Key Enumeration from the Adversarial Viewpoint
When to Stop Measuring and Start Enumerating?

Melissa Azouaoui[1,2], Romain Poussier[3],
François-Xavier Standaert[1], Vincent Verneuil[2]

[1] Université Catholique de Louvain, Belgium
[2] NXP Semiconductors, Germany
[3] Temasek Laboratories, Nanyang Technological University, Singapore

Abstract. In this work, we formulate and investigate a pragmatic question related to practical side-channel attacks complemented with key enumeration. In a real attack scenario, after an attacker has extracted side-channel information, it is possible that despite the entropy of the key has been significantly reduced, she cannot yet achieve a direct key recovery. If the correct key lies within a sufficiently small set of most probable keys, it can then be recovered with a plaintext and the corresponding ciphertext, by performing enumeration. Our proposal relates to the following question: how does an attacker know when to stop acquiring side-channel observations and when to start enumerating with a given computational effort? Since key enumeration is an expensive (i.e. time-consuming) task, this is an important question from an adversarial viewpoint. To answer this question, we present an efficient (heuristic) way to perform key-less rank estimation, based on simple entropy estimations using histograms.

Keywords: Side-channel attacks, key rank estimation, key enumeration.

1 Introduction

Key enumeration and key rank estimation are important parts of the security evaluation of cryptographic implementations. These methods allow post-processing the side-channel attack outcomes and determine the computational security of an implementation with respect to a full key recovery. Key enumeration is an adversarial tool that allows testing key candidates without the knowledge of the key by listing the key candidates starting with the most likely one according to the attack results. For example, Veyrat-Charvillon et al. presented a deterministic algorithm for key enumeration [13] that allows the optimal enumeration of full keys by decreasing order of probabilities, by reformulating the key enumeration problem as a geometric problem. Ye et al. proposed a so-called key space finding algorithm [15], that returns the enumeration workload for a given success probability, by considering the number of optimal guesses for each subkey. Bogdanov et al. proposed a score-based key enumeration algorithm [2] that reduces the high computational and memory complexity of previous proposals. Martin et al. provided a new method [9] by casting the key enumeration as an integer knapsack problem.

On the other hand, rank estimation is a part of the side-channel evaluator's tool set: given lists of discrete probability distributions for independent parts of the key

and the correct key, a key rank estimation algorithm provides tight bounds for the position of the correct key among all possible ones (i.e. the number of guesses required to find the key following an optimal enumeration like above). In 2013, Veyrat-Charvillon et al. showed [14] that in the context of security evaluations (for which the key is usually known), it is possible to estimate the rank of a key beyond the evaluator's practical enumeration capabilities. Bernstein, Lange, and van Vredendaal improved this rank estimation proposals [1] and introduced a new method using polynomial multiplication to calculate lower and upper bounds for the ranks of all keys, by re-writing the problem of counting probabilities larger than the key's probability as finding the number of terms in a generalized polynomial satisfying a specific condition. In parallel, Glowacz et al. proposed a new tool for rank estimation [6] that is conceptually simple and very efficient, based on histogram convolutions, which was later extended to key enumeration by Poussier, Standaert, and Grosso [10]. At CHES 2017, a different and faster approach [5] was finally proposed by Choudary and Popescu. They suggested to bound the guessing entropy with information theoretic measures and inequalities, that do not require the knowledge of the correct key, estimated across multiple side-channel attacks.

In this paper, we tackle a different question than key enumeration or rank estimation, which lies somewhere in between. In a realistic attack scenario, after an adversary has performed a divide-and-conquer side-channel information extraction using a set of N observations, she obtains a vector of probabilities or scores for each subkey. The goal of the adversary is to perform a full key recovery. In this context, if all the subkeys have not been fully recovered immediately from the information extracted, the natural next step is key enumeration, up to a certain limit defined by her computational capabilities. Depending on the attacks, the cryptographic operation(s) targeted, and the size of the key, enumeration can become the most time- and resource-consuming step. Hence, it is very useful (for an adversary) to efficiently estimate the key rank, and therefore to know whether it is worth to start enumerating the key candidates or if the collection of more side-channel observations is needed. We believe that this is an important problem from an adversarial point-of-view, that has not yet been investigated.

We show in this article a simple solution for this purpose and propose a key-less heuristic (i.e. that doesn't require the knowledge of the key) to efficiently approximate the rank of a key given the result of a side-channel attack. Our solution is based on the key rank estimation method from Glowacz et al. [6] using histograms, which we combine with information theoretic metrics to predict the rank of the full key from one single attack. In particular, it differs from the solution of Choudary and Popescu [5] in the sense that it allows estimating the key rank of a single attack (rather than the estimation of the average key rank). Nonetheless, we compare an adaptation of their solution to this single attack setting with our method in Section 6.

The rest of the paper is organized as follows. First, Section 2 describes more precisely the problem under investigation. Secondly, Section 3 introduces the required background. Then, Section 4 discusses our proposed method to estimate the rank without the knowledge of the key from an attack perspective. Section 5 suggests an adaptation of the CHES 2017 proposal for rank estimation to the single-attack/attacker scenario. The results of the different methods are further shown in

Section 6. We additionally discuss the usability and some limitations of our method in 7. We finally conclude in Section 8.

