Low Entropy Masking Schemes, Revisited

Vincent Grosso[1], François-Xavier Standaert[1], Emmanuel Prouff[2]

[1] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium.
[2] ANSSI, 51 Bd de la Tour-Maubourg, 75700 Paris 07 SP, France.

Abstract. Low Entropy Masking Schemes (LEMS) are a recent countermeasure against side-channel attacks. They aim at reducing the randomness requirements of masking schemes under certain (adversarial and implementation) conditions. Previous works have put forward the interest of this approach when such conditions are met. We complement these investigations by analyzing LEMS against adversaries and implementations that deviate from their expected behavior, in a realistic manner. Our conclusions are contrasted: they confirm the theoretical interest of the countermeasure, while suggesting that its exploitation in actual products may be risky, because of hard(er) to control hardware assumptions.

1 Introduction

Masking is a frequently considered countermeasure against side-channel attacks. In a masked implementation, any sensitive data is split into several shares, and all the computations are performed on the shared values only. For this purpose, the algorithm must be written in a way that is consistent with this representation of the sensitive data. The resulting process, usually called d-sharing scheme when the data is split in d shares, is expected to provide improved physical security since: (i) more "points of interest" (i.e. more dimensions in the leakage distribution) may have to be identified and exploited concurrently by the adversary, and (ii) if the masking scheme is carefully implemented (i.e. if the leakages of all the shares are independent), higher-order moments of the leakage distribution have to be estimated to reveal key-dependent information. The latter property is known as the "d-1th-order SCA security" [4]. It has been shown that the data complexity of a successful attack against such an implementation increases exponentially with the number of shares (first in the restricted context of single-bit DPA[3], then experimentally in more general contexts [13], and more recently using the mutual information put forward in [12] as evaluation metric [9]).

Quite naturally, a central condition for this SCA security guarantees to hold is that all the shares are uniformly distributed, which implies strong randomness requirements in masked implementations [5]. Starting from this observation, a recent line of works - denoted as Low Entropy Masking Schemes (LEMS) in the following- has investigated possibilities to maintain the security order of masked implementations with reduced randomness requirements [1, 2, 7, 8]. LEMS can be seen as 2-sharing schemes, with the particularity that any n-bit sensitive value x is randomized with a mask variable M chosen within a subset (aka code) of the 2^n possible masks. In this setting, preserved security orders can be obtained with reduced randomness requirements under two important conditions:

1. *Adversarial condition.* The attacks performed are only univariate, i.e. they exploit exclusively the leakage of the masked value $x \oplus M$.
2. *Implementation condition.* The leakage function's deterministic part is linear in the bits of $x \oplus M$ (such as, e.g. for the Hamming weight function).

These results directly raise the question whether such conditions are realistic - i.e. whether LEMS can give rise to actual security improvements in practical scenarios. In order to answer this question, this paper provides a systematic evaluation of these assumptions, leading to two main results.

1. *On the adversarial condition.* In general, it is of course natural to consider multivariate attacks, since the shares used in any masked implementation have to be generated on chip, which possibly leaks information. We analyze such bivariate attacks and show that despite the reduced number of masks, LEMS still provide first-order security in this case (with a slight security degradation). We further confirm that if an adversary is limited (for some reasons) to univariate attacks, LEMS allow ensuring security orders of 2 or 3, as previously demonstrated by Carlet et al. [2] and Nassar et al. [8].
2. *On the implementation condition.* We show that as soon as the leakage function's deterministic part deviates from a purely linear one, the security guarantees provided by LEMS vanish, even in the univariate attack context. We further illustrate that the security order of the countermeasure is reduced according to the degree of the leakage function, e.g. that a quadratic leakage function is less damaging than a cubic one, quartic one, . . . and additionally provide an explanation of this phenomenon (see Section 3.2).

