Vade Mecum on Side-Channels Attacks and Countermeasures for the Designer and the Evaluator

Sylvain Guilley, Olivier Meynard, Maxime Nassar, Guillaume Duc, Philippe Hoogvorst, Houssem Maghrebi, Aziz Elaabid, Shivam Bhasin, Youssef Souissi, Nicolas Debande, et al.

Vade Mecum on Side-Channels Attacks and Countermeasures for the Designer and the Evaluator

Sylvain Guilley, Olivier Meynard, Maxime Nassar, Guillaume Duc, Philippe Hoogvorst, Houssem Maghrebi,
Aziz Elaabid, Shivam Bhasin, Youssef Souissi, Nicolas Debande, Laurent Sauvage, Jean-Luc Danger
Institut TELECOM / TELECOM ParisTech, CNRS LTCI (UMR 5141)
Département COMELEC, 46 rue Barrault, 75 634 PARIS Cedex 13, FRANCE.
Email: sylvain.guilley@TELECOM-ParisTech.fr

***Abstract*—Implementation-level attacks are nowadays well known and most designers of security embedded systems are aware of them. However, both the number of vulnerabilities and of protections have seriously grown since the first public reporting of these threats in 1996. It is thus difficult to assess the correct countermeasures association to cover all the possible attack pathes. The goal of this paper is to give a clear picture of the possible adequation between actually risks and mitigation techniques. A specific focus is made on two protection techniques addressing primarily side-channel attacks: masking and hiding. For the first time, we provide with a way to estimate a tradeoff depending on the environmental conditions (amount of noise) and on the designer skills (ability to balance the design). This tradeoff is illustrated in a decision diagram, helpful for the security designer to justify choices and to account for the cost overhead.**

Key words: Implementation-level attacks, side-channel attacks, hiding and masking, leakage metric, comparison of countermeasures, decision diagram.

I. Introduction

Systems that process sensitive information can be the target of malevolent attacks that aim at recovering secrets illegitimately. Cryptography is the science that attempts to make it impossible for an attacker to retrieve private information. Encryption algorithms are typically used to conceal secrets. As a mathematical discipline, cryptography however makes some assumptions: the attacker is only expected to interact with the system through its regular interfaces. Now, when the cryptography is implemented in an embedded system, it is seriously challenged by attacks that make practical attempts to access the secrets. This means that all classical sneak tricks to access forbidden goods are possible. They include for instance spying, torturing, reversing or altering. Those actions are commonly referred to as "physical attacks".

A wealth of such attacks has been described and conducted experimentally with success on systems that were otherwise believed secure from the sole cryptographic standpoint. The first physical attack to be published was the "timing attack", presented at the conference CRYPTO in 1996 [1]. In this attack, an adversary is able to recover a secret key employed in a signature algorithm by spying on the time it takes for the system to output its result. This exploit is a typical "side-channel attack", insofar as it is completely passive: the attacked system does not even realize it is being stolen its secret key. Other side-channel attacks have been reported since then, and their study has mobilized many researchers. Those attacks unfold in two stages: side-channel collection and side-channel analysis (often abridged SCA). Side-channel collection is a straightforward "metrology" step, whereas SCA requires sophisticated tools to be efficient. Both aspects are advancing rapidly, as attested for instance by the "DPA contest" competitions [2]. In fact, the versions 3 and 4 are taking place in parallel in 2011 and address respectively the progress in acquisition and analysis of side-channel emanations.

This article focuses more particularly on SCA, because concepts involved in SCA are rich, and side-channel attacks can be conducted on virtually any embedded systems. Indeed, side-channel attacks enjoy two favorable properties. First of all, side-channel measurement is non-invasive: it seldom requires to modify or probe into the design. Second, side-channel attacks are passive, and thus the system is not aware of his being attacked, thus cannot take reactive countermeasures. This makes those attacks extremely likely to be mounted by non-professionals, with a fair chance of success unless the system is strongly leakage-proof. Thus, symmetrically, interesting countermeasures have been devised. They should have the specificity of being proactive, as the design must suppose it is constantly under attack.

The rest of the paper is structured as follows. An overview of side-channel attacks and countermeasures is given in Sec. II. Then, a more detailed analysis of specific countermeasures is described. Sec. III, IV and V address countermeasures against respectively timing attacks, simple and differential power analysis attacks. Conclusions are in Sec. VI.

II. Side-Channel Attacks and Countermeasures

A. Physical Side-Channels, and Statistical Tools to Exploit Them

The side-channels can basically be sorted in two categories:

1) those where the duration of the cryptographic process is the leakage source, and
2) those where a physical quantity depending on time is leaked.

