# Score-Based vs. Probability-Based Enumeration – A Cautionary Note

Marios O. Choudary[1(✉)], Romain Poussier[2], and François-Xavier Standaert[2]

[1] University Politehnica of Bucharest, Bucharest, Romania
marios.choudary@cs.pub.ro
[2] ICTEAM - Crypto Group, Université Catholique de Louvain, Louvain-la-Neuve, Belgium

**Abstract.** The fair evaluation of leaking devices generally requires to come with the best possible distinguishers to extract and exploit side-channel information. While the need of a sound model for the leakages is a well known issue, the risks of additional errors in the post-processing of the attack results (with key enumeration/key rank estimation) are less investigated. Namely, optimal post-processing is known to be possible with distinguishers outputting probabilities (e.g. template attacks), but the impact of a deviation from this context has not been quantified so far. We therefore provide a consolidating experimental analysis in this direction, based on simulated and actual measurements. Our main conclusions are twofold. We first show that the concrete impact of heuristic scores such as produced with a correlation power analysis can lead to non-negligible post-processing errors. We then show that such errors can be mitigated in practice, with Bayesian extensions or specialized distinguishers (e.g. on-the-fly linear regression).

## 1 Introduction

Side-channel attacks are powerful tools to recover the secret keys of cryptographic implementations. When an attacker has physical access to the device (e.g. a smart card) running the cryptographic algorithm, he may extract information using side-channels such as the power consumption or the electromagnetic radiation related to the algorithm being executed. In the following, we refer to this information as a trace or a leakage. Examples of attacks exploiting this type of leakages include (but are not limited to) Kocher et al.'s Differential Power Analysis (DPA) [11], Brier et al.'s Correlation Power Analysis (CPA) [3] or Chari et al.'s Template Attack (TA) [4]. Applied on symmetric block ciphers, they usually allow recovering information on independent parts of the full key which we shall further call subkeys. Typically, an attacker obtains a probability (or a score) for each of the possible value a subkey can take. In order to recover the full key, he has to recombine the information from the different subkeys. In this context we can distinguish three cases. Firstly, if the attacker has enough information on each subkey (i.e. the correct subkey is ranked first), he


O. Dunkelman and S.K. Sanadhya (Eds.): INDOCRYPT 2016, LNCS 10095, pp. 137–152, 2016.
DOI: 10.1007/978-3-319-49890-4_8

can recombine them by simply taking the most likely hypothesis for each subkey. Secondly, if at least one subkey does not provide enough information, a search into the key space needs to be performed. For this purpose, a key enumeration algorithm can be used in order to output the full key in decreasing order of likelihood [2,6,12,17–19,21]. The position of the actual key in this ordering will be denoted as the rank. Finally, when the rank is beyond the attacker's computational power (e.g. $>2^{50}$), a complete key enumeration is not possible anymore. In that case, one can only estimate a bound on the rank using a rank estimation algorithm, which requires knowledge of the actual key (thus is only suitable for an evaluator) [1,8,10,17,18,22,23].

In this paper, we deal with the general issue of evaluating the security of a leaking device in the most accurate manner. This is an important concern since an error might lead to an overestimation of the rank and thus a false sense of security. Sources of errors that can affect the quality of a side-channel security evaluation have been recently listed in [18] and are summarized in Fig. 1, where we can see the full process of a divide-and-conquer side-channel attack from the trace acquisition to the full key recovery, along with the errors that might be introduced. A typical (and well understood [9]) example is the case of model errors, which can significantly affect the efficiency of the subkey recoveries. Interestingly, the figure also suggests that some errors may only appear during the enumeration/rank estimation phase of the side-channel security evaluation. Namely, *combination ordering errors*, which are the core this study, arise from the fact that combining the information obtained for different subkeys can be easily done in a sound manner if the subkey distinguisher outputs probabilities (which is typically the case of optimal distinguishers such as TA), while performing this combination with heuristic scores (e.g. correlation values) is not obvious. This is an interesting observation since it goes against the equivalence

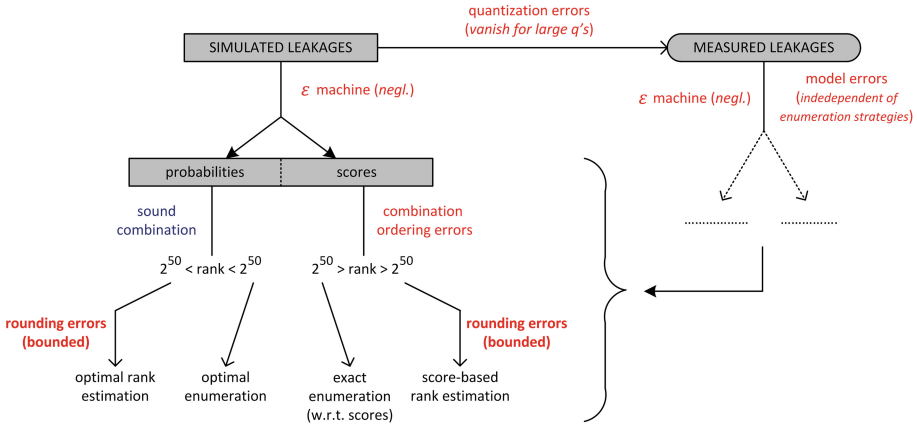


**Fig. 1.** Errors & bounds in key enumeration and rank estimation.

of TA and CPA (under the assumption that they use identical leakage models) that holds for first-order side-channel attacks [15].