2 Problem statement

Key enumeration algorithms can offer a trade-off between the required number of side-channel observations and the computational power of the adversary. That is, the use of such algorithms allows recovering the full key with fewer traces, often referred to as the data complexity of a side-channel attack, at the cost of computational/time complexity. However, in practice, it is not trivial for an adversary to know when to start the enumeration. Our goal is to tackle this problem by answering the following question: *Can the attacker infer if more side-channel measurements are required or if she expects a successful key recovery after enumeration, given the current attack outcome?*

To highlight the importance of this question, we picture the strategy that an adversary follows to recover the full key. When performing a side-channel attack it is common to follow a divide-and-conquer strategy, in which parts of the key are targeted and possibly recovered independently. For the rest of this paper, we use the same notations as in [6]. The target of a side-channel attack is an n-bit key $k \in \mathcal{K}$, divided into $N_p = \frac{n}{b}$ parts of b bits, called subkeys and denoted as k_i, for i in $[1 : N_p]$. A side-channel attack makes use of a set of q inputs $\mathcal{X}_q$ and the corresponding set of q leakages $\mathcal{L}_q$ (for example when targeting the AES S-box output, the attacker observes, given an input x_i^j $(1 \leq j \leq q)$, the leakage l_i^j of $\mathrm{S}(x_i^j \oplus k_i)$). After the attack, the adversary obtains N_p lists of probabilities $\Pr[k_i^* | \mathcal{X}_q, \mathcal{L}_q]$ where k_i^* refers to a subkey possibility out of the 2^b candidates. Attacks using Gaussian templates [4] or a linear-regression model [11] output probabilities, but for other distinguishers such as DPA [7] and CPA [3], a Bayesian extension is possible [13].

Let's assume that the adversary is able to perform key enumeration with respect to some computational effort e (e.g. $1 < e < 2^{50}$). A first attack strategy is shown in Algorithm 1. In that case, the attacker first collects a set of measurements, performs the attack, enumerates up to the first e most probable key candidates or until the correct key has been recovered. If the key has not been recovered, she then collects new side-channel observations and repeats the process.

Algorithm 1 Greedy attacker's strategy.

Input. Enumeration effort e.

1: Collect a set of side-channel measurements.
2: Update the attack N_p probability lists P_i for each subkey k_i.
3: Enumerate up to the first e key candidates or until the correct key k is found.
4: **if** k not found **then**
5: Collect new side-channel measurements and go to step 2.
6: **end if**

The greedy attacker strategy has one main drawback. Indeed, the attacker does not know if the key is reachable via enumeration after step 2. That is, she has no

way to know how many times she has to loop over steps 2 to 6. More specifically, if (e.g.) w repetitions of side-channel measurements are required for the attack, the adversary spends an effort of $w \times e$ in enumeration. As the time complexity of such a method is high, it mitigates the original goal of enumeration, which is to trade a lower measurement complexity for a higher computational power.

The aforementioned issue could be avoided if the adversary could assess if the key is reachable with an enumeration effort at most e, using the currently available measurements. Following this idea, we now assume that the attacker is provided with a tool that, from the probability lists P_i and without the knowledge of the actual key, returns an approximate $\hat{\mathrm{R}}$ of the actual rank R. Using this tool, a more efficient attack strategy is shown in Algorithm 2. With this method, enumeration is executed only if the approximated rank $\hat{\mathrm{R}}$ is found to be within the enumeration effort e. If $\hat{\mathrm{R}}$ is close enough to R, this method is obviously more optimal than the previous one and achieves the desired trade-off between side-channel observations and computational power.

Algorithm 2 Efficient attacker's strategy.

Input. Enumeration effort e, and a key-less rank estimation.

1: Collect a set of side-channel measurements.
2: Update the attack N_p probability lists P_i for each subkey k_i.
3: From P_i lists, compute an estimation $\hat{\mathrm{R}}$ of the rank R.
4: **if** $\hat{\mathrm{R}} \leq e$ **then**
5: Enumerate up to the first e key candidates or until the correct key k is found.
6: **if** k not found **then**
7: Collect new side-channel measurements and go to step 2.
8: **end if**
9: **else**
10: Collect new side-channel measurements and go to step 2.
11: **end if**

It is worth mentioning that, as a side advantage, the existence of such a tool would also give information to the adversary on whether or not the attack will succeed eventually. Indeed, observing the trend of $\hat{\mathrm{R}}$ either gives some confidence in the efficiency of the attack if it decreases steadily as the number of measurements increases, or shows that the attack has little chance to eventually recover the key (or a significant part of it) otherwise.

3 Background

In this section, we first define the metrics we investigate: the entropy, the rank and the guessing entropy. For the work described in this paper, we make use of the Glowacz et al. key rank estimation method [6], which we describe using the notations introduced in the previous section.

