IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2022, No. 4, pp. 774–799. DOI:10.46586/tches.v2022.i4.774-799

Side-Channel Expectation-Maximization Attacks

Julien Béguinot[1,2], Wei Cheng[1,2], Sylvain Guilley[2,1], and Olivier Rioul[1]

[1] LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
firstname.lastname@telecom-paris.fr
[2] Secure-IC S.A.S., Paris, France, sylvain.guilley@secure-ic.com

Abstract. Block ciphers are protected against side-channel attacks by masking. On one hand, when the leakage model is unknown, second-order correlation attacks are typically used. On the other hand, when the leakage model can be profiled, template attacks are prescribed. But what if the profiled model does not exactly match that of the attacked device?
One solution consists in regressing on-the-fly the scaling parameters from the model. In this paper, we leverage an Expectation-Maximization (EM) algorithm to implement such an attack. The resulting unprofiled EM attack, termed U-EM, is shown to be both efficient (in terms of number of traces) and effective (computationally speaking). Based on synthetic and real traces, we introduce variants of our U-EM attack to optimize its performance, depending on trade-offs between model complexity and epistemic noise. We show that the approach is flexible, in that it can easily be adapted to refinements such as different points of interest and number of parameters in the leakage model.

Keywords: Side-Channel Analysis · Masked Cryptography · Maximum Likelihood Distinguisher · Leakage Model Regression · Expectation Maximization (EM) · Unprofiled EM (U-EM) Attack · Epistemic Noise.

1 Introduction

Embedded devices compute cryptographic algorithms to secure data, in particular when it is transmitted through networks. Such devices are at risk against attackers attempting to extract their cryptographic keys. Attackers can leverage so-called side-channel emanations to recover the secret keys: Side-channel measurements are processed by statistical tools to determine which key guess best matches the correct key. Since guessing the whole key at once is computationally infeasible, the key can be targeted byte by byte in a divide-and-conquer approach [KJJ99].

Three strategies, depending on the prior knowledge of the leakage, can be implemented:

1. In the first strategy, a leakage model is assumed, based on the physical characterization of the device emanations during computation. Then, a *correlation analysis* [BCO04] is carried out to determine under which key byte the model best matches the observed leakage traces.
2. Even though the leakage model leveraged by correlation analysis can be profiled [WO13], when profiling is possible, it is best to resort to *template attacks* [CRR02]. In fact, the assumed leakage model can be quite crude. Thus, in this case, the best strategy consists in profiling the leakage first, and then apply a maximum likelihood distinguisher, which is known to be theoretically optimal. It is also akin to modern machine learning attacks. However, template attacks can only be optimal as long as templates are optimally profiled: In case of a mismatch between the actual attacked device and the profiled one, the attack's efficiency can be greatly reduced, or even compromised. As pointed out in [BHM+19] it is also necessary that the leakage model be rich enough to capture the actual leakage. It has been warned that persisting with template attacks under

Received: 2022-04-15 Accepted: 2022-06-15 Published: 2022-08-31

this condition is not only inefficient, but also misleading, because they yield a false confidence in the security level [BL20].

3. Fortunately, there is yet another way to go. This third approach is known as *stochastic attack* [SLP05]. It consists in assuming a parametric model, where the unknown parameters accommodate on-the-fly to the leakage peculiarities of the attacked device. Such peculiarities can result from many factors, such as process variability during device manufacturing [WYS$^+$18], different measurement systems, changing environmental conditions [KGD18], etc. In [SKM$^+$13], stochastic attacks are shown to be equivalent to linear combinations of correlation analyses. In particular, stochastic attacks would reduce to correlation analyses if the attacker knows the weights perfectly (which is generally not the case).

The comparison between those methods is the topic of several papers, such as [GLRP06, SKM$^+$13] for the 2nd vs. 3rd methods, or [dCGHR18] for the 1st vs. 2nd methods.

In practice, cryptographic devices should not be left unprotected and designers implement countermeasures to mitigate attempts to extract the key. Masking intermediate variables randomly in the cryptographic algorithm is one well-known protection [CJRR99, PR13]. In this paper, we assume that this masking protection has been implemented. Therefore, the three abovementioned attack methodologies should be adapted accordingly:

1. The correlation analysis is adapted to a higher-order setting. In a 2-share masking, an adversary should now combine two samples from the leakage traces: One during handling of the masked variable and another during handling of the mask itself. Such bivariate attack is referred to as "second-order correlation analysis" [PRB09]. Interestingly, [DPRS11] shows that standard univariate distinguishers with linear leakages are equivalent to CPA. However, it is shown in [SVO$^+$10] that not all multivariate distinguishers can be reduced to CPA and the optimal multivariate distinguishers usually outperform CPA (hence CPA is no longer optimal).

2. Two flavors of template attacks can be performed. The first one consists in profiling *not* knowing the masks, so as to extract a meaningful model [OM07]. For instance, machine learning based techniques like kernel density estimation (KDE), random forests (RF) and neural-network (NN) based attacks are utilized in [LM18, MDP20, LBM15]. It is shown that RF and NN-based attacks perform well on high-dimension leakages and noisy measurements. Another, more efficient approach, consists in profiling *knowing* the masks, which yields so-called high-order optimal distinguisher (HOOD) [BGHR14] of second-order.

3. In this paper, we explore the third option: How to leverage unsupervised parametric models without a prior profiling phase in the presence of masking? This approach is for instance applied in [WYS$^+$18], though in an unprotected setting. Besides, remark 2 of [SLP05] evokes that regression could be carried on-the-fly, without providing an explicit method. The problem remains as to profile masked traces on-the-fly. In [DPRS11], the case of univariate leakage is addressed, but the problem remains in multivariate contexts. We bridge the gap and solve the problem with an unprofiled Expectation-Maximization (U-EM) attack[1]. Our aim is to tackle situations where deviations between learned traces and attacked traces are too large to make HOOD attacks effective.

State-of-the-Art. Side-channel analysis is a mature field, where multiple approaches have been proposed and validated:

[1] In this article, we use "EM" as an abbreviation for "*Expectation Maximization*", as done in related papers, such as [LRP07]. This term shall not be confused with "*Electro-Magnetic*", which is also a usual abbreviation in the field of side-channel analysis.

- Profiled EM [LRP07] (abridged P-EM in the rest of this paper) is an unsupervised profiled attack. As in our proposal, it does not require the knowledge of the masks. Yet it requires a profiling phase for all key hypotheses to estimate the probability density function of a parameterized Gaussian mixture. In [LRP07], the EM algorithm is only leveraged statically to determine the leakage model, assumed to be the same for the subsequently attacked device. More recently, [BDMS22] also investigated the P-EM in a similar fashion.
- In ridge-based DPA [WYS$^+$18], the authors suggest using ridge regression and on-the-fly regression of the stochastic model as distinguishers.
- In [WOS14], the notion of generic DPA attack is defined. A *stepwise* regression algorithm is derived. This algorithm enables the attacker to select the best elements in a basis for a stochastic attack to succeed.

Contributions. We introduce the U-EM attack with the following features:

- It is an unsupervised unprofiled attack, in that it does not require to measure leakage from a cloned device;
- It applies in a “white box” context where the attacker knows the cryptographic software code, yet she/he does not have access to the exact hardware characteristics. In particular, the leakage model is only approximately known, and the position in time of points of interest (PoIs) can be inferred by studying the software code.

The open sources of our U-EM and other distinguishers are publicly available on `GitHub`[2]. The differences with related attacks are as follows:

- Contrary to P-EM [LRP07, BDMS22], we cannot learn on a clone device. Especially we do not have a dataset with all key hypothesis realizations to perform profiling.
- Contrary to ridge-based DPA [WYS$^+$18], we focus on how to actually perform on-the-fly regression for masked implementations without a profiling phase.
- Contrary to the proposal of [WOS14], we consider protected implementations where online linear regression methods would fail for unknown masks. The stepwise regression algorithm presented in [WOS14] could be used together with the Expectation Maximization algorithm derived in this article to perform unsupervised unprofiled attacks with best choice of the basis of the stochastic attack. In that sense, these works are complementary.

However, it should be noticed that our attack also faces the problems of PoIs selection and the tradeoff between model accuracy and overfitting.

Outline. Our framework is described in Section 2: we consider (a) a simple Hamming weight model on two shares and (b) a parametric model, where each bit leaks on its own. Section 3 recalls the theoretical background on the optimal distinguisher, and compares it to various known distinguishers. A specific emphasis on our proposed U-EM distinguisher is carried out in Section 4. The results on synthetic traces provided in Section 5 show that the U-EM attack improves significantly the performance over the state-of-the-art attacks (e.g., 2O-CPA). However, it eventually reduces to 2O-CPA for large measurement noise. A comparison with P-EM is also carried out in Section 5. In Section 6 we show on real world traces that the U-EM attack can advantageously be tuned to explicitly take the epistemic noise into account. The situations where our attack is particularly relevant are discussed in Section 7. Section 8 concludes and gives some perspectives.

[2] The `GitHub` repository containing the code for U-EM is: https://github.com/JulienBeg/U-EM.

2 Attack Scenario Description

2.1 Terminology

Our U-EM attack is "stochastic" for it can process a parametric leakage model.

Definition 1. A leakage model is:

- *parametric*, if there are a finite number of parameters to estimate. However, there are several cases, depending on the number of parameters. Here we have either one parameter per bit or one parameter per byte of the sensitive variable.
- *non-parametric*, if there is an infinite number of parameters to estimate. Typically, it requires a direct estimation of probability density functions (PDFs) by binning or kernel density estimation.

In that sense, the CPA is a parametric attack with two parameters (scaling and offset).

Definition 2. A distinguisher is said to be:

- *profiled*, if it requires a prior dataset collected from an other instance of the device to learn how it leaks;
- *non-profiled*, otherwise.

Eventually, as discussed in [LRP07], we distinguish *supervised* from *unsupervised* distinguishers.

Definition 3. A distinguisher is said to be:

- *supervised*, if it requires a dataset of labeled data (typically, labels are all intermediate sensitive variables) in a learning phase before actually carrying out the attack;
- *unsupervised*, if it can resort only to a dataset of unlabelled data (raw traces) to carry out the attack.

In the sequel, all supervised attacks are profiled (e.g., template attacks). The P-EM attack performed in [LRP07] is unsupervised but requires a profiling phase. A generic distinguisher (e.g., 2O-CPA, MIA/KSA) is unsupervised and does not require a profiling phase. Our U-EM attack proposal is unsupervised and non-profiled. Table 1 summarizes this classification. 2O-CPA falls in the same category as our U-EM attack. However, 2O-CPA can only tolerate two free parameters (scale and offset), whereas U-EM supports an unbounded number of parameters (e.g., 4 and 18 as described in the next Subsection).

Table 1: Assumptions for several distinguishers.

Properties	Template	P-EM	Proposed U-EM	2O-CPA	MIA/KSA
	[CRR02]	[LRP07]	[This paper]	[PRB09]	[GBTP08]/[WOM11]
Supervised	✓	×	×	×	×
Profiling phase	✓	✓	×	×	×
Parametric model	✓	✓	✓	✓	×

2.2 Three Leakage Models

Consider a masked block cipher implementation with an n-bit secret key k that encrypts Q plaintext bytes $\mathbf{t} = (t_1, t_2, \ldots, t_Q)$ with different random masks $\mathbf{M} = (M_1, M_2, \ldots, M_Q)$, which are independently drawn uniformly in $\mathbb{F}_2^n$. The masks are represented as binary vectors of $(\mathbb{F}_2)^n$ in the registers. The sensitive variable $\mathbf{X} = (X_1, X_2, \ldots, X_Q)$ is such that

$$X_q = S(k \oplus t_q) \oplus M_q \qquad (q = 1, 2, \ldots, Q) \tag{1}$$

where S is the substitution box used by the encryption algorithm.