In general, the soundest way to evaluate security is to use a Bayesian distinguisher (i.e. a TA) and to output probabilities. Yet, a possible concrete drawback of such an attack is that it requires a good enough model of the leakages Probability Density Function (PDF), i.e. some profiling. For this reason, non-profiled (and possibly sub-optimal) attacks such as the CPA are also frequently considered in the literature. As previously mentioned, distinguishers that output probabilities allow a straightforward combination of the subkey information. This is because probabilities are known to have a multiplicative relationship. By contrast, the combination of heuristic scores does not provide such an easy relationship, and therefore requires additional assumptions, which may lead to combination ordering errors. This is easily illustrated with the following example. Let us first assume two 1-bit subkeys $k_1$ and $k_2$ with probability lists $[p_1^1, p_1^2]$ and $[p_2^1, p_2^2]$ with $p_1^1 >, p_1^2$ and $p_2^1 > p_2^2$. Secondly, let us assume the same subkeys with score lists $[s_1^1, s_1^2]$ and $[s_2^1, s_2^2]$ with $s_1^1 >, s_1^2$ and $s_2^1 > s_2^2$. On the one hand, it is clear that the most (respectively least) probable key is given by $\{p_1^1, p_2^1\}$ or $\{s_1^1, s_2^1\}$ (respectively $\{p_1^2, p_2^2\}$ or $\{s_1^2, s_2^2\}$). On the other hand, only probabilities allow to compare the pair $\{p_1^1, p_2^2\}$ and $\{p_1^2, p_2^1\}$. Since we have no sound way to combine scores, we have no clue how to compare $\{s_1^1, s_2^2\}$ and $\{s_1^2, s_2^1\}$.

In the following, we therefore provide an experimental case study allowing us to discuss these combination ordering errors. For this purpose, we compare the results of side-channel attacks with enumeration in the case of TA, CPA (possibly including a bayesian extension [21]) and Linear Regression (LR) [20] and in particular its non-profiled variant put forward in [7], using both simulations and concrete measurements. Our main conclusions are that (i) combination ordering errors can have a significant impact when the leakage model of the different subkeys in an attack differ, (ii) bayesian extensions of the CPA can sometimes mitigate these drawbacks, yet include additional assumptions that may lead to other types of errors, (iii) "on-the-fly" linear regression is generally a good (and more systematic) way to avoid these errors too, but come at the cost of a model estimation phase, and (iv) only TA are guaranteed to lead to optimal results. The rest of the paper is structured as follows. In Sect. 2, we present the different attacks we analyse, together with a concise description of the key enumeration and rank estimations algorithms. Then, in Sect. 3, we present our experimental results. Conclusions are in Sect. 4.

## 2 Background

### 2.1 Attacks

This section describes the different attacks we used along with the tool for full key recovery analysis. The first one we consider is the correlation power analysis [3], along with its bayesian extension to produce probabilities. The second one is the template attack [4]. The last one is the "on-the-fly" stochastic attack [7]. We target a $b$-bit master key $k$, where an attacker recovers information on $N_s$ subkeys $k_0, \ldots, k_{N_s-1}$ of length $a = \frac{b}{N_s}$ bits (for simplicity, we assume that $a$

divides $b$). Our analysis targets the S-box computation ($\mathsf{S}$) of a cryptographic algorithm. That is, we apply a divide-and-conquer attack where a subkey $k$ is manipulated during a computation $\mathsf{S}(x \oplus k)$ for an input $x$. Given $n$ executions with inputs $\mathbf{x} = (x_i)$, $i \in [1, n]$ (where the bold notation is for vectors), we record (or simulate) the $n$ side-channel traces $\mathbf{l} = (l_i^{x,k})$ (e.g. power consumption), corresponding to the computation of the S-box output value $v_{x,k} = S(x \oplus k)$ with key (subkey) $k$. We then use these traces in our side-channel attacks.

**Correlation Power Analysis (CPA).** From a given leakage model $\mathbf{m}^\star = (m_{x,k^\star})$ (corresponding to a key hypothesis $k^\star$) and $n$ traces $\mathbf{l} = (l_i^{x,k})$, we compute Pearson's correlation coefficient $\rho_k^\star = \rho(\mathbf{l}, m_{x,k^\star})$ for all candidates $k^\star$ as shown by (1):

$$\rho_{k^\star} = \frac{\sum_{i=1}^{n}(l_i - \mathsf{E}(\mathbf{l})) \cdot (m_{x_i,k^\star} - \mathsf{E}(\mathbf{m}^\star))}{\mathsf{Std}(\mathbf{l}) \cdot \mathsf{Std}(\mathbf{m}^\star)}, \tag{1}$$

where $\mathsf{E}$ and $\mathsf{Std}$ denote the sample mean and standard deviation. If the attack is successful, the subkey $k = \arg\max_{k^\star}(\rho_{k^\star})$ is the correct one. This generally holds given a good model and a large enough number of traces. Concretely, our following experiments will always consider CPA with a Hamming weight leakage model, which is a frequent assumption in the literature [14], and was sufficient to obtain successful key recoveries in our case study.

**CPA with Bayesian Extension (BCPA).** In order to improve the results of key enumeration algorithms with CPA, we may use Fisher's Z-transform (2) on the correlation values $\rho_{k^\star}$ obtained from the CPA attack (see above) [13]:

$$z(\rho_{k^\star}) = \frac{1}{2} \log \frac{1 + \rho_{k^\star}}{1 - \rho_{k^\star}}. \tag{2}$$

Under some additional assumptions, this transformation can help us transform the vector of correlations into a vector of probabilities, which may be more suitable for the key enumeration algorithms, hence possibly leading to improved results.