Summarizing, the first (adversarial) condition may not be a too big issue in practice. Given that maximum 2-share implementations are considered[1], LEMS are a theoretically relevant solution to mask under the assumption of linear leakage functions, since it maintains the security order of univariate (resp. bivariate) attacks to two or three (resp. one). By contrast the second (implementation) condition seems more difficult to fulfill, since the shape of a leakage function is typically hard to control by cryptographic designers. We conclude that despite its theoretical interest, the deployment of LEMS in actual embedded devices should be considered with care, and standard masking schemes are generally safer to implement because of easier-to-verify hardware assumptions.

2 Background

2.1 Univariate vs. multivariate / 1^{st}-order vs. higher-order attacks

Let X be a sensitive variable and $\mathbf{L} = [L_1, L_2, \ldots, L_d]$ be a leakage trace. A side-channel attack typically exploits the conditional distributions $\Pr[X|\mathbf{L}]$ in order to recover information about X. We say that the attack is univariate if it exploits unidimensional leakage vectors $\mathbf{L} = [L_1]$. We say that the attack is bivariate if it

[1] Current results in LEMS do not provide generalizations to more shares.

exploits bidimensional leakage vectors $\mathbf{L} = [L_1, L_2]$. More generally, the attack is said to be d-variate if it exploits multidimensional leakage vectors with d samples $\mathbf{L} = [L_1, L_2, \ldots, L_d]$. Note that finding the samples of interest in a leakage trace is usually challenging, which may be a reason for some adversaries to restrict themselves to univariate attacks when it is possible. Of course, leaving leakage samples aside may only result in a loss of information, hence a suboptimal attack.

Independent of the dimensionality of the leakage distribution, the order of a side-channel attack relates to the smallest (mixed) statistical moment that leaks sensitive information. For this purpose, we use the following definitions:

Definition 1 (Central moment of order d). *Let X be a random variable, then the central moment of order d of X is defined by:*

$$\mathsf{E}((X - \mathsf{E}(X))^d),$$

Definition 2 (Central mixed moment of orders $d_1, \ldots, d_r$). *Let $\{X_i\}_{i=1}^r$ be a set of r random variables, then the central mixed moment of orders $d_1, \ldots, d_r$ of $\{X_i\}_{i=1}^r$ is defined by:*

$$\mathsf{E}((X_1 - \mathsf{E}(X_1))^{d_1} \times \cdots \times (X_r - \mathsf{E}(X_r))^{d_r}).$$

In both definitions, $\mathsf{E}(.)$ denotes the expectation operator. For simplicity, we will sometimes denote the integer value $d = \sum_i d_i$ as the order of the central mixed moment of a tuple $(X_i)_{i=1..r}$. Central moments are typically used in univariate attacks (e.g against hardware implementations, where the different shares of a masked implementation are manipulated in parallel). Central *mixed* moments are typically used in multivariate attacks (e.g. against software implementations, where the different shares of a masked implementation are processed sequentially). Intuitively, the dimensionality of an attack has a direct impact on its time complexity (since it determines the number of samples on which the distinguisher has to be applied). By contrast the order of an attack mainly relate to its data complexity (since the number of measurements required to estimate a statistical moment increases with the order of this moment) [3, 9].

2.2 Low entropy masking schemes

As detailed in the introduction, the main goal of LEMS is to guarantee high security orders for masked implementations, with less randomness requirements than traditional masking schemes. For this purpose, the mask M (which is bitwise added to the sensitive datum s) is chosen as part of a sub-set of the definition set of s. Different solutions have been published in the literature. In the rest of the paper, we will use the code proposed in [1], next referred to as C_{16}, and to the one proposed in [8], next referred to as C_{12}. Both subsets are designed for 8-bit sensitive values (i.e. are typically applicable to protect the registers of 8-bit devices). Following previous analyzes, LEMS with C_{12} is expected to provide security against first- and second-order attacks, while LEMS with C_{16} is expected to provide security against first-, second- and third-order attacks (under the adversarial and implementation conditions stated in introduction). Codes are specified as: $C_{12} = \{\texttt{0x03}, \texttt{0x18}, \texttt{0x3f}, \texttt{0x55}, \texttt{0x60}, \texttt{0x6e}, \texttt{0x8c}, \texttt{0xa5}, \texttt{0xb2}, \texttt{0xcb}, \texttt{0xd6}, \texttt{0xf9}\}$, $C_{16} = \{\texttt{0x10}, \texttt{0x1f}, \texttt{0x26}, \texttt{0x29}, \texttt{0x43}, \texttt{0x4c}, \texttt{0x75}, \texttt{0x7a}, \texttt{0x85}, \texttt{0x8a}, \texttt{0xb3}, \texttt{0xbc}, \texttt{0xd6}, \texttt{0xd9}, \texttt{0xe0}, \texttt{0xef}\}$. Both were selected amongst the lowest size set that provides the required security order, while the first one minimizes the mutual information metric defined in the next subsection as additional criteria.