The attacker measures Q traces under some measurement noise, which is commonly assumed to be additive white Gaussian. In this paper, we consider three different bivariate leakage models giving $\mathbf{Y} = (Y_1, \ldots, Y_Q)$ where for each $q \in \{1, 2, \ldots, Q\}$ one has $Y_q = (Y_q^{(1)}, Y_q^{(2)})$ with noise variance $\sigma^2 = (\sigma_1^2, \sigma_2^2)$.

❶ **Hamming Weight Leakage Model.** Here we assume that all bits in one variable leak similarly. As a result, Y_q is given by

$$\begin{cases} Y_q^{(1)} &= a^{*,(1)} w_H(X_q) + b^{*,(1)} + N_q^{(1)} \\ Y_q^{(2)} &= a^{*,(2)} w_H(M_q) + b^{*,(2)} + N_q^{(2)} \end{cases} \qquad (q = 1, 2, \ldots, Q) \tag{2}$$

where $a^{*,(i)} \in \mathbb{R}$ and $b^{*,(i)}$ are unknown parameters.

❷ **Linear Leakage Model.** Here we expect the different bits in the registers to leak differently due to some technological dispersion. As a result, Y_q is now given by

$$\begin{cases} Y_q^{(1)} &= \langle a^{*(1)}, X_q \rangle + b^{*(1)} + N_q^{(1)} \\ Y_q^{(2)} &= \langle a^{*(2)}, M_q \rangle + b^{*(2)} + N_q^{(2)} \end{cases} \qquad (q = 1, 2, \ldots, Q) \tag{3}$$

where $\langle \cdot, \cdot \rangle$ denotes a bitwise scalar product over the reals and where vector $a^{*(i)} \in \mathbb{R}^n$ and $b^{*(i)} \in \mathbb{R}$ are unknown parameters. The Hamming weight leakage model (2) is recovered for constant vectors $a^{*(1)} = (a^{(1)}, \ldots, a^{(1)})$ and $a^{*(2)} = (a^{(2)}, \ldots, a^{(2)})$.

❸ **Quadratic Leakage Model.** In this model bits can interact with each other with terms up to order two as follows:

$$\begin{cases} Y_q^{(1)} &= X_q^\top a^{*(1)} X_q + b^{*(1)} + N_q^{(1)} \\ Y_q^{(2)} &= M_q^\top a^{*(2)} M_q + b^{*(2)} + N_q^{(2)} \end{cases} \qquad (q = 1, 2, \ldots, Q), \tag{4}$$

where we have $x^2 = x$, for $x \in \mathbb{F}_2$. Formally this is a quadratic form with upper triangular (or symmetric) matrices. It is worth mentioning that if the matrices $a^{*(1)}$ and $a^{*(2)}$ are diagonal then we recover the linear leakage model as in (3).

In order to simplify notations and unify derivations for both cases at the same time, we let $x(a, b, k, t, m)$ denote the noiseless observation one would have if the parameters a^*, b^*, k^* equal a, b, k, respectively, for plaintext t and mask m. Thus,

$$\begin{cases} x^{(1)}(a, b, k, t, m) = a \cdot w_H(m) + b \\ x^{(2)}(a, b, k, t, m) = a \cdot w_H(S(k \oplus t) \oplus m) + b \end{cases} \tag{5}$$

for the Hamming weight leakage model,

$$\begin{cases} x^{(1)}(a, b, k, t, m) = \langle a, m \rangle + b \\ x^{(2)}(a, b, k, t, m) = \langle a, S(k \oplus t) \oplus m \rangle + b \end{cases} \tag{6}$$

for the linear leakage model, and

$$\begin{cases} x^{(1)}(a, b, k, t, m) = m^\top a m + b \\ x^{(2)}(a, b, k, t, m) = (S(k \oplus t) \oplus m)^\top a (S(k \oplus t) \oplus m) + b \end{cases} \tag{7}$$

for the quadratic leakage model.

In the sequel, the average of any given vector $\mathbf{v} = (v_1, \ldots, v_Q)$ is denoted by $\overline{\mathbf{v}} = \frac{1}{Q} \sum_{q=1}^{Q} v_q$ and the resulting centered vector is denoted by $\widetilde{\mathbf{v}} = (v_1 - \overline{\mathbf{v}}, \ldots, v_Q - \overline{\mathbf{v}})$.

3 Comparison of Various Distinguishers

3.1 Theoretically Optimal Distinguisher

The theoretical expression of the optimal maximum likelihood distinguisher [HRG14b] is

$$\hat{k}(\mathbf{y}) = \arg\max_{k,a,b} \mathbb{P}(\mathbf{Y} = \mathbf{y}|k, a, b). \tag{8}$$

To simplify notation, let us assume that $Y^{(1)}$ and $Y^{(2)}$ have been normalized (divided by σ_1 and σ_2, respectively), so that the noise simplifies to the standard Gaussian. Then the likelihood takes the form

$$\begin{aligned}\mathbb{P}(\mathbf{Y}{=}\mathbf{y}|k, a, b) = \prod_{q=1}^{Q} \mathbb{P}(Y_q{=}y_q|k, a, b) &= \prod_{q=1}^{Q} \sum_{m_q \in \mathbb{F}_2^n} \mathbb{P}(M_q{=}m_q)\mathbb{P}(Y_q = y_q|k, a, b, M_q{=}m_q)\\ &\propto \prod_{q=1}^{Q} \sum_{m_q \in \mathbb{F}_2^n} \exp\Big(-\frac{1}{2}\|y_q - x(a, b, k, t_q, m_q)\|^2\Big). \end{aligned}\tag{9}$$

It is more convenient to maximize the log-likelihood:

$$\mathcal{LL}(\mathbf{Y} = \mathbf{y}|k, a, b) \propto \sum_{q=1}^{Q} \log\Big[\sum_{m_q \in \mathbb{F}_2^n} \exp\Big(-\frac{1}{2}\|y_q - x(a, b, k, t_q, m_q)\|^2\Big)\Big]. \tag{10}$$

The ML-based distinguisher is then

$$\hat{k}(\mathbf{y}) = \arg\max_{k,a,b} \sum_{q=1}^{Q} \log\Big[\sum_{m_q \in \mathbb{F}_2^n} \exp\Big(-\frac{1}{2}\|y_q - x(a, b, k, t_q, m_q)\|^2\Big)\Big]. \tag{11}$$

It is important to note that there does not exist a closed-form expression to find out a, b and k. On the other hand, a direct maximization in a, b is tedious or even infeasible in practice. It is for these reasons that we suggest to use the EM algorithm.

3.2 2O-CPA with Centered Product Combination

A state-of-the-art unsupervised attack on bivariate leakage with first-order masking is the *second-order correlation power analysis* with centered product combination on leakage of different shares (see [PRB09, MCGD12]). This is simply a classical CPA is applied to the centered product of the different samples. For a given key hypothesis k, we write $\mathbf{x}(k) = (X(k)_1, \dots, X(k)_Q)$ where

$$x(k)_q = \frac{1}{2^n} \sum_{m \in \mathbb{F}_2^n} w_H(m) w_H(S(k \oplus t_q) \oplus m) \qquad (q = 1, 2, \dots, Q). \tag{12}$$

The distinguisher is then

$$\hat{k}_{\text{2O-CPA}}(\mathbf{y}) = \arg\max_{k} |\rho(\mathbf{x}(k), \widetilde{\mathbf{y}^{(1)}}\widetilde{\mathbf{y}^{(2)}})| = \arg\max_{k} |\frac{\mathsf{Cov}(\mathbf{x}(k), \widetilde{\mathbf{y}^{(1)}}\widetilde{\mathbf{y}^{(2)}})}{\sigma_{\mathbf{x}(k)}\sigma_{\mathbf{y}^{(1)}\mathbf{y}^{(2)}}}| \tag{13}$$

where ρ is the empirical Pearson correlation coefficient. The guessed key is the one which is the most correlated with the leakages. CPA is well adapted for the Hamming weight leakage model, but applying it to linear leakage model leads to a model mismatch which will be shown to limit its performance.

3.3 Maximum Likelihood with Templates

For the maximum likelihood with templates, we assume that the parameters a^* and b^* are known. Hence the maximum likelihood (8) can be directly computed from the traces. This is unrealistic as these parameters are unknown in practice. Indeed, their knowledge requires a profiling on a identical device where the masks are also known; our attack scenario makes no such assumption. This distinguisher is used to compute an upper bound on the success rate of the other distinguishers. Its expression simplifies to

$$\hat{k}_{\mathrm{ML}} = \arg\max_{k} \mathbb{P}(\mathbf{Y} = \mathbf{y} | k, a^*, b^*). \tag{14}$$

3.4 Expectation-Maximization Algorithm

While (8) can be evaluated naively (though inefficiently) with a general purpose minimizer—such as the Nelder-Mead algorithm provided in the `scipy.minimize` Python library—the *Expectation-Maximization* (EM) is a fast, iterative algorithm proposed by Dempster et al. [DLR77] that can be used to compute maximum log-likelihood in the context of hidden or unobserved variables (the masks). We detail its application to the unsupervised EM attack (U-EM) in the next Section.

3.5 Comparison with the State-of-the-Art

We now provide a critical overview of existing distinguishers for our scenario:

- HO-CPA [BGNT18, §4.1] has two advantages: it is efficient computationally and it is easy to implement in practice, because it involves a simple combination function without parameter estimation. This, however, implies two limitations. First of all, it is only optimal in the "ratio" level (in Stevens' typology), i.e., if the model is known up to a scaling factor. But when bits leak differently, the HO-CPA looses its advantage for high epistemic noise. Second, the combination function at the core of HO-CPA does not yield optimal success rate, as it is only an approximation of a HOOD [BGHR14].

- HO-DPA [Mes00, GBPV10] does not suffer from the problem stated above, but is limited to capture only the leakage from one selected bit, whereby HO-CPA captures leakage of all bits of the sensitive data.

- MIA [GBTP08] makes no assumption on the leakage model, but has a structural weakness since it can be successful only if the model is non-injective. As a result, the model should often be made weaker (e.g., by ignoring bits). Also, the main difference between MIA and our proposal (U-EM) is that MIA is non-parametric. It requires, as a first step, to build PDFs on-the-fly, from many measurements. Additionally, it requires the selection of a binning strategy, resulting in information loss. Owing to binning, MIA also faces the issue of empty bins.

- KSA [WOM11, HRG14a] shares with MIA the fact that it is non-parametric, and the requirement to estimate the leakage (under the form of a cumulative density function (CDF) whereby MIA uses a PDF). A priori, the CDF can be estimated without binning, hence it is easier to implement, and requires less measurements (no info loss from the binning, hence more faithful model). It is important to acknowledge that both MIA and KSA can outperform our proposal in the case of a model mismatch, owing to their resiliency in the case the model features non-linear combinations of bits.