More precisely, if we let $z_{k^\star} = z(\rho_{k^\star})$ be the z-transform of the correlation for the candidate key $k^\star$, then ideally we can exploit the fact that this value is normally distributed, and compute the probability:

$$\Pr[l|k^\star] = \mathcal{N}_{\mu_z,\sigma_z^2}(z_{k^\star}), \tag{3}$$

where

$$\mu_z = \frac{1}{2} \log \frac{1 + \rho}{1 - \rho}$$

is the z-transform of the real correlation $\rho$ and

$$\sigma_z^2 = \frac{1}{n - 3}$$

is the estimated variance for this distribution (with $n$ the number of attack samples). The main problem, however, is that we do not know the actual value of the correlation $\rho$ (for the correct and wrong key candidates), so we cannot determine $\mu_z$ and $\sigma_z^2$ without additional assumptions.

A possible solution for the mean $\mu_z$ is to assume that the incorrect keys will have a correlation close to zero (so the z-transform for these keys will also be close to zero), while the correct key will have a correlation as far as possible from zero. In this case, we can take the absolute value of the correlation values, $|\rho_{k^\star}|$, to obtain $z_{k^\star} = z(|\rho_{k^\star}|)$. Then, we can use the normal cumulative density function (cdf) $\mathcal{CN}_{0,\sigma_z^2}$ with mean zero to obtain a function that provides larger values for the correct key (i.e. where the z-transform of the absolute value of the correlation has higher values). After normalising this function, we obtain the associated probability of each candidate $k^\star$ as:

$$\Pr(k^\star|l) = \frac{\mathcal{CN}_{0,\frac{1}{n-3}}(z_{k^\star})}{\sum_{k'=0}^{2^a-1} \mathcal{CN}_{0,\frac{1}{n-3}}(z_{k'})}. \tag{4}$$

Alternatively, we can also use the mean of the transformed values $\bar{z} = \mathbf{E}_{k^\star}(z_{k^\star})$ as an approximation for $\mu_z$ (since we expect that $\mu_z > \bar{z} > 0$, so using this value should lead to a better heuristic than just assuming that all incorrect keys have a correlation close to zero).[1]

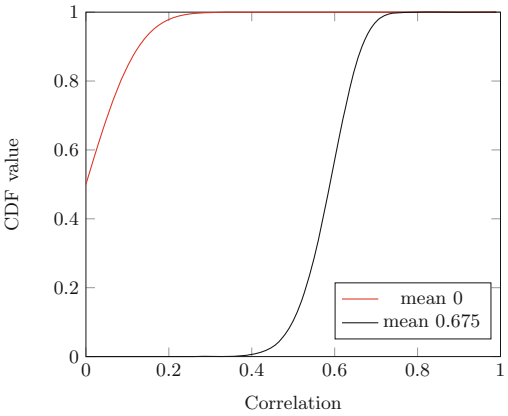


**Fig. 2.** Assumptions for $\mu_z$ in the CPA Bayesian extension.

[1] Yet another possible solution, which does not require knowledge of the key either, is to assume that the absolute correlation will be the highest for the correct key candidate, so the z-transform of the absolute correlation should also be the highest. Then, we can simply use the maximum among all the z-transformed values as the mean of the normal distribution for the z-transformed data (i.e. $\mu_z = \max_{k^\star}(z_{k^\star})$), hence obtaining $\Pr(k^\star|l) = \frac{\mathcal{N}_{\mu_z,\frac{1}{n-3}}(z_{k^\star})}{\sum_{k'=0}^{2^a-1} \mathcal{N}_{\mu_z,\frac{1}{n-3}}(z_{k'})}$. This did not lead to serious improvements in our experiments, but might provide better results in other contexts.

For illustration, Fig. 2 shows the evolution of the cdf-based probability function for these two choices of correlation mean (for an arbitrary variance of 0.01, corresponding to 103 attack traces taken from the following section). Since in this example, $\bar{z} = 0.6$, many incorrect keys have a correlation higher than 0.2. Hence, in this case, the zero-mean assumption should lead to considerably worse results than the $\bar{z}$ assumption, which we consistently observed in our experiments. In the following, we only report on this second choice.

Next, we also need an assumption for the variance $\sigma_z^2$. While the usual (statistical) approach would suggest to use $\sigma_z^2 = 1/(n-3)$, we observed that in practice, a variance $\sigma_z^2 = 1$ gave better results in our experiments. Since this may be more surprising, we report on both assumptions, and next refer to the attack using $\sigma_z^2 = 1/(n-3)$ as BCPA1 and to the one using $\sigma_z^2 = 1$ as BCPA2.

For illustration, Fig. 3 shows the evolution of the cdf-based probability function for BCPA1 and BCPA2 with different number of attack samples. The blue curve represents the BCPA2 method, which is invariant with the number of attack samples. The green, red and black curves shows the impact of the number of attack samples on the cdf. As we can see, the slope becomes more and more steep when the number of attack samples increases. This makes the key discrimination harder since at some point the variance becomes too small to explain the errors due to our heuristic assumption in the mean $\mu_z$.