2.3 Evaluation framework

We will analyze the LEMS countermeasure based on the evaluation framework introduced in [12], which holds in two main steps. First, an Information Theoretic (IT) analysis is performed, in order to analyze the leakages independent of the adversary exploiting them. It is aimed to capture the quality of a countermeasure in a worst-case scenario. Next, a security analysis is performed, in order to evaluate the actual data complexity required by an adversary to exploit the available leakage (e.g. in order to turn it into a key recovery). For this purpose, we will consider the following simulated leakages. Let s be a sensitive value (i.e. the target of the attack), M a variable representing a word of the code used to protect the sensitive value, and N_1, N_2 two normally distributed noise variables, with mean 0 and variance σ^2. We define our leakages as:

$$L_1 = \mathsf{L}(s \oplus M) + N_1,$$
$$L_2 = \mathsf{L}(M) + N_2,$$

where $\mathsf{L}(.)$ is a polynomial in the bits of the input. In the following, we will assume this polynomial to be the Hamming weight function (excepted in subsection 3.2, where we will consider higher-degree polynomials). Furthermore, we will consider both univariate attacks exploiting only the leakage sample L_1, and bivariate attacks exploiting L_1 and L_2 jointly[2]. This implies computing the following information theoretic metric in the univariate case:

$$\mathrm{PI}(S; L_1) = \mathrm{H}[S] - \sum_{s \in \mathcal{S}} \Pr[s] \sum_{l_1 \in \mathcal{L}} \Pr_{\text{chip}}[l_1|s] \cdot \log_2 \Pr_{\text{model}}[s|l_1],$$

and its extension to two dimensions in the bivariate case:

$$\mathrm{PI}(S; L_1, L_2) = \mathrm{H}[S] - \sum_{s \in \mathcal{S}} \Pr[s] \sum_{l_1, l_2 \in \mathcal{L}} \Pr_{\text{chip}}[l_1, l_2|s] \cdot \log_2 \Pr_{\text{model}}[s|l_1, l_2].$$

Let us denote the probability density function of a Gaussian distribution taken on input x, with mean μ (resp. mean vector $\boldsymbol{\mu}$) and variance σ^2 (resp. covariance matrix Σ) as $\mathcal{N}(x|\mu, \sigma^2)$ (resp. $\mathcal{N}(x|\boldsymbol{\mu}, \Sigma)$). We will generally compute the

[2] Note that the univariate attacks considered in LEMS are different than the classical univariate higher-order DPAs, where a combination of the two leakage samples (e.g. their normalized product) is exploited by the adversary [10]. Any such combination would provide leakages and successful attacks similar to the ones of a bivariate attack, with an information loss similar to the one investigated in [13].

probabilities in these equations as follows (e.g. in the bivariate case):

$$\underset{\text{model}}{\Pr}[s|l_1,l_2] = \frac{\mathcal{N}(l_1,l_2|\boldsymbol{\mu}_s,\mathit{\Sigma}_s)}{\sum_{s^*\in\mathcal{S}}\mathcal{N}(l_1,l_2|\boldsymbol{\mu}_{s^*},\mathit{\Sigma}_{s^*})}, \tag{1}$$

for unprotected implementations, and:

$$\underset{\text{model}}{\Pr}[s|l_1,l_2] = \frac{\sum_{m^*\in C}\mathcal{N}(l_1,l_2|\boldsymbol{\mu}_{s,m^*},\mathit{\Sigma}_{s,m^*})}{\sum_{s^*\in\mathcal{S}}\sum_{m^*\in C}\mathcal{N}(l_1,l_2|\boldsymbol{\mu}_{s^*,m^*},\mathit{\Sigma}_{s^*,m^*})}, \tag{2}$$

for masked implementations (and similarly for LEMS), with all the secrets and masks distributed uniformly over their specified set. That is, the leakage distributions conditioned on the sensitive values will be modeled as Gaussian mixtures, where each mode corresponds to a mask value. Following the discussion in [11] and since we are considering simulated experiments, the probability distributions $\Pr_{\text{chip}}$ and $\Pr_{\text{model}}$ will be identical in most of our evaluations. This implies that the Perceived Information (PI) will be identical to the (classical) Mutual Information (MI) in most cases. As only exception, we will also evaluate the information leakage of a suboptimal bivariate adversary, who models leakage distributions conditioned on the sensitive values as single (bivariate) Gaussians, i.e. who simplifies Equation 2 into Equation 1, even in the masked case. This boils down to summarizing the second-order information in the covariance between the leakage samples l_1 and l_2. By plotting the MI/PI metrics in function of the noise variance, we can directly obtain intuition about the order of the masking, which simply corresponds to the slope of these curves [13].

Following the information theoretic analysis, we will apply a security analysis and compute the success rate (as defined in [12]) of template attacks against the target s, using $\Pr_{\text{model}}[s|l_1]$ and $\Pr_{\text{model}}[s|l_1,l_2]$ as leakage models. This will allow us to evaluate the data complexities of these worst-case attacks in Section 4.

3 Information theoretic analysis of LEMS

3.1 Hamming weight leakages

Our IT analysis of LEMS and its comparison with other masking schemes are in Figure 1, from which the following observations can be extracted.

Starting with the univariate case (in the left part of the figure), we first observe that information leakage is only available if a strict subset of the 2^n possible masks is available (e.g. the curves (—×—) and (—○—) are stuck to zero in this case, hence not represented in this part of the figure). We also note that a badly chosen code (e.g. $C = \{\texttt{0x00},\texttt{0x01},\texttt{0x02},\ldots,\texttt{0x0B}\}$) leads to first-order univariate weaknesses for the LEMS countermeasure, as witnessed by the slope of the curve (—○—) that is parallel to the one of the unprotected implementation (—+—). This confirms the requirement to use uniform randomness in the security proofs of standard masking schemes, e.g. [6, 9]. By contrast and

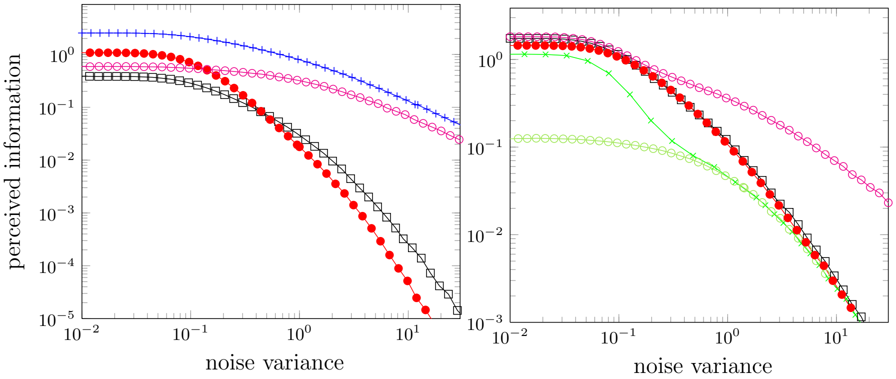

Fig. 1: Information theoretic analysis of different masking schemes. Left: univariate attacks. Right: bivariate attacks. The curve (—+—) is for the unprotected case. The curves (—□—, —●— and —○—) are for LEMS with C_{12}, C_{16} and a badly chosen code, respectively. The curve (—×—) is for masking with the full set (only non-zero in the bivariate case). The curve (—○—) is for the bivariate attack using approximated Gaussian templates (in place of Gaussian mixtures) for masking with the full set.