4 Applying EM Algorithm to Side-Channel Attacks

The EM algorithm is made up of two main steps: The expectation step (E-Step) and the maximization step (M-Step). One iteratively (a) takes the expectation over the masks given the last value of the parameters that have been computed and (b) maximizes the expression in the parameters to update them. The algorithm stops when a given convergence criterion is achieved. We initialize the value of a_0 and b_0 arbitrarily with a_0 = all-one vector and $b_0 = 0$, and at each iteration p build (a_p) and (b_p) recursively using the formula

$$a_{p+1}, b_{p+1} \longleftarrow \underbrace{\arg\max_{a,b}}_{\text{M-Step}} \underbrace{\mathbb{E}_{\mathbf{M}\sim\mathcal{U}(\mathbb{F}_2^n)^Q}\left[\log(\mathbb{P}(\mathbf{Y}=\mathbf{y},\mathbf{M}|k,a,b))\right]}_{\text{E-Step}}. \tag{15}$$

Assume we have fixed a key hypothesis k and a trace index $q \in \{1,\dots,Q\}$. If we denote $\beta_q^{(p)}(m) = \mathbb{P}(M_q = m)\exp\Big(-\frac{1}{2}\|y_q - x(a_p,b_p,k,t_q,m)\|^2\Big)$ and $\alpha_q^{(p)}(m) = \frac{\beta_q^{(p)}(m)}{\sum_{m'}\beta_q(m')^{(p)}}$, we have $\mathbb{P}(M_q = m|y,a_p,b_p) = \alpha_q^{(p)}(m)$. The coefficient $\alpha_q^{(p)}(m)$ is the Bayes posterior probability of the mask m being used for the q-th traces given a_p, b_p. Over the iteration as a_p, b_p get closer to a^*, b^*, we also improve our guess on which mask has been used.

4.1 Convergence

We recall the main convergence result:

Theorem 1. *The EM algorithm converges to a stationary point of the log-likelihood.*

Proof. One has

$$\log\frac{\mathbb{P}(\mathbf{Y}=\mathbf{y}|k,a_n,b_n)}{\mathbb{P}(\mathbf{Y}=\mathbf{y}|k,a_{p+1},b_{p+1})} = \sum_{q=1}^{Q}\log\sum_m \alpha_q^{(p)}(m)\frac{\exp\Big(-\frac{1}{2}\|y_q - x(a_p,b_p,t_q,k,m)\|^2\Big)}{\exp\Big(-\frac{1}{2}\|y_q - x(a_{p+1},b_{p+1},k,t_q,m)\|^2\Big)}$$

which by Jensen inequality is not less than

$$\frac{1}{2}\sum_{q=1}^{Q}\sum_m \alpha_q(m)\Big(\|y_q - x(a_{p+1},b_{p+1},k,t_q,m)\|^2 - \|y_q - x(a_p,b_p,k,t_q,m)\|^2\Big). \tag{16}$$

Therefore, at each M-step, the log-likelihood can only increase. As it is bounded it should converge. But at convergence, the gradient vanishes so that $(a)_n, (b)_n$ will also converge. □

Remark 1. Note, however, that in theory, the stationary point can be a local maximum, a saddle point or even a local minimum.

4.1.1 Explicit E-Step

The E-Step at the p-th iteration is derived at follows.

$$\begin{aligned}\mathbb{E}[\log(\mathbb{P}(\mathbf{Y}=\mathbf{y},\mathbf{M}|a,b))] &= \mathbb{E}\,[\sum_q \log(\mathbb{P}(Y_q = y_q, M_q|a,b))]\\ &= \sum_q\sum_{m_q}\mathbb{P}(M_q = m_q|Y_q = y_q,a_p,b_p)\log(\mathbb{P}(Y_q = y_q, M_q = m_q|a,b))\\ &= \sum_{q,m}\alpha_q^{(p)}(m)\log(\mathbb{P}(Y_q = y_q|M_q = m,a,b)) + cst\end{aligned} \tag{17}$$

where the constant cst is independent of a and b. Thus the E-Step of the EM reduces to

$$(a_{p+1},b_{p+1}) = \arg\min_{(a,b)}\sum_q\sum_{m_q}\alpha_q^{(p)}(m_q)\|y_q - x(a,b,k,t_q,m_q)\|^2. \tag{18}$$

4.1.2 Explicit M-Step in the Hamming Model Case

To simplify the presentation and reduce computation, we consider centered traces where $\overline{\mathbf{y}}^{(i)} = 0$ for $i \in \{1, 2\}$. After the E-Step, the minimization problem is a linear regression problem. Define $x_{k,t,m} = (w_H(S(k \oplus t) \oplus m), w_H(m))$ and $\overline{\mathbf{x}}^{(i)} = \frac{1}{Q} \sum_{q,m} \alpha_q(m) x^{(i)}_{k,t_q,m} \in \mathbb{R}^n$, $i = 1, 2$. Then the empirical covariance is $\widehat{\mathsf{Cov}^{(i)}_{\mathbf{xy}}} = \frac{1}{Q} \sum_{q,m} \alpha_q(m) y^{(i)}_q x^{(i)}_{k,t_q,m}$ and the empirical variance is $\widehat{\mathsf{Var}^{(i)}_{\mathbf{x}}} = \frac{1}{Q} \sum_{q,m} \alpha_q(m) y^{(i)2}_q$.

Proposition 1. *The M-Step is given by the following update rule ($i = 1, 2$):*

$$a^{(i)} = \frac{\widehat{\mathsf{Cov}^{(i)}_{\mathbf{xy}}}}{\widehat{\mathsf{Var}^{(i)}_{\mathbf{x}}}} \qquad \textit{and} \qquad b^{(i)} = -a^{(i)} \overline{\mathbf{x}}^{(i)}. \tag{19}$$

Proof. Particular case of Proposition 2 will be given next. □

4.1.3 Explicit M-Step in the Linear Model Case

Again we consider centered traces $\overline{\mathbf{y}}^{(i)} = 0$ for $i \in \{1, 2\}$ and after the E-Step, the minimization problem is a linear regression problem, that can be minimized by solving the system of *normal equations*. Define $x_{k,t,m} = (S(k \oplus t) \oplus m, m)$ and $\overline{\mathbf{x}}^{(i)} = \frac{1}{Q} \sum_{q,m} \alpha_q(m) x^{(i)}_{k,t_q,m} \in \mathbb{R}^n$. The empirical autocorrelation matrix is $\widehat{R^{(i)}_{\mathbf{xx}}} = \sum_{q,m} \alpha_q(m)(x^{(i)}_{k,t_q,m} - \overline{\mathbf{x}}^{(i)})(x^{(i)}_{k,t_q,m} - \overline{\mathbf{x}}^{(i)})^\top \in \mathbb{R}^{n \times n}$. and the empirical intercorrelation is $\widehat{R^{(i)}_{\mathbf{xy}}} = \sum_{q,m} \alpha_q(m)(x^{(i)}_{k,t_q,m} - \overline{\mathbf{x}}^{(i)}) {y^{(i)}_q}^\top \in \mathbb{R}^n$.

Proposition 2. *The M-Step is given by the following update rule ($i = 1, 2$):*

$$a^{(i)} = \left(\widehat{R^{(i)}_{\mathbf{xx}}}\right)^{-1} \widehat{R^{(i)}_{\mathbf{xy}}} \qquad \textit{and} \qquad b^{(i)} = -\langle a^{(i)}, \overline{\mathbf{x}}^{(i)} \rangle \tag{20}$$

Proof. We aim to find a, b that minimize the quantity $\sum_{q,m} \alpha_q(m) \| y_q - \langle a, x_{k,t_q,m} \rangle - b \|^2 = \mathbb{E}[\| y_q - \langle a, x_{k,t_q,m} \rangle \|^2]$. Assuming centered quantities $\mathbf{x}$ and $\mathbf{y}$, we can remove b from the equation and simply find a that minimizes $\mathbb{E}[\| y_q - \langle a, x_{k,t_q,m} \rangle \|^2]$. Expanding the scalar product one finds that this is equivalent to minimizing $\mathbb{E}[(a^\top x)^2] - 2\mathbb{E}[y a^\top x]$. The gradient in a of this quantity is $2(\mathbb{E}[xx^\top]a - \mathbb{E}[xy]) = 2(\widehat{R_{\mathbf{xx}}} a - \widehat{R_{\mathbf{xy}}})$, which vanishes for $a = \widehat{R_{\mathbf{xx}}}^{-1} \widehat{R_{\mathbf{xy}}}$. In the Hamming model case we recover the CPA equation with covariance and variance instead of autocorrelation matrix and intercorrelation (Proposition 1). □

Remark 2. Interestingly, we note that this M-step is actually equivalent to a regular CPA. While CPA cannot be performed on a masked block cipher implementation, by averaging over the masks each M-Step turns out to boil down to a classical CPA. Therefore, one can see the EM algorithm as a repetition of classical CPAs by averaging over the masks until convergence is reached.

Remark 3. The rank of the autocorrelation matrix is upper bounded by the number Q of traces. So for very few traces it may become singular. In that case, the linear system to solve is degenerated to a least square minimization problem. For such a few number of traces we suggest to work with the Hamming weight leakage model instead.

4.1.4 Explicit M-Step in the Quadratic Leakage Model

In this case, the M-Step can be efficiently implemented with a gradient based optimizer (e.g., BFGS minimizer from `scipy`). To speed up the optimization, it is useful to provide an expression for the Jacobian to the optimizer. For the term of index (i, j) in the matrix a, it is given by

$$-2\sum_{q,m}\alpha_q(m)(y_q - x(k,t_q,m)^\top a x(k,t_q,m) - b)x(k,t_q,m)_i x(k,t_q,m)_j \ . \qquad (21)$$

The implementations of the U-EM attacks are given in Algorithms 1 and 2 for the Hamming leakage model and linear leakage model, respectively. For the sake of simplicity, we omit the implementation of the quadratic leakage model, as it only differs from the other two implementations by M-step.

4.2 Tradeoff Between Epistemic Noise and Model Complexity

To model the mismatch of the Hamming weight leakage model to the linear leakage model we recall the concept of *epistemic noise*. We assume that the coefficients a_{LIN} of a share for the linear leakage model are drawn as $\mathcal{N}((a_{HAM}, \dots, a_{HAM}), \sigma_a^2 I_n)$ where a_{HAM} is the coefficient associated to the Hamming weight leakage model and $\mathcal{N}((0, \dots, 0), \sigma_a^2 I_n)$ is the epistemic noise (a static noise on the model) with standard deviation σ_a.

Our purpose is to maximize the performance of the U-EM attack depending on the noise of the measure, the model complexity and the epistemic noise. A more complex model enables the attack to fit the model even with high epistemic noise ($\sigma_a \gg \sigma$).

In a supervised context, when the model is regressed over a training set, this model complexity comes with a risk of *overfitting*, a phenomenon by which the model fits perfectly the training set but performs poorly on a test set. Indeed, the model is too complex and fits the noise of the data, typically when $\sigma_a \ll \sigma$. So in a supervised learning context the model complexity is chosen as a trade off between the epistemic noise and the noise over the measurements.

In the unsupervised context of the U-EM another problem appears that we also call *overfitting*. For each key hypothesis k in the U-EM algorithm different parameters are regressed. If the model is too complex a risk is that we can find parameters for the model that can explain very well a concurrent bad key hypothesis $\tilde{k}$. In this case a more complex model is a burden to discriminate between the different key hypothesis. In the next two sections we suggest two strategies to ensure the best trade off for the U-EM attacks.