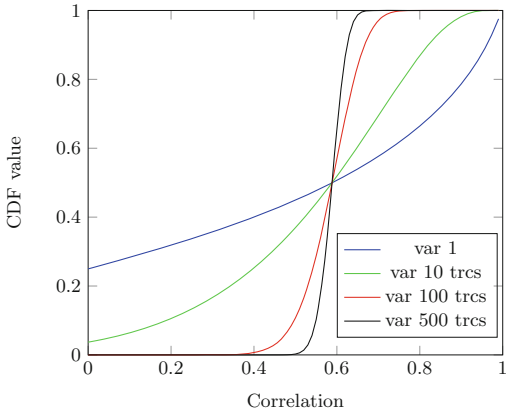


**Fig. 3.** Normal CDF with mean $\bar{z} = 0.6$ for variance 1 (blue), variance $\frac{1}{7}$ (green, 10 attack samples), variance $\frac{1}{97}$ (red, 100 attack samples) and variance $\frac{1}{497}$ (black, 500 attack samples). (Color figure online)

Summarizing, the formulae for the heuristics BCPA1 and BCPA2 are given by Eqs. (5) and (6):

$$\text{BCPA1} \;:\; \Pr[k^\star|l] = \frac{\mathcal{CN}_{\bar{z},\frac{1}{n-3}}(z_{k^\star})}{\sum_{k'=0}^{2^a-1}\mathcal{CN}_{\bar{z},\frac{1}{n-3}}(z_{k'})}. \tag{5}$$

$$\text{BCPA2} \ : \ \Pr[k^\star|l] = \frac{\mathcal{CN}_{\bar{z},1}(z_{k^\star})}{\sum_{k'=0}^{2^a-1} \mathcal{CN}_{\bar{z},1}(z_{k'})}. \tag{6}$$

**Gaussian Template Attack (GTA).** From a first set of profiling traces $\mathbf{l}_p = (l_i^{x,k})$, the attacker estimates the parameters of a normal distribution. He computes a model $\mathbf{m} = (m_{x,k^\star})$, where $m_{x,k^\star}$ contains the mean vector and the covariance matrix associated to the computation of $v_{x,k^\star}$. We consider the approach where a single covariance is computed from all the samples [5]. The probability associated to a subkey candidate $k^\star$ is computed as shown by (7):

$$\Pr[k^\star|l] = \prod_{i=1}^{n} \frac{\mathcal{N}_{m_{x_i,k^\star}}(l_i)}{\sum_{k'=0}^{2^a-1} \mathcal{N}_{m_{x_i,k'}}(l_i)}, \tag{7}$$

where $\mathcal{N}_m$ is the normal probability density function (pdf) with mean and covariance given by the model $m$.[2] As mentioned in introduction, it is important to note that combining probabilities from different subkeys implies a multiplicative relationship. Since the division in the computation of $\Pr(k^\star|l)$ only multiplies all the probabilities by a constant, this has no impact on the ordering when combining subkeys. We can simply omit this division, thus only multiplying the values of $\Pr[l_i|k^\star] = \mathcal{N}_{m_{x_i,k^\star}}(l_i)$. This may allow us to use an efficient template attack based on the linear discriminant [5], adapted so we can use different plaintexts (see Sect. 2.3 for more details).

**Unprofiled Linear Regression (LR).** Stochastic attacks [20] aim at approximating the deterministic part of the real leakage function $\theta(v_{x,k})$ with a model $\theta^*(v_{x,k})$ using a basis $g(v_{x,k}) = \{g_0(v_{x,k}), \ldots, g_b(v_{x,k})\}$ chosen by the attacker. For this purpose, he uses linear regression to find the basis coefficients $c = \{c_0, \ldots, c_b\}$ so that the leakage function $\theta(v_{x,k})$ is approximated by $\theta^*(v_{x,k}) = c_0 \cdot g_0(v_{x,k}) + \ldots + c_b \cdot g_b(v_{x,k})$. The unprofiled version of this attack simply estimates on the fly the models $\theta^*(v_{x,k^\star})$ for each key candidate [7]. The probability associated to a subkey candidate $k^\star$ is computed as shown by (8):

$$\Pr[k^\star|l] = \frac{\mathsf{Std}(\mathbf{1} - \theta^*(v_{x,k^\star}))^{-n}}{\sum_{k'=0}^{2^a-1} \mathsf{Std}(\mathbf{1} - \theta^*(v_{x,k'}))^{-n}}, \tag{8}$$

where $\mathsf{Std}$ denotes the sample standard deviation. As detailed in [21], a significant advantage of this non-profiled attack is that is produces sound probabilities without additional assumptions on the distribution of the distinguisher. So it does not suffer from combination ordering errors. By contrast, it may be slower to converge (i.e. lead to less efficient subkey recovery) since it has to estimate a model on the fly. In our experiments, we consider two linear basis. The first one uses the hamming weight ($\mathsf{HW}$) of the sensitive value: $g(v_{x,k}) = \{1, \mathsf{HW}(v_{x,k})\}$. The second one uses the bits of the sensitive value $v_{x,k}$, denoted as $b_i(v_{x,k})$. We refer to the first one as LRH and to the second one as LRL.

[2] In our experiments with GTA, we only considered one leakage sample per trace, so our distribution is univariate and the covariance matrix becomes a variance.

### 2.2 Key Enumeration

After the subkey recovery phase of a divide-and-conquer attack, the full key is trivially recovered if all the correct subkeys are ranked first. If this is not the case, a key enumeration algorithm has to be used by the attacker in order to output all keys from the most probable one to the least probable one. In this case, the number of keys having a higher probability than the actual full key (plus one) is called the key rank. An optimal key enumeration algorithm was first described in [21]. This algorithm is limited by its high memory cost and its sequential execution. Recently, new algorithms based on rounded log probabilities have emerged [2,12,17,19]. They overcome these memory and sequential limitations at the cost of non optimality (which is a parameter of these algorithms). Different suboptimal approaches can be found in [6,18].