In the first case, for every invocation of the cryptographic primitive, a scalar is measured, whereas in the second case, many samples are collected. We call those samples a "trace",

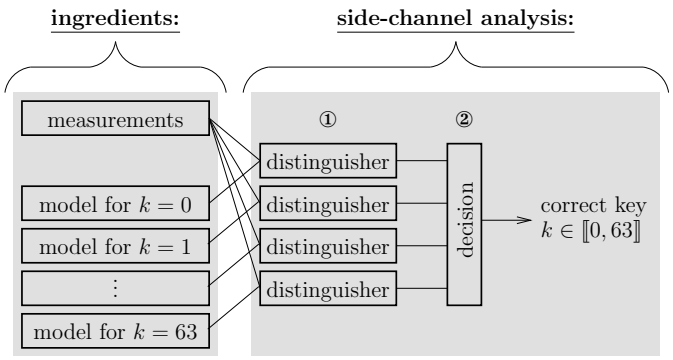

Fig. 1. Sketch of a side-channel attack where one correct key shall be extracted out of 64 key candidates.

by reference to the name given to measurement files captured by digital oscilloscopes. The measured quantity can be for example the instant current drawn by the cryptographic device (power analysis [3], [4]) or the magnetic field it radiates (electromagnetic analysis [5]).

However, in both cases, the SCA unfolds according to a classical cryptanalytic scenario, that is depicted in Fig. 1. The observations, either scalar or vectorial, are confronted to a model thanks to a distinguisher. More precisely, as many models as secret key hypotheses are derived. In Fig. 1, that applies to the case of DES key extraction, 64 models are considered. Indeed, in the DES algorithm (*confer* NIST FIPS PUB 46-3), each round key is consumed per words of 6 bits; guessing 6 bits of the first round key thus allows the attacker to predict 4 bits (because DES makes use of $6 \rightarrow 4$ substitution boxes) involved internally. The models can be any function of those four bits. Then, after the distinguisher has been applied, the attacker retains the most likely key.

Typically, the options for choosing a distinguisher are listed in Tab. I. For attacks that do not attempt to combine many samples from vectorial measurements, it has been argued in [6] that all these distinguishers are equivalent, *i.e.* that they eventually provide the correct key and differ only by statistical deviations when the number of observations is insufficient.

B. Typical Attacks

Attacks can be divided into two categories, depending on the characteristic of the side-channel:

- **Simple** attacks consist in the direct analysis of the side-channel, which requires only one measurement per analysis.
- **Differential** attacks require many measurements to test one hypothesis on a secret.

Timing attacks are attacks where the side-channel is the computation duration. In practice, simple timing attacks do not exist. Indeed, a system that would have a response time that directly depends on the secret would be very badly designed (unless this behaviour is intentional). However, differential timing attacks have been described. They exploit the horizontal variations of a cryptographic process. For instance, in [1], Kocher *et al.* describe how an attacker can test the secret key bits of a remote server by comparing the time it takes to answer to a local simulation (same programme, same hardware).

The attacks that require traces use vectorial observations. The analyzed quantities are the traces vertical values. Under favorable experimental conditions, RSA can be analyzed using a single power trace. Indeed, if the two operations involved in the computations can be visually distinguished, the sequence of operations is revealed by only one trace. In this case, referred to as single power analysis (SPA), the attacks consist indeed in the analysis of simple vertical variations. In [21], Kasper *et al.* show how to break KeeLoq with SPA. Also, elliptic curve cryptography is especially vulnerable to both timing attacks and SPA, because the "double" and "add" operations in the inner iteration loop notably execute differently.

Differential vertical variations are exploited by the other attacks, when timing attacks and SPA are unpractical due to countermeasures. They consist in statistical extraction of the secrets based on the study of dependence between the observations and the models. The literature has studied many of them: all those listed in Tab. I apply to SCAs taking advantage of differences in vertical variations (later on referred to as "DPA").

We provide in the code of Tab. II an example of DPA using the Pearson linear correlation as a distinguisher. The example considers a key extraction from an acquisition campaign comprised of $10,000$ traces made up of $1,000$ samples each. The campaign is integrally saved in RAM in one matrix called `measurements`. The 2^6 models have been precomputed in variable `models`. The SCA itself consists in two steps, as already mentioned in Fig. 1. The first step (①) is the evaluation of a distinguisher, whose result is stored in a $1,000 \times 64$ matrix, customarily called "differential traces". The second step (②) is the selection of the largest distinguisher value, which yields the correct key if the attack is successful.