4.3 Ridge Regression for Expectation Maximization Attack

To mitigate overfitting under the linear leakage model assumption we introduce a variation of the U-EM-LIN algorithm by assigning a certain probability distribution function to the dispersion of the parameters at the M-Step. For instance we can assume that $a - \bar{a}$ is distributed according to some $\mathcal{N}(\mathbf{0}, \sigma_a^2 I_n)$. Then when we maximize the log-likelihood we have to add the term $\log(\frac{1}{\sqrt{2\pi\sigma_a^2}}\exp(-\frac{\|a-\bar{a}\|^2}{2\sigma_a^2}))$. The new quantity to minimize at M-Step is now

$$\frac{1}{Q}\sum_q\sum_{m_q}\alpha_q^{(n)}(m_q)\|y_q - x(a,b,k,t_q,m_q)\|^2 + \frac{1}{2\sigma_a^2}\|a-\bar{a}\|^2. \qquad (22)$$

This yields in the so called *ridge regression* (a.k.a. *Tikhonov regularization*) with a coefficient $\lambda = \frac{1}{2\sigma_a^2}$. In our experiment we typically used $\lambda = 20$. This approach of regularization is well known in statistical learning and has already been introduced for side-channel analysis with on-the-fly regression in [WYS+18]. To optimize we first found a scalar a as for the Hamming model. Then we compute the linear term assuming that

Algorithm 1: Pseudo-code: U-EM-HAM.

```
Data: The traces $\mathbf{y} = (y_1, \dots, y_Q)$ and the noise standard deviation
      $\sigma = (\sigma^{(1)}, \sigma^{(2)})$
Input: Convergence threshold $\epsilon$
Output: Estimated key $\hat{k}$
$\bar{\mathbf{y}} \leftarrow \frac{1}{Q} \sum_{q=1}^{Q} y_q$ ;                    // Precompute the mean of the traces
$y^{(i)} \leftarrow \frac{y^{(i)} - \mathbf{y}^{\bar{(\mathbf{i})}}}{\sigma^{(i)}}$ for $i = 1, 2$ ;          // Center and normalise the traces by $\sigma$
forall key hypothesis $k \in \mathbb{F}_2^n$ do
    /* Initializations of the parameters $a$ and $b$                          */
    $a, b \leftarrow (1,1), (0,0)$ ;                      // Arbitrary, but could be chosen
    while True do
        /* E-Step                                                              */
        forall q,m compute do
            $x_{k,t_q,m} \leftarrow (w_H(S(k \oplus t_q) \oplus m), w_H(m))$
        forall q do
            $c_q \leftarrow \max_m -\frac{1}{2}\|y_q - \langle a, x_{k,t_q,m} \rangle - b\|^2$
        forall q,m do
            $\beta_q(m) \leftarrow p(m) \exp(c_q - \frac{1}{2}\|y_q - a x_{k,t_q,m} - b\|^2)$
        forall q compute do
            forall m compute do
                $\alpha_q(m) \leftarrow \frac{\beta_q(m)}{\sum \beta_q(m')}$
            $\widetilde{x}_q \leftarrow \sum_m \alpha_q(m) x_{k,t_q,m}$
        $\bar{\mathbf{x}} \leftarrow \frac{1}{Q} \sum_{q=1}^{Q} \widetilde{x}_q$
        for i =1,2 do
            $\widehat{\mathsf{Var}_{\mathbf{x}}^{(i)}} \leftarrow \sum_{q,m} \alpha_q(m)(x_{k,t_q,m}^{(i)} - \bar{\mathbf{x}}^{(i)})^2, \quad \widehat{\mathsf{Cov}_{\mathbf{xy}}^{(i)}} \leftarrow \sum_q \alpha_q(m)(\widetilde{x}_q^{(i)} - \bar{\mathbf{x}}^{(i)}) y_q^{(i)}$
            /* M-Step                                                              */
            $a'^{(i)} \leftarrow \widehat{\mathsf{Cov}_{\mathbf{xy}}^{(i)}} / \widehat{\mathsf{Var}_{\mathbf{x}}^{(i)}}, \quad b'^{(i)} \leftarrow -a'^{(i)} \bar{\mathbf{x}}^{(i)}$
        if $(\|a - a'\|^2 + \|b - b'\|^2) < \epsilon$ then
            Break ;                                                // Exit condition
        $a, b \leftarrow a', b'$
    $LogLikelihood(k) \leftarrow \sum_{q=1}^{Q} \log(\sum \beta_q) + c_q$
Result: $\hat{k} = \arg\max_k LogLikelihood(k)$
```

$\bar{a} = 0$. By doing the optimization this way the linear term computation is a strongly convex problem.

As we are in an unsupervised context we cannot optimize the hyper-parameter λ over a cross-validation set (contrary to [WYS+18]). Hence we suggest to use $\lambda \approx \frac{1}{2\sigma_a^2}$ where σ_a^2 is the variance of the epistemic noise (provided you have a guess on this value). We refer to this variation as U-EM-λ where λ is replaced by the value of λ used in the algorithm.

Algorithm 2: Pseudo-code: U-EM-LIN.

```
Data: The traces $\mathbf{y} = (y_1, \ldots, y_Q)$ and the noise standard deviation
      $\sigma = (\sigma^{(1)}, \sigma^{(2)})$
Input: Convergence threshold $\epsilon$
Output: Estimated key $\hat{k}$
$\bar{\mathbf{y}} \leftarrow \frac{1}{Q} \sum_{q=1}^{Q} y_q$ ;                    // Precompute the mean of the traces
$y^{(i)} \leftarrow \frac{y^{(i)} - \bar{\mathbf{y}}^{(\mathbf{i})}}{\sigma^{(i)}}$ for $i = 1, 2$ ;      // Center and normalise the traces by $\sigma$
forall key hypothesis $k \in \mathbb{F}_2^n$ do
    /* Initializations of the parameters $a$ and $b$                          */
    $a, b \leftarrow ((1, \ldots, 1), (1, \ldots, 1)), (0, 0)$ ;   // Arbitrary, but could be chosen
    while True do
        /* E-Step                                                           */
        forall $q,m$ compute do
            $x_{k,t_q,m} \leftarrow (S(k \oplus t_q) \oplus m, m)$
        forall $q$ do
            $c_q \leftarrow \max_m -\frac{1}{2}\|y_q - \langle a, x_{k,t_q,m}\rangle - b\|^2$
        forall $q,m$ do
            $\beta_q(m) \leftarrow p(m) \exp(c_q - \frac{1}{2}\|y_q - \langle a, x_{k,t_q,m}\rangle - b\|^2)$
        forall $q$ compute do
            forall $m$ compute do
                $\alpha_q(m) \leftarrow \frac{\beta_q(m)}{\sum \beta_q(m')}$
            $\widetilde{x}_q \leftarrow \sum_m \alpha_q(m) x_{k,t_q,m}$
        $\bar{\mathbf{x}} \leftarrow \frac{1}{Q} \sum_{q=1}^{Q} \widetilde{x}_q$
        for $i = 1,2$ do
            $\widehat{R^{(i)}_{\mathbf{xx}}} \leftarrow \sum_{q,m} \alpha_q(m)(x^{(i)}_{k,t_q,m} - \bar{\mathbf{x}}^{(i)})(x^{(i)}_{k,t_q,m} - \bar{\mathbf{x}}^{(i)})^\top \in \mathbb{R}^{n \times n}$
            $\widehat{R^{(i)}_{\mathbf{xy}}} \leftarrow \sum_{q} \alpha_q(m)(\widetilde{x}^{(i)}_q - \bar{\mathbf{x}}^{(i)}) {y^{(i)}_q}^\top \in \mathbb{R}^n$
            /* M-Step                                                       */
            $a'^{(i)} \leftarrow \widehat{R^{(i)}_{\mathbf{xx}}}^{-1} \widehat{R^{(i)}_{\mathbf{xy}}}$ ,    $b'^{(i)} \leftarrow -\langle a'^{(i)}, \bar{\mathbf{x}}^{(i)}\rangle$
        if $(\|a - a'\|^2 + \|b - b'\|^2) < \epsilon$ then
            Break ;                                          // Exit condition
        $a, b \leftarrow a', b'$
    $LogLikelihood(k) \leftarrow \sum_{q=1}^{Q} \log(\sum \beta_q) + c_q$
Result: $\hat{k} = \arg\max_k LogLikelihood(k)$
```

4.4 Hybrid Expectation Maximization Attack

Another perspective to handle the trade-off between the model complexity and overfitting is to to run *hybrid attacks*. For instance, one can run the U-EM with linear leakage model on the first components and Hamming weight leakage model on the second components. This may be especially interesting when the epistemic noise is uneven on the different shares. We refer to this variation as U-EM-HYB.

Table 2 summarizes the different attacks evaluated in the sequel.

Table 2: Distinguishers compared in this paper.

Name	Reference	Descriptions
2O-CPA	Eq. (13)	State-of-the-art non-profiled attack, as a reference
ML-HAM	Eq. (14)	Maximum likelihood distinguisher (a.k.a. template attack), assuming Hamming weight leakage model (2), which has been profiled beforehand. This distinguisher provided an upper bound (recall it requires a profiling)
ML-LIN	Eq. (14)	Same as ML-HAM, assuming linear leakage model (3)
P-EM	[LRP07]	Profiled EM with linear leakage model
U-EM-HAM	Alg. 1	Our new U-EM attack, assuming Hamming weight leakage model (2), which is profiled online by the EM algorithm
U-EM-LIN	Alg. 2	Same as U-EM-HAM, assuming linear leakage model (3)
U-EM-HYB	Sec. 4.4	A mix between U-EM-HAM and U-EM-LIN where we assume different leakage model on the two components
U-EM-λ	Eq. (22)	The derivation of U-EM with ridge regression at the M-Step with hyperparameter λ
U-EM-QUAD-λ	Eq. (22)	The derivation of U-EM-QUAD with ridge regression at the M-Step with hyperparameter λ

5 Experimental Results on Synthetic Traces

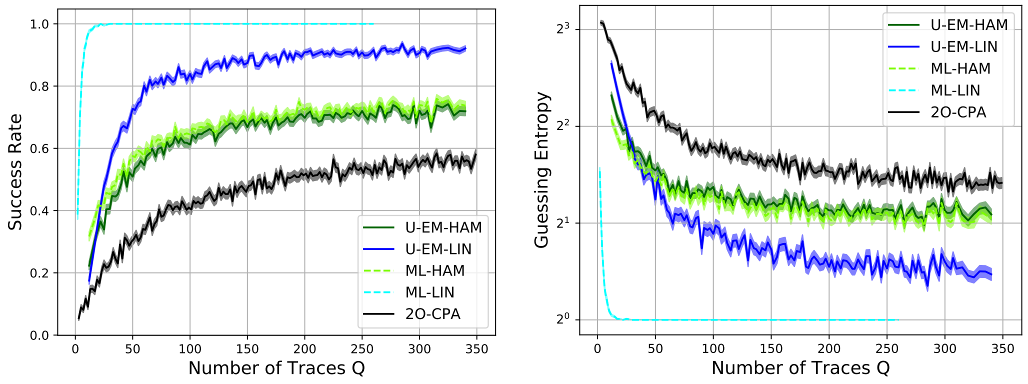
5.1 Experimental Setup and Comparison Metrics

We generate synthetic traces using linear leakage model, described earlier in (3). We set different values of the noise variance σ^2. We use the same σ^2 for the two shares. We worked on 4 bits with the PRESENT substitution box. For each experiment the unknown coefficient a^* are drawn randomly as $\mathcal{N}(1, \sigma_a^2)$. As σ_a gets closer to zero the model tends to the Hamming leakage model. When σ_a gets bigger so does the model mismatch with the Hamming weight leakage model. Then they are normalized to $\|(1,1,1,1)\| = 2$ which is the norm for the Hamming weight leakage. Hence for a given σ the signal to noise ratio verifies $\mathrm{SNR} = \frac{1}{\sigma^2} \sum\limits_{i=1}^{n} a_i^{*2} \frac{1}{2}(1 - \frac{1}{2}) = \frac{\|a^*\|^2}{4\sigma^2} = \frac{1}{\sigma^2}$. In particular it is independent of the drawn a^*.