### 2.3 Rank Estimation

Rank estimation algorithms are the tools associated to key enumeration from an evaluation point-of-view. Taking advantage of the knowledge of the correct key, they output an estimation of the full key rank which would have been outputted by a key enumeration algorithm with an unbounded computational power. The main difference with key enumeration algorithms is that they are very fast and allow to estimate a rank that would be unreachable with key enumeration (e.g. $2^{100}$). These algorithms have attracted some attention recently, as suggested by various publications [1,8,10,17,18,22,23]. For convenience, and since our following experiments are in an evaluation context, we therefore used rank estimation to discuss combination ordering errors. More precisely, we used the simple algorithm of Glowacz et al. [10] (using [1] or [17] would not affect our conclusions since these references have similar efficiencies and accuracies). With this algorithm, all our probabilities are first converted to the log domain (since the algorithm uses an additive relationship between the log probabilities). Then, for each subkey, the algorithm uses a histogram to represent all the log probabilities. The subkey combination is done by histogram convolution. Finally, the estimated key rank is approximated by summing up the number of keys in each bin from the last one to the one containing the actual key. We apply the same method for the correlation attacks, thus assuming a multiplication relationship for these scores too. Admittedly, and while it seems a reasonable assumption for the bayesian extension of CPA, we of course have no guarantee for the standard CPA. Note also that in this case the linear discriminant approach for template attacks in [5] becomes particularly interesting, because the linear discriminant is already in the logarithm domain, so can be used directly with the rank estimation algorithm, while providing comparable results to a key enumeration algorithm using probabilities derived from the normal distribution. Hence, this algorithm also avoids all the numerical problems related to the multivariate normal distribution when using a large number of leakage samples.

## 3 Experiments

In order to evaluate the results of the different attack approaches, we first ran simulated experiments. We target the output of an AES S-box leading to 16 univariate simulated leakages of the form $\mathsf{HW}(\mathsf{S}(x_i \oplus k_i)) + \mathcal{N}_i$ for $i \in [0, 15]$. $\mathsf{HW}$ represents the hamming weight function and $\mathcal{N}_i$ is a random noise following a zero-mean Gaussian distribution. For a given set of parameters, we study the full key rank using 250 attack repetitions, and compute the ranking entropy as suggested by [16]. By denoting $\mathsf{R}_i$ the rank of the full key for a given attack, we compute the expectation of the logarithm of the ranks given by $\mathsf{E}_i(\log_2(\mathsf{R}_i))$ (and not the logarithm of the average ranks[3] equals to $\log_2(\mathsf{E}_i(\mathsf{R}_i))$). The ranking entropy is closer to the median than the mean which gives smoother curves. However, we insist that using one or the other metric does not change the conclusions of our experiments.

### 3.1 Simulations with Identical S-Box Leakages

We first look at the case where the noise is constant for all attacked bytes, meaning $\mathcal{N}_i = \mathcal{N}_j$ (we used a noise variance of 10). We want to investigate if the full key rank is impacted by the way an attacker produces his probabilities or scores. Figure 4 (left) shows the full key rank in function of the number of attack traces for the different methods. One can first notice the poor behavior of BCPA1 (in red). Although it starts by working slightly better than CPA (in blue), it quickly becomes worse.

This is due to the variance that quickly becomes too small and does not allow to discriminate between the different key hypotheses. Secondly, we see that CPA and BCPA2 give very similar results in term of full key success rate. This can be explained since the noise level is constant for the different subkeys in this experiment. Hence, they are equally difficult to recover which limits the combination ordering errors. Thirdly, both LRH (purple) and LRL (purple, dashed) are not impacted by combination ordering errors. However, they suffer from the need to estimate a model. As expected, their slower convergence impacts LRL more, because of its large basis. By contrast, LRH produces the same results as the CPA and BCPA2. This suggest that the estimation errors due to a slower model convergence are more critical than the ordering errors in this experiment. (Note that none of the models suffer from assumption errors in this section, since we simulate Hamming weight leakages). Finally, the GTA (in black) provides the best result overall. Since the GTA is not impacted by combination ordering errors nor by model convergence issues, this curve can be seen as an optimal reference.

[3] Using the ranking entropy lowers the impact of an outlier when averaging the result of many experiments. As an example, let's assume that an evaluator does 4 experiments where the key is ranked 1 three times and $2^{24}$ one time. The ranking entropy would be equal to 6 while the logarithm of the average ranks would be equal to 22, thus being more affected by the presence of an outlier.

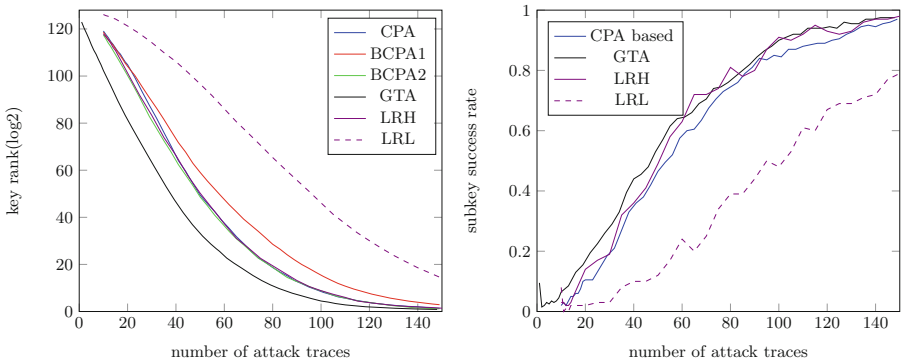


**Fig. 4.** Key rank (left) and success rate for the byte 0 (right) for constant noise variance in function of the number of attack traces for the different attack methods. CPA (blue), BCPA1 (red), BCPA2 (green), GTA (black), LRH (purple) and LRL (dashed purple). (Color figure online)