It is not always trivial to define the most efficient attack. In this paper [23], authors mentioned that they succeeded in attacking KeeLoq in DPA when the algorithm was hardcoded. Now, when executed in software, the traces were misaligned due to a variable duration of the encryption. Hence, an SPA happened to be the most efficient attack. In conclusion, the authors also note that timing attacks could be less error-prone than SPA on this device.

C. Provable Countermeasures: Information Masking or Hiding

In this article, we discuss so-called provable countermeasures. By provable, we assume two conditions:

1) The countermeasure must be sound, meaning that in the framework of a given model, it can be demonstrated that its principle do indeed protect efficiently.
2) The countermeasure must adhere to Kerckhoffs' principle: it shall work even if its rational is completely exposed.

Two counter-examples are for instance the dummy cycles insertion, since it is not sound [24], and the code obfuscation,

TABLE I
VARIOUS DISTINGUISHERS SUITABLE FOR SCA.

Distinguisher	Decision	Comments
Difference of means (DoM)	Max.	Models are called "selection functions" [3]; refinements are provided in [7].
Covariance	Max.	Introduced initially as the multi-bit generalization of the DoM [8].
Correlation	Max.	Variants are Pearson [9] (often noted "ρ"), Spearman [10] or Kendall ("τ") correlation coefficients.
Likelihood	Max.	Used when probability density functions (PDF) can be estimated, and leads to Bayesian attacks [11].
Mutual information	Max.	Rely on off- or on-line PDF estimations [12], [13]. Models are also called "partitioning functions".
Least squares	Min.	Introduced in stochastic attacks [14]. Winning distinguisher for the 1st DPA contest (by Ch. Clavier).
Variance	Min.	Many references are available [15]–[18].
Principal components analysis (PCA)	Max.	First PCA (FPCA) [19] is a typical example of differential cluster analysis (DCA) [20].

TABLE II
SYNOPTIC OF A SCA IN MATLAB. OTHER CODE EXAMPLES CAN BE FOUND IN THE DPA CONTEST WEBSITE [2] OR IN THE OPENSCA [22] TOOLBOX.

```
% Ingredients:
measurements = [[...];[...];[...]]; % Side-channel traces, 10000 x 1000 matrix
models       = [[...];[...];[...]]; % Models for all hypotheses, 10000 x 64 matrix

% Analysis:
distinguishers = corr( measurements, models, 'type', 'Pearson' ); % 1000 x 64 matrix
plot( distinguishers ); % Optional "sanity check" step, to see the 64 differential traces
[ maxcorr, maxindex ] = max( max( distinguishers )); % Decision function associated with the correlation
% The correct key is maxindex-1 (since in MATLAB, the indices start from 1 and not from 0),
% and corresponds to the greatest correlation for all the 1000 dates and for all the 64 key candidates.
```

since it involves a secret method that is not expected to hold long against a determined attacker.

The two provable examples we consider in the sequel are:

1) **information masking** [4, Chp. 9], which aims at randomizing the side-channel, and
2) **information hiding** [4, Chp. 7], which aims at balancing the side-channel.

III. PROTECTION AGAINST TIMING ATTACKS

A. Masking

Let us take the example of the computation of a modular exponentiation $M^d \bmod N$ of a message M to the power d modulo the RSA modulus N. To eliminate the derivation of links between d and the computation time of $M^d \bmod N$, one could think to take advantage of the following identity:

$$\left(M^{d_1} \bmod N\right) \cdot \left(M^{d_2} \bmod N\right) \equiv M^{d_1+d_2} \bmod N \ . \quad (1)$$

It makes a "secret splitting" strategy possible. At every RSA computation that involves private key d, the system draws a random number d_1, and derives d_2 such that $d = d_1 + d_2$. The computation time using Eqn. (1) now also depends on d_1, unknown to the attacker.

Another masking countermeasure against timing attacks is called "secret blinding". For all random number r, we have: $M^{d+r\cdot\phi(N)} \equiv M^d \bmod N$. Hence a trivial way to randomize the execution length of RSA.

B. Hiding

The hiding countermeasure consists in having the computation unfold in a fixed amount of time. This solution works perfectly, because the timing is quantified (as clock periods). However, in practice, it is hard to really have a compiler produce portable and constant-time executables [1]. Hence assembly-level countermeasures, such as `xtime` for AES.

IV. PROTECTION AGAINST SPA

The protection of implementations against SPA requires greater skills than the protection against timing attacks. Indeed, if the attacker has at her disposal a complete trace of execution, she can distinguish internal operations by their different timing if they leak information this way. We thus suppose as a prerequisite that all key conditional operations execute in constant time.