We compare the results with two metrics, namely *success rate* and *guessing entropy*. The success rate is the empirical probability that the attack succeeds and the guessing entropy is the average indexed position of the secret key in the key ranking provided by the algorithms. Both metrics are estimated based on $1,000$ independent attacks and we shall superimpose the estimation error on the curves with the estimated standard deviation.

5.2 Attack Results for Several Epistemic and Measurement Noises

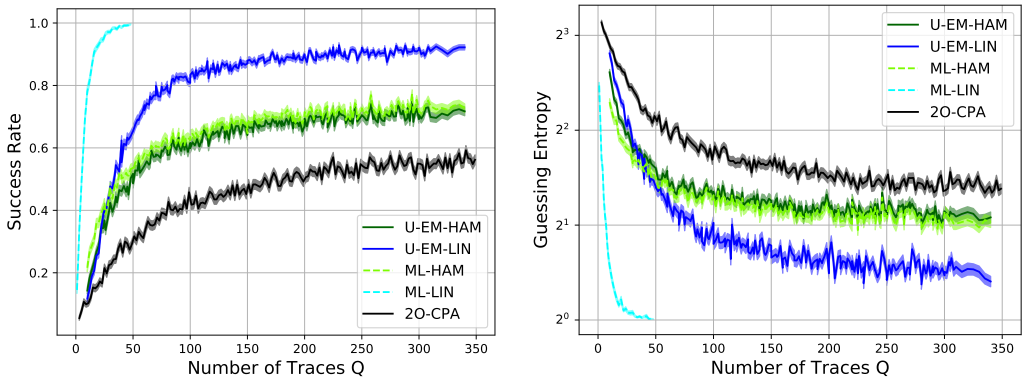
Attack results for $\sigma_a = 0.8$ are presented in Fig. 1 and 2 for $\sigma = 0.2$ (resp. $\sigma = 0.3$). Results for $\sigma_a = 0.4$ and $\sigma = 0.3$ are presented in Fig. 3. We recall that U-EM-LIN denotes the U-EM we have introduced in the article and U-EM-HAM is the U-EM algorithm applied to traces as if they were leaking according to the Hamming leakage model (model mismatch). These two algorithms only differ by their maximization step, the second requires less computations though.



(a) Success Rate.

(b) Guessing Entropy.

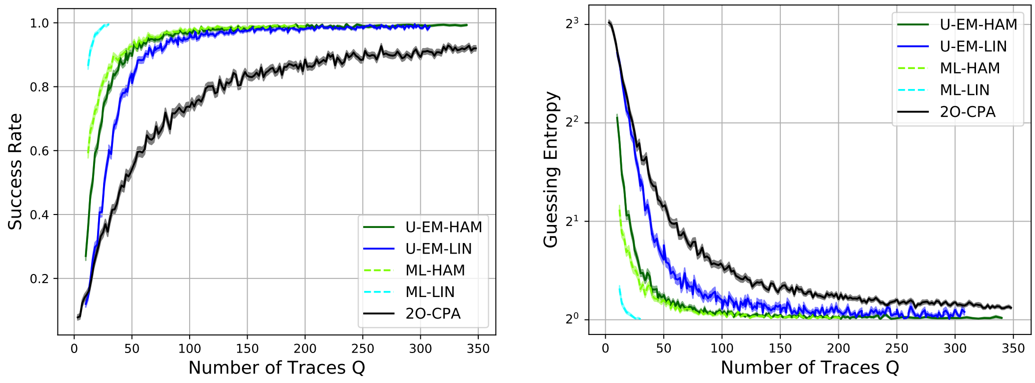
Figure 1: Attack results for $\sigma = 0.2$ and $\sigma_a = 0.8$.



(a) Success Rate.

(b) Guessing Entropy.

Figure 2: Attack results for $\sigma = 0.3$ and $\sigma_a = 0.8$.



(a) Success Rate.

(b) Guessing Entropy.

Figure 3: Attack results for $\sigma = 0.3$ and $\sigma_a = 0.4$.

As expected, both U-EMs outperform the 2O-CPA in terms of key extraction success rate. Additionally, the success rates are upper bounded by the template attacks.

The guessing entropy yields similar behavior. This confirms that our U-EM distinguisher improves compared to state-of-the-art 2O-CPA, and features distinguishing capability close to template attacks, even though it is a non-profiled attack.

Comparing U-EM-LIN with U-EM-HAM, we notice that the best algorithm depends on the values of σ and σ_a. As expected, when the epistemic noise σ_a is close to zero (here 0.4) then U-EM-HAM is providing better results than U-EM-LIN. Indeed, the extra-parameters of U-EM-LIN lead to an "overfitting effect". In contrast, when σ_a gets larger (here 0.8) U-EM-LIN outperforms U-EM-HAM when provided enough traces. Indeed we notice that for small amount of traces (typically $q < 30$ for $\sigma = 0.2$ or $q < 20$ for $\sigma = 0.3$), U-EM-HAM performs slightly better than U-EM-LIN. In conclusion we suggest to consider the relative importance of epistemic noise and observation noise to select the most appropriated model.

In general, U-EM algorithms work given a convergence threshold ϵ. We choose $\epsilon = 10^{-8}$ to ensure that the performance of the algorithms in terms of success rate and guessing entropy are optimal, while also maintaining computational complexity at a reasonable level.

We measured the CPU time per traces for the five algorithms on an Intel(R) Xeon(R) Gold 6128 server, running at 3.40 GHz. Results are provided in Table 3. It appears that, despite our new U-EM attacks require a bit more computations, their running time remains comparable to 2O-CPA. In conclusion, the U-EM attacks are computationally effective.

Table 3: CPU time per trace for the different distinguishers envisioned in this paper.

Name	**CPU time per traces (seconds)**
2O-CPA	3.2×10^{-4}
ML-HAM	3.2×10^{-4}
ML-LIN	5.1×10^{-4}
U-EM-HAM	1.1×10^{-3}
U-EM-LIN	1.9×10^{-3}

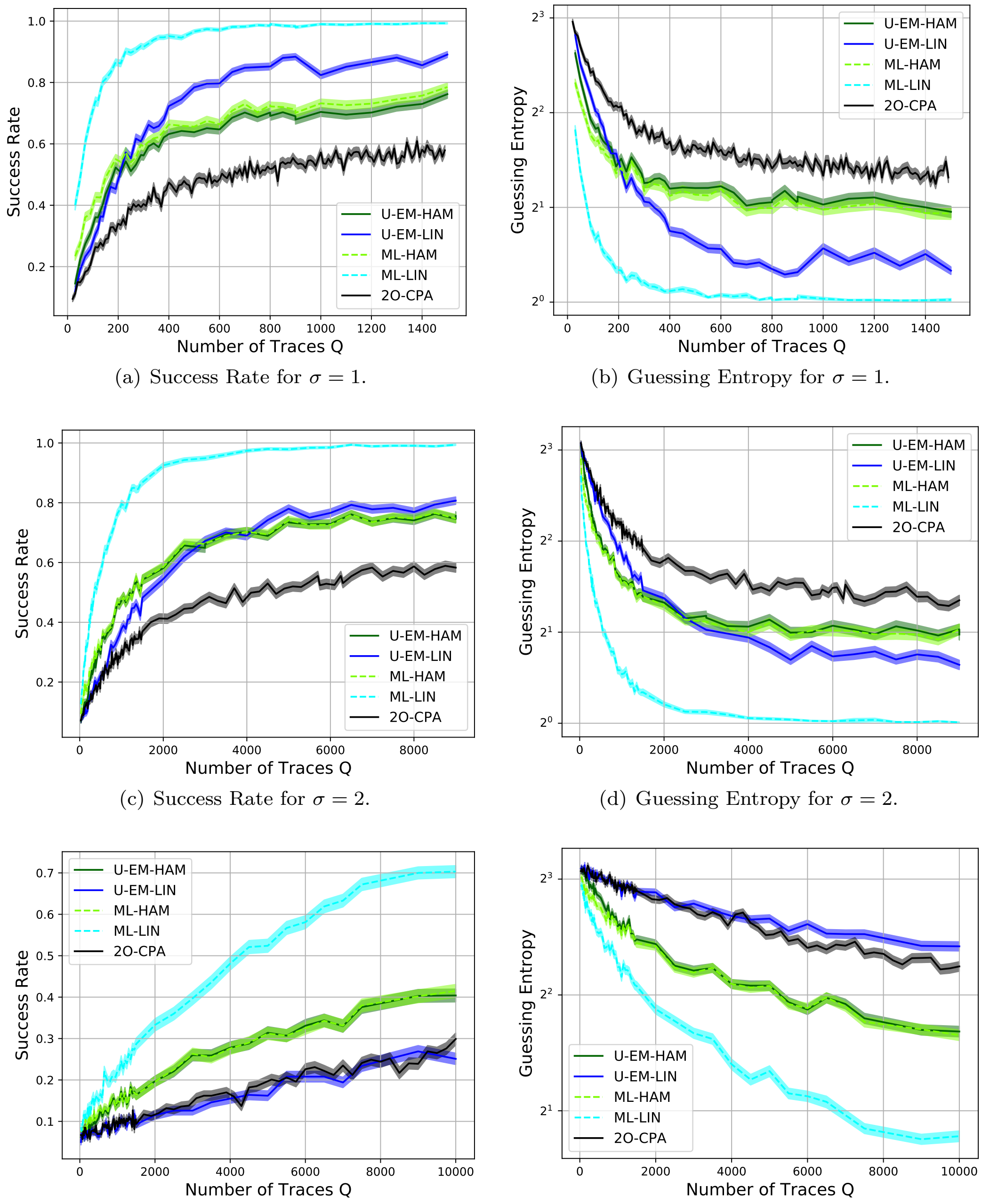
5.3 Performance of Attacks in the Presence of Large Noise

The simulation-based results are shown in Fig. 4 and Fig. 5, when noise incrementally increases, for $\sigma = \{1, 2, 4\}$, for a given epistemic noise level equal to $\sigma_\alpha = 0.8$ and 0.4 (already shown in Fig. 2 and 3 for a smaller noise $\sigma = 0.3$).

It is shown in [SVO+10] that under strong noise with Hamming weight leakages the performances of 2O-CPA and templates gradually gets closer. It is also pointed out that multivariate distinguishers are affected differently by noise (especially when absolute difference is used as a combination function). We can indeed observe that our U-EM-LIN attack reduces to 2O-CPA under high measurement noise σ when the epistemic noise is relatively small ($\sigma_a = 0.4$) compared to σ. Actually, Fig. 5 reveals that the performance of all considered attacks tends to become comparable as the measurement noise σ increases. Yet for higher epistemic noise ($\sigma_a = 0.8$) even though the performances are closer the reduction is not obvious. In the presence of large noise, U-EM-HAM performances are getting closer to the ML-HAM and ML-LIN whereas U-EM-LIN requires more traces. In such a scenario we prescribe U-EM-HAM. Intuitively, the reason is that under strong noise, the Gaussian mixtures are covered by noise.