For completeness, Fig. 4 (right) shows the success rates of the different methods for the byte 0 (other key bytes lead to similar results in uur experiments were the SNR of different S-boxes was essentially constant). The single byte success rate does not suffer from ordering errors. Thus, this figure is a reference to compare the attack efficiencies before these errors occur. As expected, all the CPA methods (in blue) provide the same single byte success rate. More interestingly, we see that CPA methods, GTA (in black) and LRH (in purple) provide a quite similar success rate. This confirms that the differences in full key recovery for these methods are mainly due to ordering errors in our experiment. By contrast, LRL (in purple, dashed) provides the worst results because of a slower model estimation.

Overall, this constant noise (with correct assumptions) scenario suggests that applying a bayesian extension to CPA is not required as long as the S-boxes leak similarly (and therefore the combination ordering errors are low). In this case, model estimation errors are dominating for all the non-profiled attacks. (Of course, model assumption errors would also play a role if incorrect assumptions were considered).

### 3.2 Simulations with Different S-Box Leakages

We are now interested in the case where the noise level of each subkey is considerably different[4]. To simulate such a scenario, we chose very different noise levels in order to observe how these differences affect the key enumeration. Namely, we set the noise variances of the subkeys to $[20, 10, 5, 4, 2, 1, 0.67, 0.5, 0.33, 0.25, 0.2, 0.17, 0.14, 0.125, 0.11, 0.1]$. The results of the subkey recoveries for

[4] This could happen for example in a hardware implementation of a cryptographic algorithm, where each S-box lookup could involve different transistors.

each subkey will now differ because of the different noise levels. Figure 5 shows the rank of the different methods in function of the number of attack traces for this case. The impact of the combination ordering errors is now higher, as we can see from the gap between the different methods. BCPA1 is still worse than CPA, but this time with a bigger gap. Interestingly, we now clearly see an improvement when using the BCPA2 over the CPA (gain of roughly $2^{10}$ up to 40 attack traces). Moreover, the gap between BCPA2 and GTA remains the same as it was in the constant noise experiments as soon as the key rank is lower than $2^{80}$. This confirms the good behavior of the BCPA2 method, which limits the impact of the ordering errors.

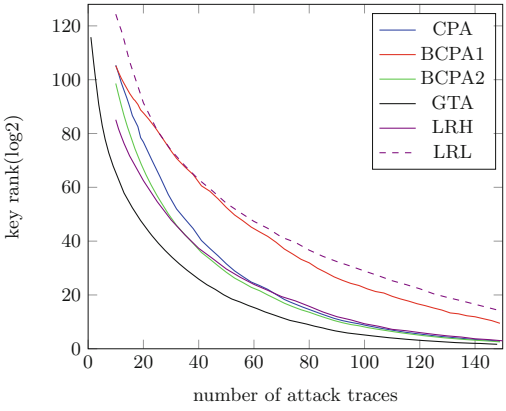


**Fig. 5.** Key rank (y coordinate) in function of the number of attack traces (x coordinate) with different noise variance for the different attack methods. CPA (blue), BCPA1 (red), BCPA2 (green), GTA (black), LRH (purple) and LRL (dashed purple). (Color figure online)

As for the linear regression, we again witness the advantage of a good starting assumption (with the difference between LRL and LRH). More interestingly, we see that for large key ranks, LRH leads to slightly better results than BCPA2, which suggests that combination ordering errors dominate for these ranks. By contrast, the two methods become again very similar when the rank decreases (which may be because of lower combination ordering errors or model convergence issues). And of course, GTA still works best.

We again provide the single bytes success rates for both byte 0 (left) and byte 8 (right) in Fig. 6. The gap between all the methods tends to be similar as in the constant noise case, where the CPAs, GTA and LRH perform similarly and LRL is less efficient. This confirms that the differences between the constant and different noise scenarios are due to ordering errors.

Overall, this experiment showed that the combination ordering errors can be significant in case the S-boxes leak differently. In this case, Bayesian extensions gain interest. BCPA2 provides a heuristic way to mitigate this drawback. Linear regression is a more systematic alternative (since it does not require assumptions in the distribution of the distinguisher).

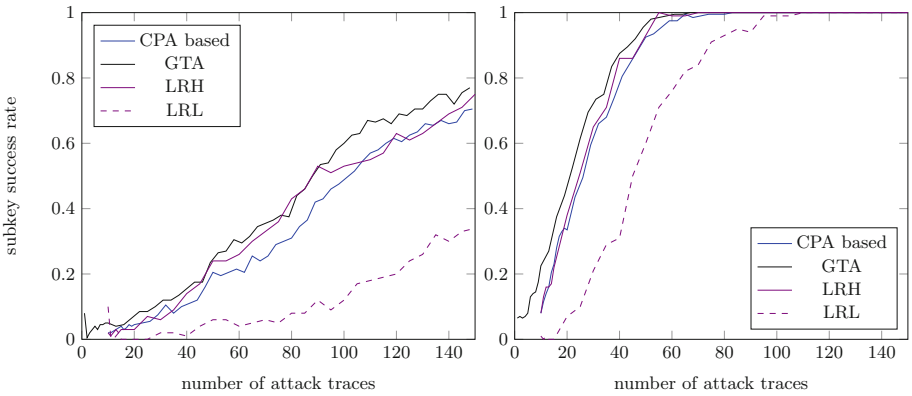


**Fig. 6.** Success rate (y coordinate) in function of the number of attack traces (x coordinate) for the byte 0 (left) and the byte 8 (right) (different noise variance case) for the different attack methods. CPA-like (blue), GTA (black), LRH (purple) and LRL (dashed purple). (Color figure online)