as expected, the LEMS countermeasure with codes C_{12} and C_{16} enforces second- and third-order security against univariate side-channel attacks (i.e. curve (—□—) has slope 3 and curve (—●—) has slope 4). Interestingly, we also see that C_{12} leads to a slightly smaller information leakage than C_{16} for low noise values - which is also expected since minimizing the information leakages was considered as an additional optimization criteria in the selection of C_{12} only.

Next in the bivariate case, we first observe that most attacks (i.e. using all masks with Gaussian or Gaussian mixture modeling, and using C_{12} or C_{16}) converge towards the same slope as the noise increases. The slope of these curves is 2 implying first-order security in all these cases The curve (—○—) is again a counter-example, because of a badly chosen code. So an important conclusion is that the first (adversarial) condition mentioned in introduction for LEMS to provide improved security against univariate attacks does not imply a penalty in the security order when considering bivariate attacks. By contrast, we observe a small security degradation for small noise values, i.e. a constant information leakage loss between curves (—□—), (—●—) and (—×—), similar to the difference between C_{12} and C_{16} in univariate attacks. Interestingly, we also observe the impact of incorrect modeling for these small noise values. That is, when considering Gaussian mixture leakage models - as for curve (—×—) - we see a "wave" in the information theoretic curve that is not found when simplifying the mixtures into a simpler Gaussian model - as for curve (—○—). This wave can be explained by the fact that characterizing the full distribution with a Gaussian mixture allows exploiting higher-order moments that are easy to estimate for low noise values (and hard to estimate with more noise). By contrast, the Gaussian modeling only exploits two statistical moments (i.e. mean vector, covariance matrix), leading to less (and more regular) information leakage. A similar reason makes the Gaussian modeling impossible to apply to univariate attacks against LEMS with C_{12} and C_{16}: since such attacks only leak in the third- and fourth-order moments of the conditional leakage distributions, a Gaussian model with only two statistical moments will not be able to characterize this information.

3.2 Polynomial leakages

The previous subsection provided IT curves under the assumption that the implementation constraint mentioned in introduction is fulfilled. Since such a constraint may be difficult to verify in practice, we now investigate the consequences of a leakage function deviating from purely linear. For this purpose, we replace the previously used Hamming weight leakage function by a polynomial of higher degree. Such a polynomial is of the form $\mathsf{L}(s) := \sum_i a_i s_i + \sum_i \sum_j b_{i,j} s_i \times s_j + \sum_i \sum_j \sum_k c_{i,j,k} s_i \times s_j \times s_k$, where s_i denotes the ith bit of the sensitive value s, and a_i, $b_{i,j}$ and $c_{i,j,k}$ are some constants. For simplicity, we will consider the case where $\forall i\ a_i = a \in \{0,1\}$, $\forall i,j\ b_{i,j} = b \in \{0,1\}$ and $\forall i,j,k\ c_{i,j,k} = c \in \{0,1\}$.

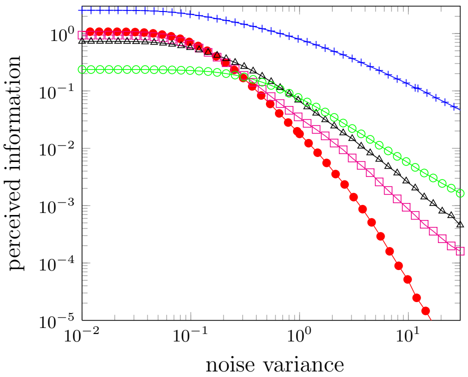

Fig. 2: IT analysis for polynomial leakage functions and LEMS with C_{16}. The curve (—●—) is for the Hamming weight leakage function. The curve (—□—) is for the leakage function with a=0, b=1 and c=0. The curve (—○—) is for the leakage function with a=0, b=0 and c=1. The curve (—△—) is for the leakage function with a=1, b=1 and c=1. The curve (—+—) is for the unprotected case in the previous subsection.