A. Masking

The masking countermeasures presented against timing attacks do not apply to the protection against SPA. Indeed, let us assume internal operations can be distinguished via the observation of the side-channel [25]. Then the attacker retrieves d_1 and d_2 from implementations protected by exponent blinding, which trivially leads to $d = d_1 + d_2$. In the exponent splitting countermeasure, the attacker manages to extract $d + r \cdot \phi(N)$, that can be used as a legitimate private key.

Masking any internal operation seems very chancy. Thus, the protections against SPA rather rely on hiding.

B. Hiding

Basically, two approaches compete for the protection by hiding against SPA. The first one consists in having all the internal operations look similar. This is exemplified by the side-channel atomicity [26]. The second option is higher level. It aims at making the sequence of operations constant, using dummy operations (which proves to be dangerous, because of safe-errors [27]) or special redundant algorithms. For instance, the exponentiation based on the Montgomery ladder also performs the same operations irrespective of the secret key.

V. Protection against DPA

A. Masking

Masking the operations consists in changing the representation of the sensitive data x, possibly each time they are used. This requires to find identities where the injected randomness m can be canceled out. Such identities are for instance:

1) $\forall m, (x \oplus m) \oplus m = x$, which gives rise to Boolean masking [28],
2) $\forall m \neq 0, (x \times m) \times m^{-1} = x$, which gives rise to arithmetic masking [29] (value 0 requires special care).

In these identities, x is the sensitive variable and m the random mask. If x is n-bit long, so is m. Other possibilities are affine masking [30], a combination of Boolean and arithmetic masking, and homographic masking [31].

Those countermeasures prevent first-order attacks, but still leak information. Therefore advanced attacks are possible. Notably, high-order attacks [32] exploit the residual leakage of masking schemes.

B. Hiding

The hiding countermeasure against DPA is predominantly implemented as dual-rail with precharge logic (*aka* DPL [33]). In this representation, every Boolean variable x is implemented as a couple of wires (x_t, x_f), such that:

- $(x_t, x_f) = (0, 0)$ or $(1, 1)$ in precharge phase, which prevents memory effects and enables positive (glitch-free [34]) computation, and
- $(x_t, x_f) = (x, \overline{x})$ in evaluation phase, which makes the activity independent of x.

This protection is easier to implement in hardware than in software. Indeed, in software, it is difficult to control the register transfers, all the more so as most of times, the internal architecture of the CPU is unknown. However, some works tend to show that hiding can be achieved in software too [35].

C. Comparison of Masking and Hiding against DPA

It is relatively easy and straightforward to get rid off design flaws that open the door to timing attacks and SPA. Now, fighting DPA is more difficult, and moreover, masking and hiding against DPA are costly countermeasures. It is thus important to compare them, because the designer has a major choice to make between them.

At first glance, masking seems easier to code properly, because it is a "source-level" countermeasure. However, if implemented at source-level, the masking is certainly doomed to fail. Indeed, a clever compiler will remove all the redundant data, and eventually end up with the optimized (and thus unprotected) description of the algorithm. Thus both masking and hiding schemes require writing the description of the countermeasure manually, at assembly language level for software or at netlist level for hardware.

TABLE III
Illustration of the unbalance α on the resources' relative importance in the leakage.

Countermeasure	Resource	Weight	Leakage ($\mathcal{L}$)
Masking	n-bit mask	$1+\alpha$	$(1+\alpha)\cdot \mathrm{HW}(m)$
	n-bit masked data	1	$1\cdot \mathrm{HW}(x\oplus m)$
Hiding	n-bit true data	$1+\alpha$	$(1+\alpha)\cdot \mathrm{HW}(x)$
	n-bit false data	1	$1\cdot \mathrm{HW}(\overline{x})$

In terms of area overhead, both masking and hiding require to duplicate the datapath. Variable x is represented as a masked variable and a mask in masking, and as a true and a false variable in hiding. In terms of throughput, no change occurs for masking in hardware, since the masked data and the mask can be computed in parallel. By default, the throughput of DPL is halved with respect to the unprotected implementation, because of the precharge / evaluation sequence. However, some logic styles [36] manage to optimize this throughput by squeezing the precharge step. All in one, masking and hiding have a roughly comparable impact on the overhead.

Thus, to compare them, we consider only their level of security. The known flaw of masking is its susceptibility against high-order or information theoretic attacks, whereas hiding is rather susceptible to inaccurate balancing at the layout-level. To grasp both aspects, we introduce two parameters:

1) the amount of noise (assumed to be normally distributed) in the measurements, quantified by its variance σ^2, and
2) the backend unbalance, measured by α, defined in Tab. III.