### 3.3 Actual Measurements

In order to validate our results, we ran actual attacks on an unprotected software implementation of the AES 128, implemented on a 32-bits ARM microcontroller (Cortex-M4) running at 100 MHz. We performed the trace acquisition using a Lecroy WaveRunner HRO 66 ZI oscilloscope running at 200 megasamples per second. We monitored the voltage variation using a 4.7 Ω resistor set in the supply circuit of the chip. We acquired 50,000 profiling traces using random plaintexts and keys. We also acquired 37,500 attack traces (150 traces per attack with 250 repetitions). For each AES execution, we triggered the measurement at the beginning of the encryption and recorded approximately the execution of one round. The attacks target univariate samples selected using a set of profiling traces as the one giving the maximum correlation with the Hamming weight of the S-box output.

Figure 7 again shows the key rank in function of the number of attack traces for the different attacks against the Cortex-M4 microcontroller. Interestingly, we can still see a small improvement when using the BCPA2 method over the CPA (around $2^5$). Looking at the linear regression results suggests that the Hamming weight model is reasonably accurate in these experiments (since LRH is close to GTA). Yet, we noticed that the correlation values for each S-box varied by approximately 0.1 around an average value of 0.45. We conjecture that this $2^5$ factor comes from small combination ordering errors. The figure also confirms the good behavior of the BCPA2 and LRH methods. By contrast, LRL is now even slower than BCPA1, which shows that the model estimation errors dominate in this case.

Once more, Fig. 8 shows the individual success rates for bytes 0 and 15. It confirms a that the different distinguishers behave in a similar way as in

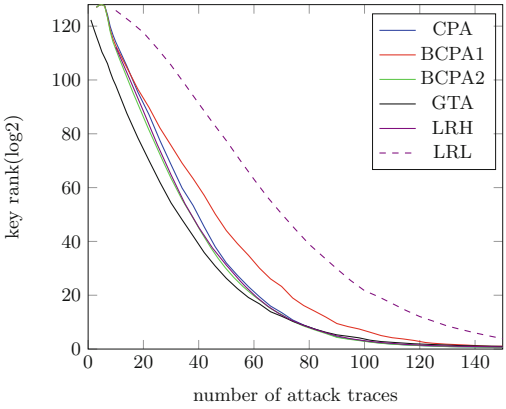


**Fig. 7.** Key rank (y coordinate) in function of the number of attack traces (x coordinate) on the cortex-m4 microcontroller for the different attack methods. CPA (blue), BCPA1 (red), BCPA2 (green), GTA (black), LRH (purple) and LRL (dashed purple). (Color figure online)

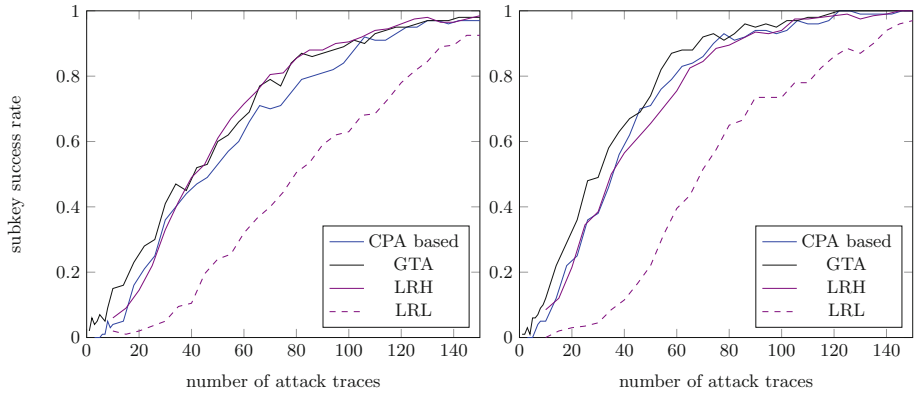


**Fig. 8.** Success rate (y coordinate) in function of the number of attack traces (x coordinate) for the byte 0 (left) and the byte 15 (right) (cortex-m4 microcontroller case) for the different attack methods. CPA-like (blue), GTA (black), LRH (purple) and LRL (dashed purple). (Color figure online)

simulations and therefore that the differences between the CPAs, GTA and LRH are again dominated by ordering errors. It also confirms that the errors from the LRL method come from model estimation.

Eventually, we stress again that the results would have been different if the real leakages were less close to the Hamming weight leakage function. In that case, model assumption errors would have additionally affected the efficiency of the non-profiled attacks.

### 3.4 Additional Heuristics

Before concluding, we note that various other types of heuristics could be considered to manipulate the scores produced by a CPA. For example, we assumed a multiplicative relation for those scores, but since we do not know how to combine them in a theoretically sound manner, one could also try to use other heuristics for the subkeys combinations in this case (such as summing the scores, summing the square of the scores, ...). We made some experiments in this direction, without any significant conclusions. These alternative heuristics sometimes came close to the efficiency of BCPA2 or LRH, but never provided better results.