The results of our investigations in this advanced context are plotted in Figure 2. The main conclusion is that the security guarantee claimed by LEMS does not hold in this case. Interestingly, we can even observe a relation between the degree of the leakage function polynomial and the security order. Namely, the higher the degree, the lower the order - see, e.g. curves (—□—,—○—,—△—). This relation can be explained as follows. Say the leakage corresponding to $s \oplus M$ in the LEMS countermeasure only contains information in its fourth-order moment (as for C_{16}). Since M is not uniform, we know that raising this leakage to the fourth power, i.e. computing $(\mathsf{L}_{\text{lin}}(s \oplus M) + N_1)^4$ will lead to first-order information, while raising the noise to the fourth power as well. Say now the leakage function is not linear anymore, but quartic. Then the same first-order information will be found in samples of the form $\mathsf{L}_{\text{quart}}(s \oplus M) + N_1$, i.e. without amplifying the noise. More generally, if the leakage function only contains terms of a single degree, the security order of LEMS will be divided accordingly. For example, the curve (—○—) for which L has degree 3 has slope 4/3, the curve (—□—) for which L has degree 2 has slope 4/2=2, ... As for leakage functions with terms of various degrees, the situation is intermediate, e.g. the curve (—△—) for which L has degree 3 but contains terms of degree 1 and 2, has slope between the previous ones.

4 Security analysis of LEMS

We now confirm the previous IT evaluations with security analyses. For this purpose, we compute 1st-order success rates (as defined in [12]) estimated over 10000 independent experiments, in various scenarios. These results aim to translate information leakages into a number of measurements to recover the key. Note that higher-order success rates could be considered as well (to express the tradeoff between time and data complexities in side-channel attacks). However, they do not reveal more intuition regarding the security of LEMS vs. masking.

4.1 Univariate attacks

Our first experiments correspond to univariate template attacks with different noise levels, and are given in Figure 3. A preliminary observation is that, as in the previous section, Gaussian templates are not able to exploit information in this case (i.e. only Gaussian mixture models lead to successful key recoveries). Next and more importantly, the two parts of the figure clearly illustrate that the impact of estimating higher-order statistical moments in masking and LEMS mostly reveals itself as noise increases (as already highlighted in [13]). That is, the difference between the success rates attacking an unprotected implementation vs. LEMS with C_{12} or C_{16} is more significant in the right part of the figure. This confirms the information theoretic evaluations in the previous section, where the slope of the different curves also becomes stable as noise increases.

As additional experiment, we also wanted to test the usual intuition that the success rate of a template attack is highly correlated with the information leakage measured with the PI estimated thanks to the same (here Gaussian mixture) leakage model. For this purpose, it is interesting to observe that the IT curves corresponding to LEMS with C_{12} and C_{16} intersect in the left part of Figure 1. Therefore, we launched template attacks against these two countermeasures, with noise variance just left ($\sigma^2 = 0.4$) and right ($\sigma^2 = 0.5$) of this intersection. The results of these attacks are plotted in Figure 4, were we indeed observe that the success rate is slightly higher (resp. lower) when using codes C_{12} and C_{16},

