)," in *CHES*, ser. Lecture Notes in Computer Science, Ç. K. Koç and C. Paar, Eds., vol. 1717. Springer, 1999, pp. 158–172.

[17] S. Mangard, N. Pramstaller, and E. Oswald, "Successfully attacking masked AES hardware implementations," in *CHES*, ser. Lecture Notes in Computer Science, J. Rao and B. Sunar, Eds., vol. 3659. Springer, 2005, pp. 157–171.

[18] T. Cover and J. Thomas, *Elements of Information Theory*. Wiley, 2006.

[19] E. Weisstein, "Skewness," From MathWorld–A Wolfram Web Resource, September 2012. [Online]. Available: http://mathworld.wolfram.com/Skewness.html

[20] P. McCullagh, *Tensor methods in statistics*, ser. Monographs on statistics and applied probability. London [u.a.]: Chapman and Hall, 1987. [Online]. Available: http://www.stat.uchicago.edu/ pmcc/tensorbook/

[21] L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F.-X. Standaert, and N. Veyrat-Charvillon, "Mutual information analysis: a comprehensive study," *J. Cryptology*, vol. 24, no. 2, pp. 269–291, 2011.

[22] B. Gierlichs, L. Batina, B. Preneel, and I. Verbauwhede, "Revisiting higher-order DPA attacks: Multivariate mutual information analysis," in *CT-RSA*, ser. Lecture Notes in Computer Science, J. Pieprzyk, Ed., vol. 5985. Springer, 2010, pp. 221–234.

[23] T. Le and M. Berthier, "Mutual information analysis under the view of higher-order statistics," in *IWSEC*, ser. Lecture Notes in Computer Science, I. Echizen, N. Kunihiro, and R. Sasaki, Eds., vol. 6434. Springer, 2010, pp. 285–300.

[24] G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "Duplexing the sponge: single-pass authenticated encryption and other applications," in *Selected Areas in Cryptography (SAC)*, 2011.

[25] ——, "The Keccak reference," January 2011.

[26] M. Joye and J.-J. Quisquater, Eds., *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, ser. Lecture Notes in Computer Science, vol. 3156. Springer, 2004.

[27] M. Wiener, Ed., *Advances in Cryptology – Crypto '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, ser. Lecture Notes in Computer Science, vol. 1666. Springer, 1999.