## 4 Conclusions

Evaluating the security level of a leaking device is a complex task. Various types of errors can limit the confidence in the evaluation results, including the model assumption and estimation errors discussed in [9], and the combination ordering errors discussed in this paper. Overall, our results suggest that using GTA remains the method of choice to evaluate the worst-case security level of a leaking device (both to avoid incorrect a priori assumptions on the model and for optimal enumeration/rank estimation). Yet, good heuristics also exist for non-profiled attacks, in particular BCPA2 and LRH in our experiments.

Concretely, our results lead to the informal conclusion that model estimation and assumption errors usually dominate over combination ordering errors. This conclusion should be amplified for more complicated attacks (e.g. higher order attacks against masked implementations) where the model estimation is generally more challenging. So while being aware of combination ordering errors is important from a theoretical point-of-view, our experiments suggest that they are rarely the bottleneck in a security evaluation (which incidentally confirms previous works in the field, where enumeration and rank estimation were based on heuristic scores).

**Acknowledgments.** François-Xavier Standaert is a research associate of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in parts by the CHIST-ERA project SECODE and by the ERC project 280141. Marios O. Choudary has been funded in part by University Politehnica of Bucharest, Excellence Research Grants Program, UPB – GEX 2016, contract number 17.

## References

1. Bernstein, D.J., Lange, T., van Vredendaal, C.: Tighter, faster, simpler side-channel security evaluations beyond computing power. IACR Cryptology ePrint Archive, 2015:221 (2015)
2. Bogdanov, A., Kizhvatov, I., Manzoor, K., Tischhauser, E., Witteman, M.: Fast and memory-efficient key recovery in side-channel attacks. IACR Cryptology ePrint Archive, 2015:795 (2015)

3. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004). doi:10.1007/978-3-540-28632-5_2
4. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski, B.S., Koç, K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003). doi:10.1007/3-540-36400-5_3
5. Choudary, O., Kuhn, M.G.: Efficient template attacks. In: Francillon, A., Rohatgi, P. (eds.) CARDIS 2013. LNCS, vol. 8419, pp. 253–270. Springer, Heidelberg (2014). doi:10.1007/978-3-319-08302-5_17
6. David, L., Wool, A.: A bounded-space near-optimal key enumeration algorithm for multi-dimensional side-channel attacks. IACR Cryptology ePrint Archive, 2015:1236 (2015)
7. Doget, J., Prouff, E., Rivain, M., Standaert, F.-X.: Univariate side channel attacks and leakage modeling. J. Cryptogr. Eng. **1**(2), 123–144 (2011)
8. Duc, A., Faust, S., Standaert, F.-X.: Making masking security proofs concrete. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 401–429. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46800-5_16
9. Durvaux, F., Standaert, F.-X., Veyrat-Charvillon, N.: How to certify the leakage of a chip? In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 459–476. Springer, Heidelberg (2014). doi:10.1007/978-3-642-55220-5_26
10. Glowacz, C., Grosso, V., Poussier, R., Schüth, J., Standaert, F.-X.: Simpler and more efficient rank estimation for side-channel security assessment. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 117–129. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48116-5_6
11. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999). doi:10.1007/3-540-48405-1_25
12. Longo, J., Martin, D.P., Mather, L., Oswald, E., Sach, B., Stam, M.: How low can you go? using side-channel data to enhance brute-force key recovery. Cryptology ePrint Archive, Report 2016/609 (2016). http://eprint.iacr.org/
13. Mangard, S.: Hardware countermeasures against DPA – a statistical analysis of their effectiveness. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 222–235. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24660-2_18
14. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks - Revealing the Secrets of Smart Cards. Springer, Heidelberg (2007)
15. Mangard, S., Oswald, E., Standaert, F.-X.: One for all - all for one: unifying standard differential power analysis attacks. IET Inf. Secur. **5**(2), 100–110 (2011)
16. Martin, D.P., Mather, L., Oswald, E., Stam, M.: Characterisation and estimation of the key rank distribution in the context of side channel evaluations. IACR Cryptology ePrint Archive, 2016:491 (2016)
17. Martin, D.P., O'Connell, J.F., Oswald, E., Stam, M.: Counting keys in parallel after a side channel attack. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 313–337. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48800-3_13
18. Poussier, R., Grosso, V., Standaert, F.-X.: Comparing approaches to rank estimation for side-channel security evaluations. In: Homma, N., Medwed, M. (eds.) CARDIS 2015. LNCS, vol. 9514, pp. 125–142. Springer, Heidelberg (2016). doi:10.1007/978-3-319-31271-2_8
19. Poussier, R., Standaert, F.-X., Grosso, V.: Simple key enumeration (and rank estimation) using histograms: an integrated approach. IACR Cryptology ePrint Archive, 2016:571 (2016)

20. Schindler, W., Lemke, K., Paar, C.: A stochastic model for differential side channel cryptanalysis. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 30–46. Springer, Heidelberg (2005). doi:10.1007/11545262_3
21. Veyrat-Charvillon, N., Gérard, B., Renauld, M., Standaert, F.-X.: An optimal key enumeration algorithm and its application to side-channel attacks. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 390–406. Springer, Heidelberg (2013). doi:10.1007/978-3-642-35999-6_25
22. Veyrat-Charvillon, N., Gérard, B., Standaert, F.-X.: Security evaluations beyond computing power. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 126–141. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38348-9_8
23. Ye, X., Eisenbarth, T., Martin, W.: Bounded, yet sufficient? How to determine whether limited side channel information enables key recovery. In: Joye, M., Moradi, A. (eds.) CARDIS 2014. LNCS, vol. 8968, pp. 215–232. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16763-3_13