Ideal conditions for the defender correspond to $\sigma^2 = +\infty$ and $\alpha = 0$.

Hence the leakage models for n-bit resource x:

1) $\mathcal{L}_{\text{masking}}(x, m) \sim (1+\alpha)\cdot \mathrm{HW}(m) + \mathrm{HW}(x \oplus m) + \mathcal{N}(0, \sigma^2)$, where m is independent from x and follows a uniform distribution in $\{0,1\}^n$, and
2) $\mathcal{L}_{\text{hiding}}(x) \sim (1+\alpha)\cdot \mathrm{HW}(x) + \mathrm{HW}(\overline{x}) + \mathcal{N}(0, \sigma^2) = \alpha \cdot \mathrm{HW}(x) + \mathcal{N}(n, \sigma^2)$.

There are two kinds of security analyses that can be performed [37]. They lead to those metrics:

1) the success rate or the guessing entropy after an attack, and
2) the estimation of the leakage by information theoretic tools, such as the mutual information as a metric (MIM).

The first option is difficult, since masking and hiding countermeasures are not jeopardized by the same attacks. For instance, against first-order CPA [9], we have:

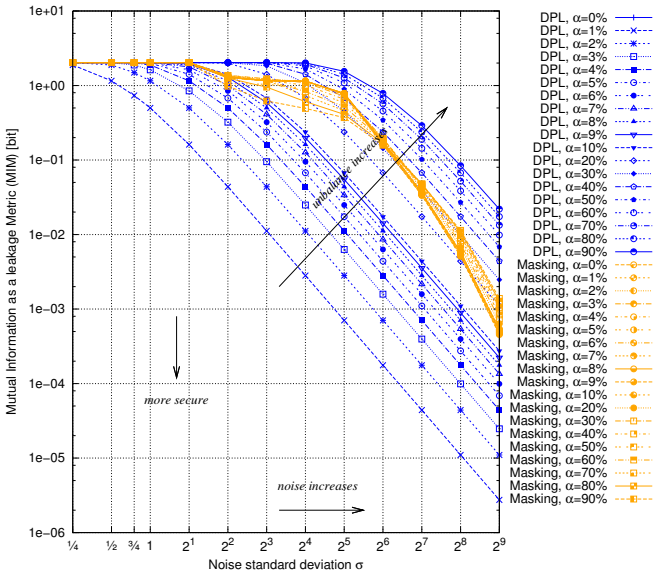

Fig. 2. Comparison between the leakage of DPL and masking countermeasures as a function of the experimental noise, for various α and $n = 4$ bit.

- $\rho_{x,m}\left(\mathcal{L}_{\text{masking}}(x,m); \text{HW}(x)\right) = 0, \forall\alpha$, whereas
- $\rho_{x}\left(\mathcal{L}_{\text{hiding}}(x); \text{HW}(x)\right) = \frac{\alpha\sqrt{n}}{\sqrt{n\alpha^2+4\sigma^2}} \neq 0$ if $\alpha \neq 0$.

Thus the information theoretic analysis is more suited in our case to compare the two countermeasures. Results are shown in Fig. 2. It appears logically that the noise (quantified by its variance σ^2) reduces the mutual information, whereas the unbalance (quantified by α) increases it. However, the masking is much less impacted by the technological unbalance. The curves show that the less leaking countermeasure depends on the value of the couple (σ, α).

The leakage of the best countermeasure is plotted as a function of σ and α in Fig. 3. The leakage is expressed in bits, and represented in logarithmic scale. The areas without color correspond to the equality between the two countermeasures. It appears that, roughly speaking, for unbalances up to 17 %, DPL is the most secure choice. And for some values of the noise, namely $\sigma \in [2^4, 2^8]$, DPL remains the most secure solution for α up to 30 %. This graph therefore enables the designer to choose the most adequate countermeasure depending on the estimated environmental noise and on his ability to properly balance the layout.

D. General Picture

Before concluding, we wish to replace the problematic of protecting embedded systems into its general context. Side-channel attacks are only one class of attacks: what is thus the suitability of masking and hiding against the other attack strategies? The suitability of countermeasures to thwart attacks (as discussed in the previous paragraphs) is given in Fig. 4.