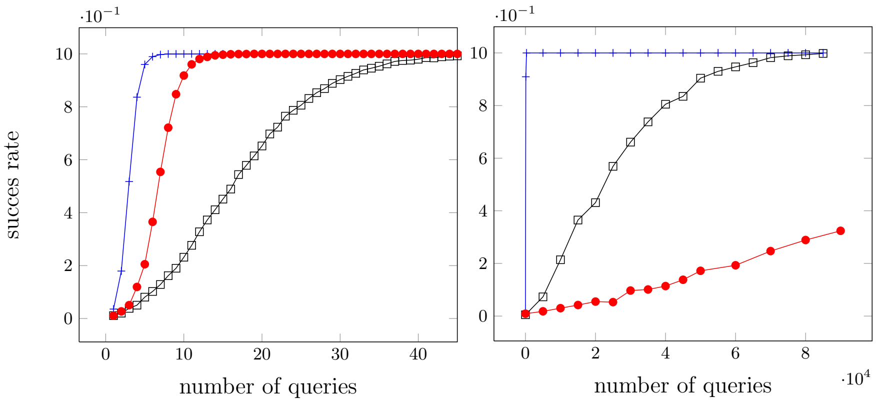

Fig. 3: Univariate template attacks with Gaussian mixture leakage model. Left: σ^2=10^{-4}. Right σ^2=10. The curves (—□—) are for LEMS with C_{12}. The curves (—●—) are for LEMS with C_{16}. The curves (—+—) are for the unprotected implementation.

depending on the noise. That is, LEMS with C_{16} delivers more information at low noise levels, but has higher security order, and consequently becomes less informative when enough noise is present in the measurements.

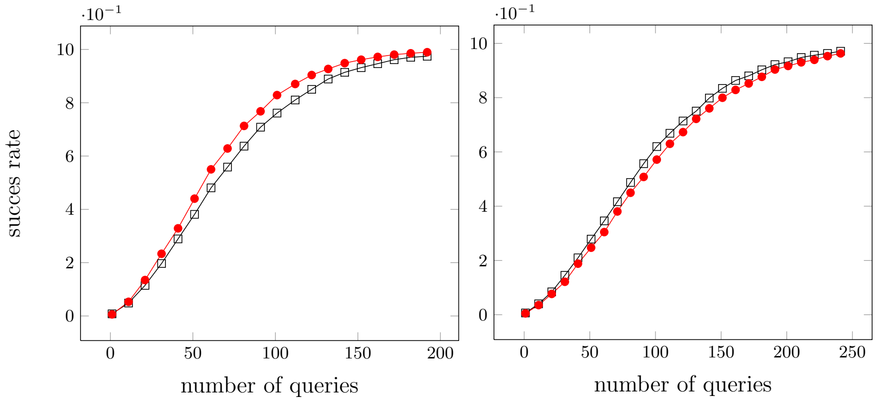

Fig. 4: Univariate template attacks with Gaussian mixture leakage model. Left: σ^2=0.4. Right σ^2=0.5. The curves (—□—) and (—●—) are for LEMS with C_{12} and C_{16}.

4.2 Bivariate attacks

To conclude this work, we also paid attention to the efficiency of bivariate template attacks with Gaussian mixture modeling, as reported in Figure 5. Here, the most revealing feature is that, as already indicated by the information theoretic analysis in the right part of Figure 1, both LEMS and masking with the full set have the same security order. As a result, the impact of noise on the separation between the success rate curves is the opposite of the one in the previous subsection. Namely, as noise increases, these curves get closer. This effect is particularly significant in attacks using Gaussian modeling, i.e. curves (—○—) - because it implies a significant loss of information for low noise values (see Figure 1). Besides, and as they all correspond to the estimation of a second-order moment in the leakage probability distribution, the data complexity of these attacks is naturally lower than the one when considering univariate attacks against LEMS with C_{12} and C_{16} in Figure 3. This eventually confirms that while LEMS indeed provides interesting security guarantees against univariate attacks, their worst-case security level is only obtained by analyzing bivariate ones.