5.4 Quadratic Leakage Model

We launch simulations with the non-linear leakage model (quadratic model of Eqn. (4)) and the experimental results are plotted in Fig. 6. As for the U-EM-LIN, the U-EM-QUAD suffers even more from its model complexity so it has to be used with ridge regression [WYS+18]. We tried parameters for the ridge coefficient $\lambda \in \{1, 2\}$. We confirmed the



(a) Success Rate for $\sigma = 1$.

(b) Guessing Entropy for $\sigma = 1$.

(c) Success Rate for $\sigma = 2$.

(d) Guessing Entropy for $\sigma = 2$.

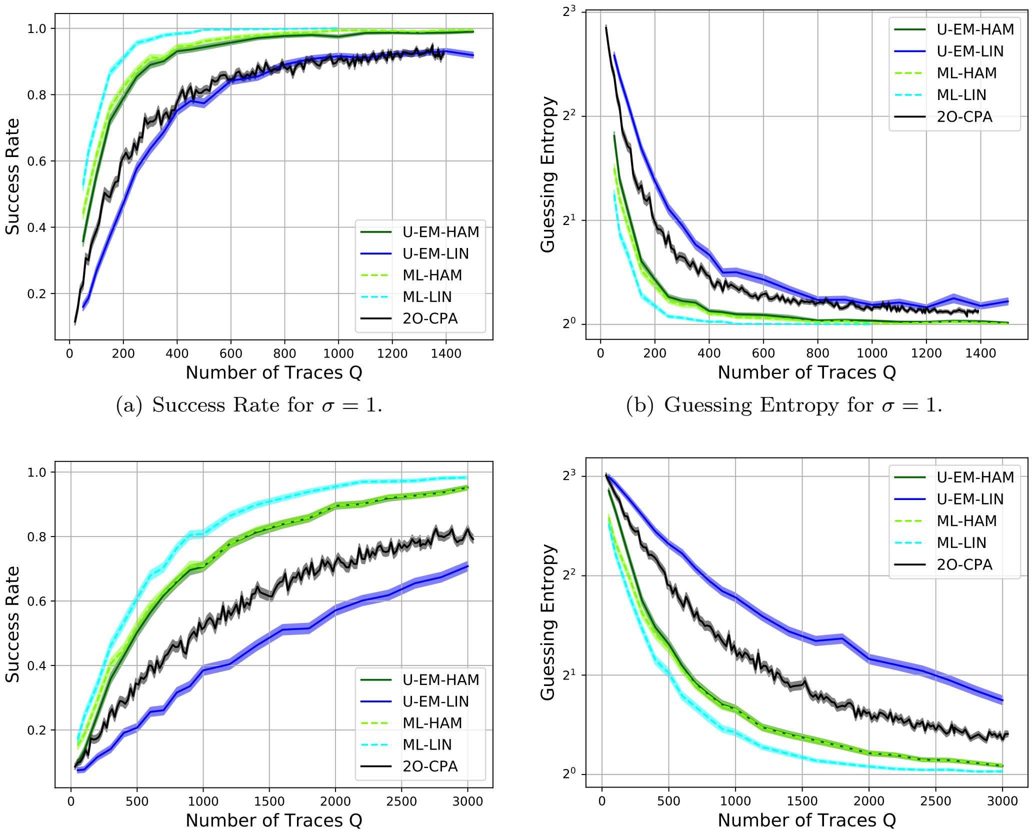
(e) Success Rate for $\sigma = 4$.

(f) Guessing Entropy for $\sigma = 4$.

Figure 4: Attack results for $\sigma_a = 0.8$ but different $\sigma \in \{1, 2, 4\}$.

observation of [WYS+18] that if the ridge parameter is too large for unprofiled distinguishers then the model looses its flexibility. The U-EM-QUAD-1 works well but is outperformed by U-EM-HAM most of the time.

As a perspective, it would be interesting to find if tuning the regularization parameter or adding more points of interest with the same leaking coefficients would improve the performance of this attack. However, from our empirical results, it seems preferable to use U-EM with a limited number of parameters.



(a) Success Rate for $\sigma = 1$.

(b) Guessing Entropy for $\sigma = 1$.

(c) Success Rate for $\sigma = 2$.

(d) Guessing Entropy for $\sigma = 2$.

Figure 5: Attack results for $\sigma_a = 0.4$ but different $\sigma \in \{1, 2\}$.

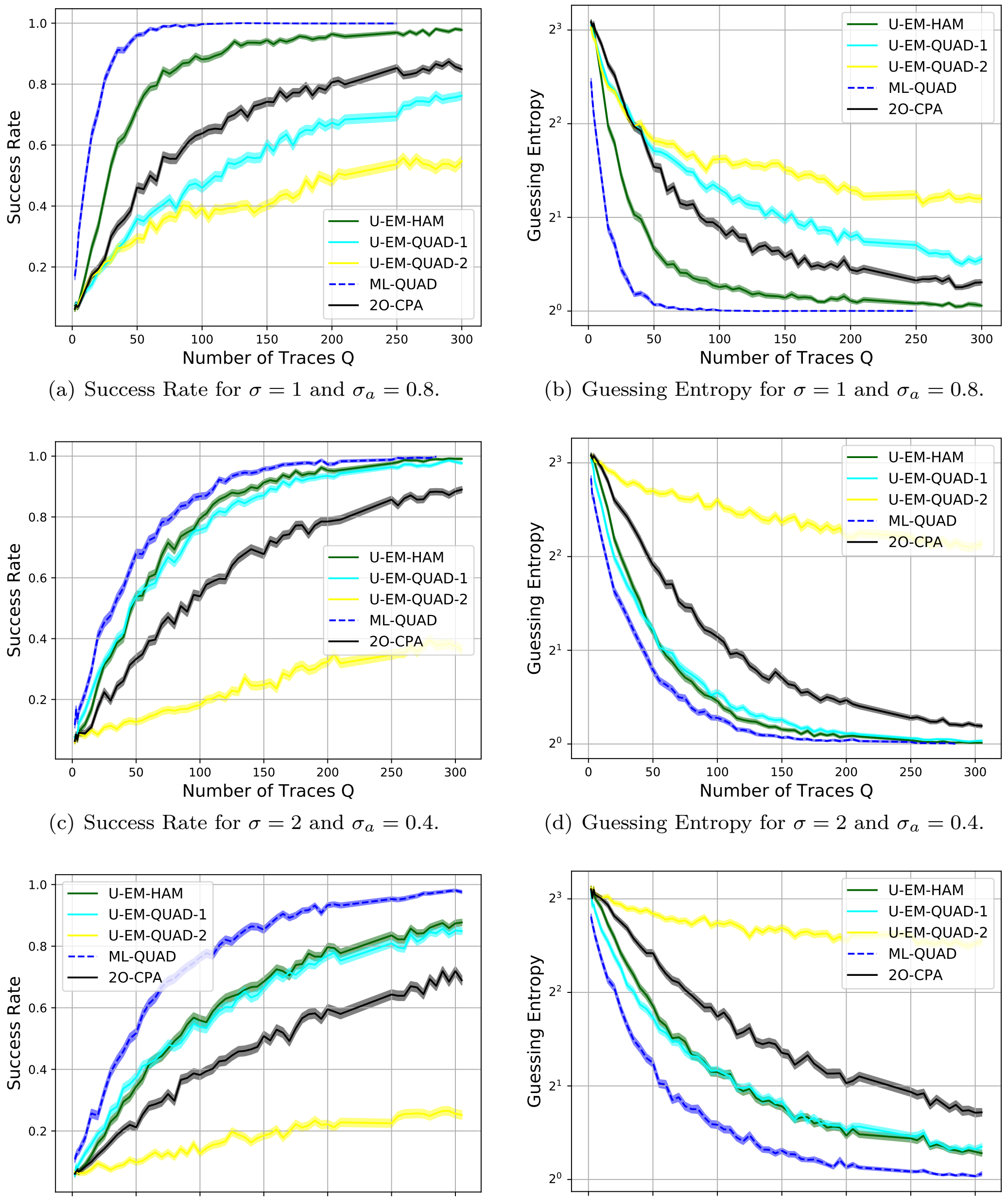
5.5 Comparison with Profiled EM

In this section we compare the performances of the P-EM and the U-EM proposed in this article, in different scenarii. Namely, various levels of noise (measurement and epistemic) are investigated. For the P-EM each key hypothesis is provided $Q_p = 50$, 150, or 300 traces for the profiling. Hence, $2^n Q_p$ traces are necessary in the profiling phase of P-EM.

5.5.1 Attack on the Same Device Used for Profiling

We are first interested in the scenario of attacking the very same device that has been profiled beforehand. This can happen when devices are profiled in early life cycle state, before they are deployed on the market. The attack goal is to recover the key which has been provisioned in the pre-market phase.

The attack results with respect to success rate and guessing entropy are shown in Fig. 7. As expected, the P-EM outperforms our U-EM, especially when few traces are available for the attack. In particular the P-EM does not suffer from the overfitting observed on our U-EM. Yet it should be kept in mind that the profiling phase of the P-EM requires many traces. If the laboratory evaluator has the time to measure $2^n Q_p$ traces, he/she could as well devote this time to directly run an U-EM, In that sense, the profiling stage of P-EM negatively impacts the performances of P-EM. In this case the U-EM could be a more efficient alternative. But if the laboratory evaluator has to evaluate the security of several devices leaking the same way, the profiling stage can be seen as an investment: the profile is indeed reused, making P-EM fast on each evaluated device.



(a) Success Rate for $\sigma = 1$ and $\sigma_a = 0.8$.

(b) Guessing Entropy for $\sigma = 1$ and $\sigma_a = 0.8$.

(c) Success Rate for $\sigma = 2$ and $\sigma_a = 0.4$.

(d) Guessing Entropy for $\sigma = 2$ and $\sigma_a = 0.4$.

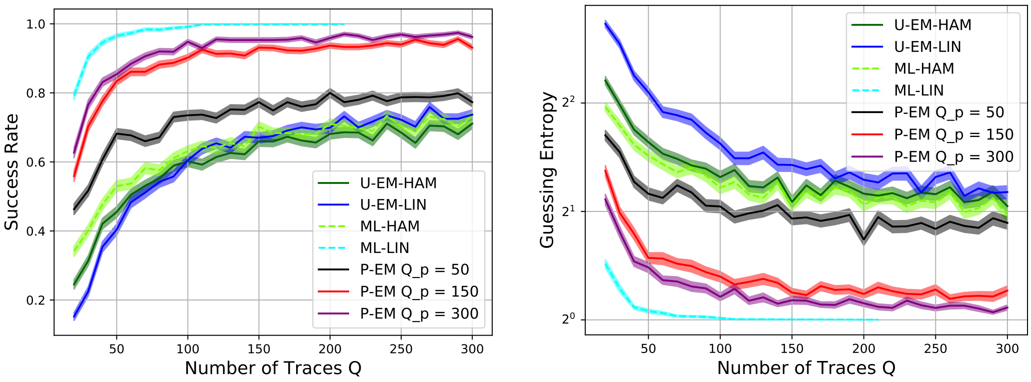
(e) Success Rate for $\sigma = 2$ and $\sigma_a = 0.8$.

(f) Guessing Entropy for $\sigma = 2$ and $\sigma_a = 0.8$.

Figure 6: Attack results for quadratic leakages.