This figure shows that masking is also a countermeasure against probing attacks, since the value of the probed node becomes random. Also, hiding is a countermeasure against most fault injection attacks since the attacker erases the value

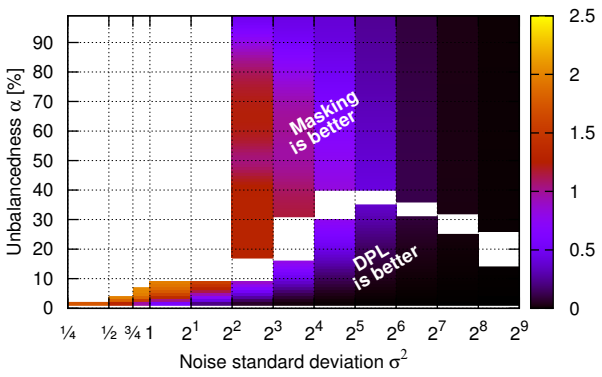

Fig. 3. Plot of domains where either masking or DPL leaks less (units: bit).

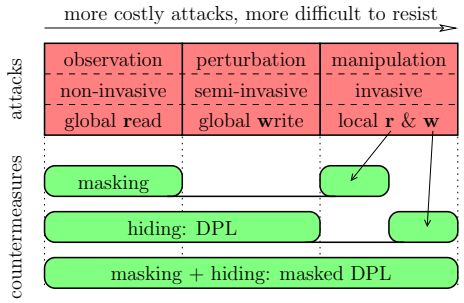

Fig. 4. Coverage of countermeasures for all physical attacks classes.

stored redundantly in one pair of wires by changing only one of them. The case of symmetric faults is covered in [38] and of arbitrary faults in [39].

An interesting noting is that by associating masking and hiding, the protection extends to semi-invasive and invasive attacks. This association must be realized with care, since otherwise some attacks become possible, such as the "folding attack" [40] or the "subset attack" [41]. The synopsis of this attack consists in recovering the masking bit and then to defeat the hiding countermeasure. However, by using more than one bit of mask, these attacks become impossible.

VI. Conclusions

Cryptographic implementations can leak information in both time and amplitude. In this article, we provide a survey of known side-channels and we classify them according to their nature (horizontal / vertical) and the bias they disclose (simple / differential). Then, we review suitable countermeasures, and insist in particular on the masking and the hiding protection techniques. We specifically investigate these countermeasures in the context of vertical differential attacks, generically nicknamed DPA. It appears that they have roughly speaking the same cost, and thus differ only in the added security they bring to the design. We use a mutual information analysis to quantify their leakage, in the context of noisy measurements and imper-

fect resources matching. It appears that no countermeasure is better than the other in the complete studied domain. Instead, the choice depends on the environmental noise and on the skill of the designer to balance the resources at the backend-level. Eventually, we mention that masking and hiding can be constructively combined to achieve an immunity against all implementation-level attacks.

REFERENCES

[1] P. C. Kocher, J. Jaffe, and B. Jun, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Proceedings of CRYPTO'96*, ser. LNCS, vol. 1109. Springer-Verlag, 1996, pp. 104–113, (PDF).

[2] TELECOM ParisTech SEN research group, "DPA Contests," 2008–2011, http://www.DPAcontest.org/.

[3] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Proceedings of CRYPTO'99*, ser. LNCS, vol. 1666. Springer-Verlag, 1999, pp. 388–397.

[4] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, December 2006, iSBN 0-387-30857-1, http://www.dpabook.org/.

[5] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic Analysis: Concrete Results," in *CHES*, ser. LNCS, vol. 2162. Springer, May 14-16 2001, pp. 251–261, Paris, France.

[6] S. Mangard, E. Oswald, and F.-X. Standaert, "One for All - All for One: Unifying Standard DPA Attacks," Cryptology ePrint Archive, Report 2009/449, 2009, http://eprint.iacr.org/2009/449. To appear in "IET Information Security".

[7] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining Smart-Card Security under the Threat of Power Analysis Attacks," *IEEE Trans. Computers*, vol. 51, no. 5, pp. 541–552, 2002.

[8] R. Bevan and E. Knudsen, "Ways to Enhance Differential Power Analysis," in *ICISC*, ser. Lecture Notes in Computer Science, vol. 2587. Springer, November 28-29 2002, pp. 327–342, Seoul, Korea.

[9] É. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," in *CHES*, ser. LNCS, vol. 3156. Springer, August 11–13 2004, pp. 16–29, Cambridge, MA, USA.

[10] L. Batina, B. Gierlichs, and K. Lemke-Rust, "Comparative Evaluation of Rank Correlation Based DPA on an AES Prototype Chip," in *ISC*, ser. Lecture Notes in Computer Science, vol. 5222. Springer, September 15-18 2008, pp. 341–354, Taipei, Taiwan.