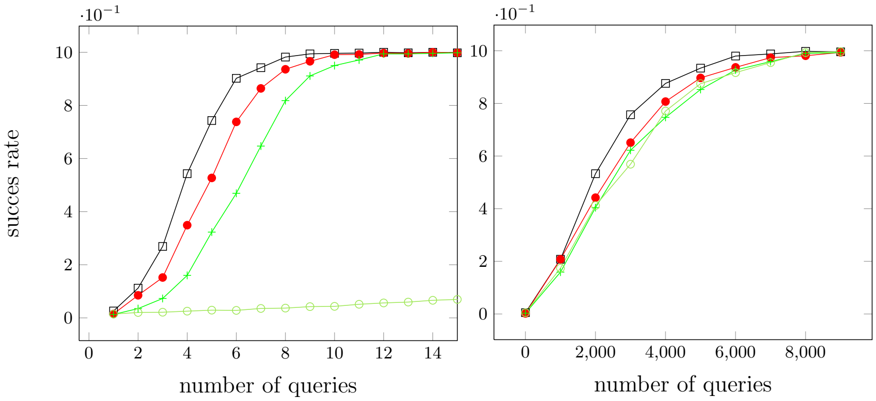

Fig. 5: Bivariate template attacks with Gaussian mixture leakage model. Left: σ^2=10^{-4}. Right σ^2=10. The curves (—□—) are for LEMS with C_{12}. The curves (—●—) are for LEMS with C_{16}. The curves (—○—) and (—+—) are for masking with the full set, using Gaussian and Gaussian mixture leakage modeling, respectively.

Wrapping up

The consequences of our analysis for LEMS are contrasted. First, while its adversarial condition may not always be practically relevant, the investigations in Section 3.1 and 4.2 suggest that the countermeasure remains an interesting alternative to mask with reduced randomness requirements, even if adversaries exploit bivariate leakages (as there is no penalty for the security order in this case). By contrast, the observations in Section 3.2 suggest that the security of LEMS is highly dependent on the (hard to control) leakage function. In particular, the apparition of higher-degree terms in this function directly implies an exploitable penalty in the security order of the countermeasure.

Acknowledgments. Work funded in parts by the European Commission through the ERC project 280141 (acronym CRASH) and the European ISEC action grant HOME/2010/ISEC/AG/INT-011 B-CCENTRE project. F.-X. Standaert is an associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.).

References

1. Shivam Bhasin, Claude Carlet, and Sylvain Guilley. Theory of masking with codewords in hardware: low-weight dth-order correlation-immune boolean functions. Cryptology ePrint Archive, Report 2013/303, 2013. `http://eprint.iacr.org/`.
2. Claude Carlet, Jean-Luc Danger, Sylvain Guilley, and Houssem Maghrebi. Leakage squeezing of order two. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT*, volume 7668 of *LNCS*, pages 120–139. Springer, 2012.
3. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *LNCS*, pages 398–412. Springer, 1999.
4. Jean-Sébastien Coron, Emmanuel Prouff, and Matthieu Rivain. Side channel cryptanalysis of a higher order masking scheme. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES*, volume 4727 of *LNCS*, pages 28–44. Springer, 2007.
5. Vincent Grosso, François-Xavier Standaert, and Sebastian Faust. Masking vs. multiparty computation: How large is the gap for AES? In Guido Bertoni and Jean-Sébastien Coron, editors, *CHES*, volume 8086 of *LNCS*, pages 400–416. Springer, 2013.
6. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *LNCS*, pages 463–481. Springer, 2003.
7. Houssem Maghrebi, Sylvain Guilley, and Jean-Luc Danger. Leakage squeezing countermeasure against high-order attacks. In Claudio A. Ardagna and Jianying Zhou, editors, *WISTP*, volume 6633 of *LNCS*, pages 208–223. Springer, 2011.
8. Maxime Nassar, Sylvain Guilley, and Jean-Luc Danger. Formal analysis of the entropy / security trade-off in first-order masking countermeasures against side-channel attacks. In Daniel J. Bernstein and Sanjit Chatterjee, editors, *INDOCRYPT*, volume 7107 of *LNCS*, pages 22–39. Springer, 2011.
9. Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *LNCS*, pages 142–159. Springer, 2013.
10. Emmanuel Prouff, Matthieu Rivain, and Régis Bevan. Statistical analysis of second order differential power analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009.
11. Mathieu Renauld, François-Xavier Standaert, Nicolas Veyrat-Charvillon, Dina Kamel, and Denis Flandre. A formal study of power variability issues and side-channel attacks for nanoscale devices. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *LNCS*, pages 109–128. Springer, 2011.
12. 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 *LNCS*, pages 443–461. Springer, 2009.
13. François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *LNCS*, pages 112–129. Springer, 2010.