5.5.2 Attack on Different Devices from the Profiling

In practice, however, the profiling device is usually not available, resulting in profiling mismatch between different devices. Subsequently, the leakage function differs from one device to another, and from one acquisition to another. In this scenario, unprofiled distinguishers will not be affected by this issue. In our simulations, we compare the attack performances when the profiling is done on an independent realization of the parameters drawn from the same epistemic noise distribution. Thus, the leakage model assumption is the same, but we first randomly draw coefficients to generate the actual leakages and



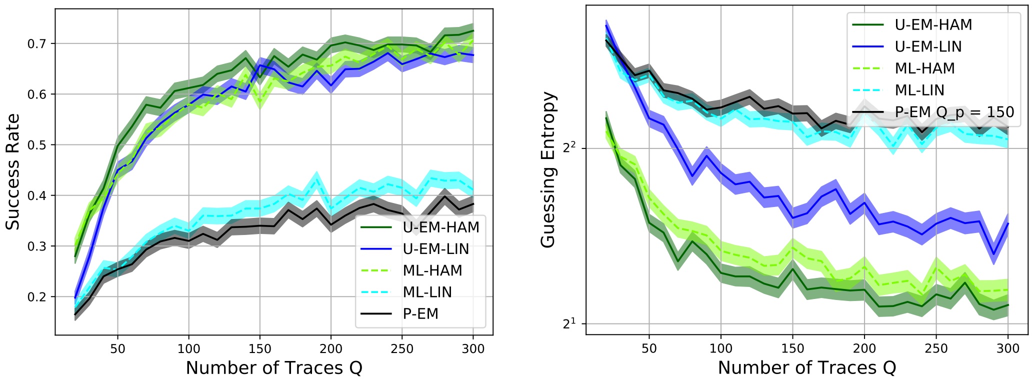
(a) Success Rate for $\sigma = 0.4$ and $\sigma_a = 0.4$.

(b) Guessing Entropy for $\sigma = 0.4$ and $\sigma_a = 0.4$.

Figure 7: Attack outcome when profiling is performed with exact same device (or perfect clone).

then draw another independent set of leakage coefficients for the templates. This amounts to profiling on another device with the same leakage model but which suffers from a mismatch.

The attack results are displayed in Fig. 8. As expected, we observe that the performance of ML-LIN and P-EM degrades drastically as a consequence of profiling mismatch. However, the poor performance of P-EM shall not be interpreted as a good level of security (recall [BL20]). The U-EM is not affected by this scenario, hence unambiguously outperforms P-EM.



(a) Success Rate for $\sigma = 0.3$ and $\sigma_a = 0.8$

(b) Guessing Entropy for $\sigma = 0.3$ and $\sigma_a = 0.8$

Figure 8: Attack outcome when profiling and attack are performed on two different devices.

6 Results on Real Traces from the DPAv2 Contest

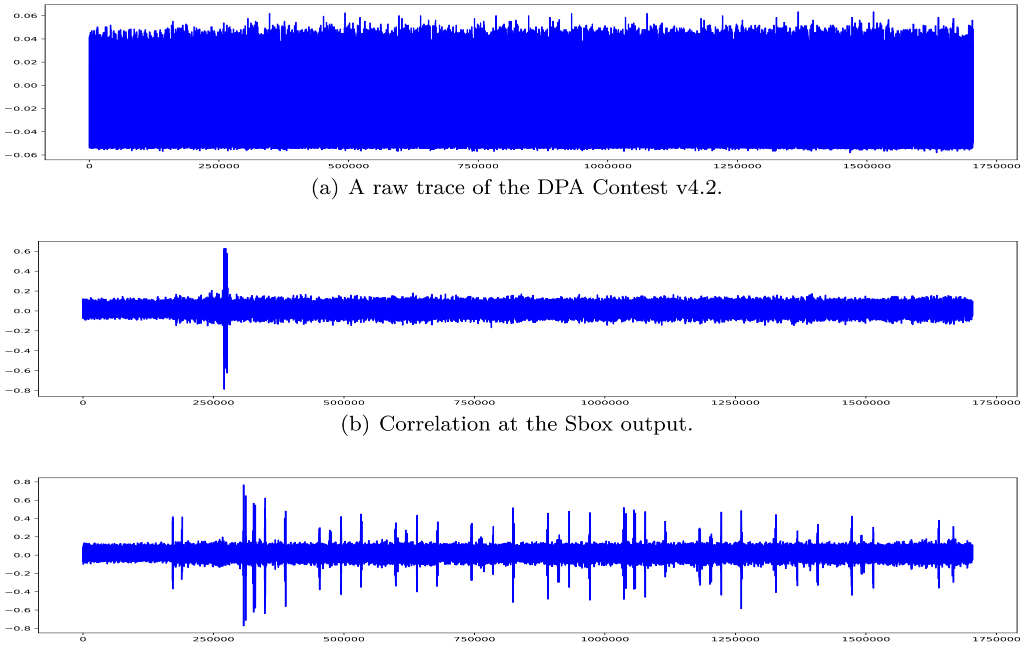
6.1 Experimental Setup and Comparative Metrics

We test the performance of the above distinguishers on the real traces provided in the DPA Contest v4.2. We did not run P-EM on DPA Contest though because it does not contain a dataset with all key hypothesis necessary for profiling the P-EM. These traces are measured on an *Atmel ATMega-163 smart card* with masked AES implementations.

Each byte is masked with one mask drawn uniformly out of 16 possible masks. Using the mask, plaintext and key we computed the point that is the most correlated to the model as shown in Alg. 3. The obtained results are provided in Fig. 9. The maximum correlation is around 0.8 (resp. 0.75) at the output of the S-Box (resp. mask). The noise's standard deviation for the two shares is $\sigma \approx (2.1 \times 10^{-3}, 3.9 \times 10^{-3})$. Note that we computed the PoIs this way by convenience (to save time in this "leakage detection" phase). However our attack is designed for an unsupervised context in which PoIs cannot be determined by traces correlation analysis. Please note that this problem is common to all unsupervised distinguishers and has already been observed for instance in [LRP07], and is not the object of this article. In an end-to-end key extraction scenario, the attacker should analyse the cryptographic source code to pinpoint instructions to target; alternatively, the attack could be performed pairwise for different PoIs locations assumptions. As pointed out in [LRP07], the number of possibilities can be reduced with dimensionality reduction techniques such as *principal components analysis* (PCA) which groups together samples with similar leakage structure.

Algorithm 3: Pseudo-code: Selection of the Points of Interest (PoIs)

```
Data: The $Q$ plaintexts $\mathbf{t}$ ,the $Q$ masks used $\mathbf{m}$, the $Q$ full traces $\mathbf{y} \in \mathbb{R}^{Q \times N}$ with
      $N$ values per trace and the secret key $k^*$.
Output: The two points of interest $(t_0, t_1)$.
$\mathbf{k}^* \leftarrow (k^*, \ldots, k^*)$
$S \leftarrow w_H(Sbox(\mathbf{k}^* \oplus \mathbf{t}) \oplus \mathbf{m}) \in \mathbb{R}^Q$
$M \leftarrow w_H(\mathbf{m}) \in \mathbb{R}^Q$
$t_0 \leftarrow \arg\max_{p \in \{1,\ldots,N\}} \widehat{\rho}(S, \mathbf{y}_p)$
$t_1 \leftarrow \arg\max_{p \in \{1,\ldots,N\}} \widehat{\rho}(M, \mathbf{y}_p)$
Result: $(t_0, t_1)$
```



(a) A raw trace of the DPA Contest v4.2.

(b) Correlation at the Sbox output.

(c) Correlation with the mask.

Figure 9: Selection of the point of interest on the DPA Contest v4.2.

In order to compute the upper bounds on the template attacks, we had to extract the leakage model from experimental traces. We place ourselves into the best case for the attacker, in that the model is extracted from the very attacked traces (hence no mismatch). Linear regression is used to extract the coefficients and the equivalent σ_a, which are provided in Table 4.

Table 4: Linear Regression results.

key	$a \times 10^3$	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	σ_a
k00	SBox	-1.9	-1.9	-2.1	-2.0	-1.7	-1.9	-1.9	-1.4	8.4×10^{-2}
	Mask	-3.6	-2.1	-6.5	-1.8	-6.7	-1.7	-1.3	-1.2	5.6×10^{-1}
k01	Sbox	-2.0	-2.0	-2.2	-2.0	-1.7	-1.9	-1.9	-1.3	1.0×10^{-1}
	Mask	-4.9	-3.0	-4.4	-1.5	-3.1	-1.6	-1.0	-1.5	4.7×10^{-1}
k02	Sbox	-1.9	-2.0	-2.2	-1.8	-1.7	-2.0	-1.9	-1.2	1.0×10^{-1}
	Mask	-4.8	-3.0	-5.0	-1.5	-3.5	-1.7	-1.2	-1.2	4.7×10^{-1}
k03	SBox	-1.8	-1.9	-2.1	-1.9	-1.7	-1.9	-1.6	-1.3	1.0×10^{-1}
	Mask	-4.2	-2.5	-6.3	-1.4	-3.7	-1.2	-1.0	-1.1	5.5×10^{-1}
k04	SBox	-1.9	-1.9	-1.9	-1.9	-1.7	-1.9	-1.6	-1.3	9.0×10^{-2}
	Mask	-3.9	-2.6	-6.4	-1.4	-3.9	-1.3	-0.8	-1.3	5.5×10^{-1}

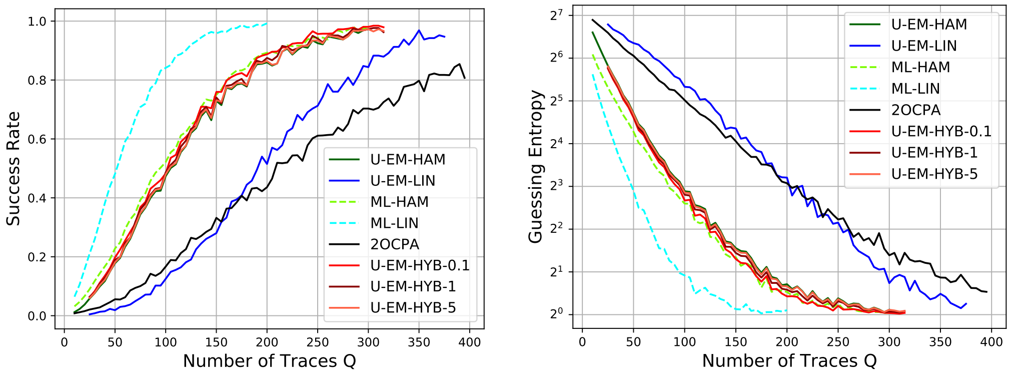
Coefficients provided in Table 4 suggest that the Hamming weight leakage model is underfitting the leakages. Indeed, the 8 coefficients characterizing the leakage from the mask bits strongly differ. Hence we expect the linear model to outperform both the Hamming weight leakage model and the 2O-CPA (computed with Hamming weight model). Though this should be shaded since the relative importance of σ_a and σ is uneven for the mask and the S-Box.

As for comparative metrics, the same metrics as for the synthetic traces are examined. The dataset size consist in 5000 traces per key folder, with 16 different keys to attack. Hence their are 80000 traces. We used all the traces to compute the metrics, which means that they are estimated with $80000/q$ traces, for a given q (represented in the abscissa). Therefore, the estimate is more crude when q is larger. Notice that, since the attacked algorithm is AES (8-bit cipher), the guessing entropy ranges from 1 to 256, whereas in the previous section on synthetic traces, the guessing entropy ranged from 1 to 16.

6.2 Attack Results

The maximum likelihood attacks have been run with the coefficients obtained from the linear regression for both the Hamming weight leakage model (ML-HAM) and the linear leakage model (ML-LIN). The computational complexity of U-EM-LIN being more important it is computed for fewer q than the other curves. Results in terms success rate (resp. guessing entropy) are shown in Fig. 10(a) (resp. Fig. 10(b)).