[11] S. Chari, J. R. Rao, and P. Rohatgi, "Template Attacks," in *CHES*, ser. LNCS, vol. 2523. Springer, August 2002, pp. 13–28, San Francisco Bay (Redwood City), USA.

[12] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel, "Mutual information analysis," in *CHES, 10th International Workshop*, ser. LNCS, vol. 5154. Springer, August 10-13 2008, pp. 426–442, Washington, D.C., USA.

[13] H. Maghrebi, S. Guilley, J.-L. Danger, and F. Flament, "Entropy-based Power Attack," in *HOST*, ser. IEEE Computer Society, June 13-14 2010, pp. 1–6, Anaheim Convention Center, Anaheim, CA, USA. DOI: 10.1109/HST.2010.5513124.

[14] W. Schindler, K. Lemke, and C. Paar, "A Stochastic Model for Differential Side Channel Cryptanalysis," in *CHES*, ser. LNCS, vol. 3659. Springer, Sept 2005, pp. 30–46, Edinburgh, Scotland, UK.

[15] F.-X. Standaert, B. Gierlichs, and I. Verbauwhede, "Partition vs. Comparison Side-Channel Distinguishers: An Empirical Evaluation of Statistical Tests for Univariate Side-Channel Attacks against Two Unprotected CMOS Devices," in *ICISC*, ser. LNCS, vol. 5461. Springer, December 3-5 2008, pp. 253–267, Seoul, Korea.

[16] H. Maghrebi, J.-L. Danger, F. Flament, and S. Guilley, "Evaluation of Countermeasures Implementation Based on Boolean Masking to Thwart First and Second Order Side-Channel Attacks," in *SCS*, ser. IEEE, November 6–8 2009, pp. 1–6, Jerba, Tunisia. Complete version online: http://hal.archives-ouvertes.fr/hal-00425523/en/. DOI: 10.1109/ICSCS.2009.5412597.

[17] Y. Li, K. Sakiyama, L. Batina, D. Nakatsu, and K. Ohta, "Power Variance Analysis breaks a masked ASIC implementation of AES," in *DATE*. IEEE, March 8-12 2010, pp. 1059–1064, Dresden, Germany.

[18] P. Hoogvorst, "The Variance Power Attack," in *COSADE*, February 4-5 2010, pp. 4–9, Darmstadt, Germany. http://cosade2010.cased.de/files/proceedings/cosade2010_paper_2.pdf.

[19] Y. Souissi, M. Nassar, S. Guilley, J.-L. Danger, and F. Flament, "First Principal Components Analysis: A New Side Channel Distinguisher," in *ICISC*, ser. LNCS. Springer, December 1-3 2010, Seoul, Korea.

[20] L. Batina, B. Gierlichs, and K. Lemke-Rust, "Differential Cluster Analysis," in *Cryptographic Hardware and Embedded Systems – CHES 2009*, ser. Lecture Notes in Computer Science, C. Clavier and K. Gaj, Eds., vol. 5747. Lausanne, Switzerland: Springer-Verlag, 2009, pp. 112–127.

[21] M. Kasper, T. Kasper, A. Moradi, and C. Paar, "Breaking KeeLoq in a Flash: On Extracting Keys at Lightning Speed," in *AFRICACRYPT*, ser. LNCS, B. Preneel, Ed., vol. 5580. Springer, 2009, pp. 403–420.

[22] E. Oswald, "http://opensca.sourceforge.net/, U. of Bristol, UK," 2010.

[23] C. Paar, T. Eisenbarth, M. Kasper, T. Kasper, and A. Moradi, "Keeloq and side-channel analysis-evolution of an attack," in *FDTC*, L. Breveglieri, I. Koren, D. Naccache, E. Oswald, and J.-P. Seifert, Eds. IEEE Computer Society, 2009, pp. 65–69.

[24] C. Clavier, J.-S. Coron, and N. Dabbous, "Differential Power Analysis in the Presence of Hardware Countermeasures," in *CHES*, ser. LNCS. London, UK: Springer-Verlag, August 2000, pp. 252–263.

[25] O. Meynard, D. Réal, S. Guilley, J.-L. Danger, and N. Homma, "Enhancement of Simple Electro-Magnetic Attacks by Pre-characterization in Frequency Domain and Demodulation Techniques," in *DATE*. IEEE Computer Society, March 14-18 2011, Grenoble, France.

[26] B. Chevallier-Mames, M. Ciet, and M. Joye, "Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity," *IEEE Trans. Computers*, vol. 53, no. 6, pp. 760–768, 2004.