These experimental results confirm that our U-EM algorithm is better than 2O-CPA. As expected the ML-LIN outperforms ML-HAM and ML-HAM outperforms U-EM-HAM. The performance of the U-EM-HAM is very close to that upper-bound. In that we confirm the observation of [BDMS22] that unprofiled attacks achieve similar performances as their profiled counterparts when the noise is high enough. This only imply a computation overhead. The epistemic noise seems too small to make U-EM-LIN interesting here. Hence we applied our different strategies to improve the U-EM attack on these real traces. We observe that the U-EM-HYB with ridge regression performs the best on the traces. It is even slightly better than the ML-HAM for $q \geq 200$. We notice that the performance does not depend sharply on the value of the selected λ. So even though we cannot optimize the



(a) Success rate.

(b) Guessing entropy.

Figure 10: Attack metrics on the traces of the DPA Contest v4.2.

performance of the attack over the hyperparameter λ the performance are robust to the choice of reasonable values of λ.

7 Discussion

The U-EM requires more traces to reach a given success rate than its profiled counterpart (namely, P-EM). This is expected since less *a priori* information is available. At least, this holds true in the absence of model mismatch, namely when profiling is perfect, when the target is leaking strictly identically, when the setup is same, and when environmental conditions during traces acquisition are the same. On the contrary, in case of any deviation, our U-EM attack comes to the front. Indeed, it tolerates any variety of conditions, because our a priori information concerns the sensitive variables (digital) but we make no assumption on the leakage function (analog weights). In summary, we prescribe our U-EM attack to evaluators when:

- there is no way to procure a training device with known or chosen key,
- or when no such perfect "clone" device exists (owing to the advanced fabrication technology which features significant inter-chip variability [RSV$^+$11], or owing to device aging properties decay [KGD18], etc.),
- or when the leakage is hard to probe (because the device is so small that it is chancy to place the probe at a reproducible location upon either profiling or matching phases, or also because environmental conditions cannot be precisely controlled, etc.)

As observed in [BDMS22] attacks performed without knowledge of the masks offer similar performances as their profiled counterparts for noise large enough. This suggests that our U-EM attack may be advantageous to avoid a lengthy profiling phase.

Finally, we notice that the U-EM attack needs to be initialized (refer to line 4 in Alg. 1 and/or to line 4 in Alg. 2). When a template is profiled on a given device, our approach can also be used with ridge-regression [WYS$^+$18] centered on the considered profiled template to accommodate the inter-chip variability and the different measurement conditions (exact position of the probe, temperature, humidity, etc.). In this respect our U-EM would be complementary to template attacks with Gaussian mixtures. This would be a way to exploit better a priori knowledge about the leakage while accommodating to the inter-chip and experimental variations.

8 Conclusion and Perspectives

We have shown that the U-EM attack is the most relevant distinguisher known so far when profiling is not feasible. These attacks are indeed computationally efficient and outperform the state-of-the-art 2O-CPA both in terms of success rate and guessing entropy. Moreover, it enables to use more flexible models if necessary and it is very easy to increase the number of points of interest to integrate them in the U-EM attack. Provided that the attacker is aware of the the epistemic noise associated to the different model parameters, a *hybrid* attack mixing the different models can be easily devised.

When applied to real world traces, we noticed that the modeling with Hamming weight allows this U-EM attacks to be as effective as the maximum likelihood attack. If the model is refined to be an arbitrary linear combination of bits, we noticed that maximum likelihood attack improves considerably, but that U-EM attacks does not manage to improve accordingly. We attribute this deficiency to the overfitting of U-EM, when the number of parameters is so high that even incorrect key guesses are matched with a high likelihood.

As a perspective, we notice that the transposition to higher masking orders should in theory be straightforward since it consists in appending one more parameter in the regression M-step. It should be interesting to compare performance of such attacks with the different variants presented here.

Acknowledgments

The authors sincerely thank the anonymous reviewers for their valuable comments, which significantly improved the quality of the paper. This work has partly benefited from the bilateral MESRI-BMBF project “APRIORI” from the ANR cybersecurity 2020 call. It is also part of the Horizon 2020 “SPARTA” project under grant agreement number 830892. Besides, the authors acknowledge financial support of the French national Bank (BPI) under Securyzr-V grant (Contract n° DOS0144216/00), a RISC-V centric platform integrating security co-processors.

References

[BCO04] Éric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.

[BDMS22] Olivier Bronchain, François Durvaux, Loïc Masure, and François-Xavier Standaert. Efficient profiled side-channel analysis of masked implementations, extended. *IEEE Transactions on Information Forensics and Security*, 17:574–584, 2022.

[BGHR14] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, and Olivier Rioul. Masks Will Fall Off – Higher-Order Optimal Distinguishers. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 344–365. Springer, 2014.

[BGNT18] Nicolas Bruneau, Sylvain Guilley, Zakaria Najm, and Yannick Teglia. Multivariate high-order attacks of shuffled tables recomputation. *J. Cryptol.*, 31(2):351–393, 2018.

[BHM$^+$19] Olivier Bronchain, Julien M. Hendrickx, Clément Massart, Alex Olshevsky, and François-Xavier Standaert. Leakage certification revisited: Bounding model errors in side-channel security evaluations. Cryptology ePrint Archive, Report 2019/132, 2019. https://ia.cr/2019/132.

[BL20] Rinat Breuer and Itamar Levi. How Bad Are Bad Templates? Optimistic Design-Stage Side-Channel Security Evaluation and its Cost. *Cryptogr.*, 4(4):36, 2020.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In Wiener [Wie99], pages 398–412.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski, Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.

[dCGHR18] Éloi de Chérisey, Sylvain Guilley, Annelie Heuser, and Olivier Rioul. On the optimality and practicability of mutual information analysis in some scenarios. *Cryptography and Communications*, 10(1):101–121, 2018.

[DLR77] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[DPRS11] Julien Doget, Emmanuel Prouff, Matthieu Rivain, and François-Xavier Standaert. Univariate side channel attacks and leakage modeling. *J. Cryptographic Engineering*, 1(2):123–144, 2011.

[GBPV10] Benedikt Gierlichs, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. Revisiting Higher-Order DPA Attacks: Multivariate Mutual Information Analysis. In *CT-RSA*, volume 5985 of *LNCS*, pages 221–234. Springer, March 1-5 2010. San Francisco, CA, USA.

[GBTP08] Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual information analysis. In *CHES, 10th International Workshop*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442. Springer, August 10-13 2008. Washington, D.C., USA.

[GLRP06] Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. Templates vs. Stochastic Methods. In *CHES*, volume 4249 of *LNCS*, pages 15–29. Springer, October 10-13 2006. Yokohama, Japan.

[HRG14a] Annelie Heuser, Olivier Rioul, and Sylvain Guilley. A Theoretical Study of Kolmogorov-Smirnov Distinguishers — Side-Channel Analysis vs. Differential Cryptanalysis. In Emmanuel Prouff, editor, *Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers*, volume 8622 of *Lecture Notes in Computer Science*, pages 9–28. Springer, 2014.

[HRG14b] Annelie Heuser, Olivier Rioul, and Sylvain Guilley. Good Is Not Good Enough - Deriving Optimal Distinguishers from Communication Theory. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 55–74. Springer, 2014.

[KGD18] Naghmeh Karimi, Sylvain Guilley, and Jean-Luc Danger. Impact of Aging on Template Attacks. In *Proceedings of the 28th ACM Great Lakes Symposium on VLSI*, GLSVLSI '18. ACM, May 23-25 2018. Chicago, Illinois, USA.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Wiener [Wie99], pages 388–397.

[LBM15] Liran Lerman, Gianluca Bontempi, and Olivier Markowitch. A machine learning approach against a masked AES - Reaching the limit of side-channel attacks with a learning model. *J. Cryptographic Engineering*, 5(2):123–139, 2015.

[LM18] Liran Lerman and Olivier Markowitch. Efficient profiled attacks on masking schemes. *IEEE Transactions on Information Forensics and Security*, 14(6):1445–1454, 2018.

[LRP07] Kerstin Lemke-Rust and Christof Paar. Gaussian mixture models for higher-order side channel analysis. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007*, pages 14–27, Berlin, Heidelberg, 2007. Springer.

[MCGD12] Houssem Maghrebi, Claude Carlet, Sylvain Guilley, and Jean-Luc Danger. Optimal first-order masking with linear and non-linear bijections. In Aikaterini Mitrokotsa and Serge Vaudenay, editors, *Progress in Cryptology - AFRICACRYPT 2012*, pages 360–377, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[MDP20] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A Comprehensive Study of Deep Learning for Side-Channel Analysis. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(1):348–375, 2020.

[Mes00] Thomas S. Messerges. Using Second-Order Power Analysis to Attack DPA Resistant Software. In *CHES*, volume 1965 of *LNCS*, pages 238–251. Springer-Verlag, August 17-18 2000. Worcester, MA, USA.

[OM07] Elisabeth Oswald and Stefan Mangard. Template Attacks on Masking — Resistance Is Futile. In Masayuki Abe, editor, *CT-RSA*, volume 4377 of *Lecture Notes in Computer Science*, pages 243–256. Springer, 2007.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against Side-Channel Attacks: A Formal Security Proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 142–159. Springer, 2013.

[PRB09] Emmanuel Prouff, Matthieu Rivain, and Régis Bevan. Statistical Analysis of Second Order Differential Power Analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009.

[RSV$^+$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, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 109–128. Springer, 2011.

[SKM$^+$13] Fabrizio De Santis, Michael Kasper, Stefan Mangard, Georg Sigl, Oliver Stein, and Marc Stöttinger. On the Relationship between Correlation Power Analysis and the Stochastic Approach: An ASIC Designer Perspective. In Goutam Paul and Serge Vaudenay, editors, *Progress in Cryptology - INDOCRYPT 2013 - 14th International Conference on Cryptology in India, Mumbai, India, December 7-10, 2013. Proceedings*, volume 8250 of *Lecture Notes in Computer Science*, pages 215–226. Springer, 2013.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In LNCS, editor, *CHES*, volume 3659 of *LNCS*, pages 30–46. Springer, Sept 2005. Edinburgh, Scotland, UK.

[SVO$^+$10] 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, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2010.

[Wie99] Michael J. Wiener, editor. *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*. Springer, 1999.

[WO13] Carolyn Whitnall and Elisabeth Oswald. Profiling DPA: efficacy and efficiency trade-offs. In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, volume 8086 of *Lecture Notes in Computer Science*, pages 37–54. Springer, 2013.

[WOM11] Carolyn Whitnall, Elisabeth Oswald, and Luke Mather. An Exploration of the Kolmogorov-Smirnov Test as a Competitor to Mutual Information Analysis. In Emmanuel Prouff, editor, *CARDIS*, volume 7079 of *Lecture Notes in Computer Science*, pages 234–251. Springer, 2011.

[WOS14] Carolyn Whitnall, Elisabeth Oswald, and François-Xavier Standaert. The Myth of Generic DPA…and the Magic of Learning. In Josh Benaloh, editor, *Topics in Cryptology – CT-RSA 2014*, pages 183–205, Cham, 2014. Springer International Publishing.

[WYS$^+$18] Weijia Wang, Yu Yu, François-Xavier Standaert, Junrong Liu, Zheng Guo, and Dawu Gu. Ridge-Based DPA: Improvement of Differential Power Analysis For Nanoscale Chips. *IEEE Trans. Inf. Forensics Secur.*, 13(5):1301–1316, 2018.