[27] S.-M. Yen and M. Joye, "Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis," *IEEE Trans. Computers*, vol. 49, no. 9, pp. 967–970, 2000, DOI: 10.1109/12.869328.

[28] L. Goubin and J. Patarin, "DES and Differential Power Analysis," in *CHES*, ser. LNCS. Springer, Aug 1999, pp. 158–172, worcester, MA, USA.

[29] M.-L. Akkar and C. Giraud, "An Implementation of DES and AES Secure against Some Attacks," in *Proceedings of CHES'01*, ser. LNCS, LNCS, Ed., vol. 2162. Springer, May 2001, pp. 309–318, Paris, France.

[30] G. Fumaroli, A. Martinelli, E. Prouff, and M. Rivain, "Affine masking against higher-order side channel analysis," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, A. Biryukov, G. Gong, and D. R. Stinson, Eds., vol. 6544. Springer, 2010, pp. 262–280.

[31] E. Prouff and T. Roche, "Attack on a Higher-Order Masking of the AES Based on Homographic Functions," in *INDOCRYPT*, ser. LNCS, G. Gong and K. C. Gupta, Eds., vol. 6498. Springer, 2010, pp. 262–281.

[32] T. S. Messerges, "Using Second-Order Power Analysis to Attack DPA Resistant Software," in *CHES*, ser. LNCS, vol. 1965. Springer-Verlag, August 17-18 2000, pp. 238–251, Worcester, MA, USA.

[33] J.-L. Danger, S. Guilley, S. Bhasin, and M. Nassar, "Overview of Dual Rail with Precharge Logic Styles to Thwart Implementation-Level Attacks on Hardware Cryptoprocessors, — *New Attacks and Improved Counter-Measures* —," in *SCS*, ser. IEEE, November 6–8 2009, pp. 1–8, Jerba, Tunisia. Complete version online: http://hal.archives-ouvertes.fr/hal-00431261/en/. DOI: 10.1109/ICSCS.2009.5412599.

[34] S. Guilley, L. Sauvage, J.-L. Danger, T. Graba, and Y. Mathieu, "Evaluation of Power-Constant Dual-Rail Logic as a Protection of Cryptographic Applications in FPGAs," in *SSIRI*. Yokohama, Japan: IEEE Computer Society, jul 2008, pp. 16–23, DOI: 10.1109/SSIRI.2008.31, http://hal.archives-ouvertes.fr/hal-00259153/en/.

[35] P. Hoogvorst, G. Duc, and J.-L. Danger, "Software Implementation of Dual-Rail Representation," in *COSADE*, February 24-25 2011, pp. 73–81, Darmstadt, Germany. http://cosade2011.cased.de/files/2011/cosade2011_talk8_paper.pdf.

[36] M. Nassar, S. Bhasin, J.-L. Danger, G. Duc, and S. Guilley, "BCDL: A high performance balanced DPL with global precharge and without early-evaluation," in *DATE'10*. IEEE Computer Society, March 8-12 2010, pp. 849–854, Dresden, Germany.

[37] F.-X. Standaert, T. Malkin, and M. Yung, "A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks," in *EUROCRYPT*, ser. LNCS, vol. 5479. Springer, April 26-30 2009, pp. 443–461, Cologne, Germany.

[38] S. Guilley, L. Sauvage, J.-L. Danger, and N. Selmane, "Fault Injection Resilience," in *FDTC*. IEEE Computer Society, August 21 2010, pp. 51–65, Santa Barbara, CA, USA. DOI: 10.1109/FDTC.2010.15; Complete version: http://hal.archives-ouvertes.fr/hal-00482194/en/.

[39] S. Bhasin, J.-L. Danger, F. Flament, T. Graba, S. Guilley, Y. Mathieu, M. Nassar, L. Sauvage, and N. Selmane, "Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow," in *ReConFig*. IEEE Computer Society, December 9–11 2009, pp. 213–218, Cancún, Quintana Roo, México, DOI: 10.1109/ReConFig.2009.50, http://hal.archives-ouvertes.fr/hal-00411843/en/.

[40] P. Schaumont and K. Tiri, "Masking and Dual Rail Logic Don't Add Up," in *CHES*, ser. LNCS, vol. 4727. Springer, September 10-13 2007, pp. 95–106, Vienna, Austria.

[41] E. D. Mulder, B. Gierlichs, B. Preneel, and I. Verbauwhede, "Practical DPA Attacks on MDPL," in *First International Workshop on Information Forensics and Security (WIFS)*. IEEE Signal Processing Society, December 6-9 2009, london, United Kingdom. Also http://eprint.iacr.org/2009/231.