3.1 Entropy, rank and guessing entropy

Entropy. The Shannon entropy H of a discrete random variable $\mathrm{X} \in \mathcal{X}$ following a probability distribution Pr is defined as:

$$\mathrm{H}(\mathrm{X}) = -\sum_{x \in \mathcal{X}} \Pr(x) \cdot \log \Pr(x).$$

Rank. The rank R, after a side-channel attack using a set of q inputs $\mathcal{X}_q$ and a corresponding set of q leakages $\mathcal{L}_q$, provides the position of the correct key k in the sorted vector of $|\mathcal{K}| = 2^n$ key candidate probabilities $\mathbf{p} = [p_1, p_2, ..., p_{|\mathcal{K}|}]$, i.e:

$$\mathrm{R}(k) = i \text{ if } \Pr[k|\mathcal{X}_q, \mathcal{L}_q] = p_i.$$

Guessing entropy. The guessing entropy GE [12] measures the average number of key candidates to test after a side-channel attack. It corresponds to the average rank and is defined as:

$$\mathrm{GE} = \mathop{\mathbb{E}}_{k \in \mathcal{K}}(R(k)).$$

3.2 Key rank estimation

Given a set of discrete probability distributions for independent parts of a key and a correct key k, a rank estimation algorithm provides tight bounds for the rank among the set of all possible candidates. Among the different proposals for key rank estimation, we use the histogram-based approach of Glowacz et al. [6]. This algorithm provides efficiently tight bounds for the rank of the key and gives the probability distribution of the full key expressed as a histogram. Given the N_p lists of the log probabilities of the subkeys $LP_i = \log(\Pr[k_i^*|\mathcal{X}_q, \mathcal{L}_q])$, the method of Glowacz et al. starts by constructing the corresponding N_p histograms $H_i = \mathrm{hist}(LP_i, \mathrm{bins})$ where bins is a set of N_{bin} equally-sized bins, used for all the histograms. The convolution of two histograms is denoted as $\mathsf{conv}(H_i, H_j)$. Knowing the key k, its log probability is $\log(\Pr[k|\mathcal{X}_q, \mathcal{L}_q]) = \sum_{i=1}^{N_p} \log(\Pr[k_i|\mathcal{X}_q, \mathcal{L}_q])$. The following steps are described by Algorithm 3. In a nutshell, the rank estimation algorithm provides a very efficient way to estimate and bound the number of key candidates with a probability higher than k.

Algorithm 3 Rank estimation.

Input: The key log probability $\log(\Pr[k|\mathcal{X}_q, \mathcal{L}_q])$ and the histograms H_i.
Output: An approximation of k's rank.

initialization: $H_{\text{curr}} = H_1$;

histograms convolution:
for $i = 2 : N_p$
    $H_{\text{curr}} = \mathsf{conv}(H_{\text{curr}}, H_i)$;
end

rank estimation:

$$\text{estimated\_rank} \approx \sum_{i=\mathsf{bin}(\log(\Pr[k|\mathcal{X}_q,\mathcal{L}_q]))}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H_{\text{curr}}(i).$$

The estimated rank is then bounded by tracking the quantization error of the histograms as:

$$\text{rank\_lower\_bound} = \sum_{i=\mathsf{bin}(\log(\Pr[k|\mathcal{X}_q,\mathcal{L}_q]))+N_p}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H_{\text{curr}}(i),$$

and:

$$\text{rank\_upper\_bound} = \sum_{i=\mathsf{bin}(\log(\Pr[k|\mathcal{X}_q,\mathcal{L}_q]))-N_p}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H_{\text{curr}}(i),$$

By increasing the number of bins, the tightness of the bounds can be arbitrarily reduced. In the rest of this paper, and for the practical experiments, we use a number of bins large enough such that the tightness is below 1 bit. As this precision is enough for our experiments, we consider the estimated rank as the "true" rank and ignore the bounds. We emphasize that the histogram convolutions do not require the knowledge of the key, and only the rank estimation itself does. For the rest of this paper, we use the final histogram constructed, which corresponds to the full key distribution.

4 Using the entropy to approximate the rank

The entropy of the key candidate probabilities produced by an attack intuitively brings some information on the outcome of the attack, since it measures the uncertainty on the key, which amounts to the number of bits of information left to recover on the key. For instance, an attack ran on data that is uncorrelated with the key would tend to attribute average probabilities to most candidates, yielding a high entropy (n bits in the extreme case where all candidates have a probability of 2^{-n}, while the average rank is 2^{n-1}). On the other hand, an attack performing extremely well would give a high probability to the correct key candidate and very low ones to all other candidates. In that case, the entropy tends to 0 (like the rank)

as the probability of the wrong candidates also tends to 0. This intuition that in realistic cases entropy and rank are linked – although it's a loose link, as one can show that entropy and rank can diverge in specific cases – leads us to consider the use of the entropy to estimate the remaining enumeration effort of an attack.

In the following, we present how the entropy of the full key can be estimated using the histogram obtained with the convolution-based method from Glowacz et al. Although the histogram is only a compressed representation of the full key candidates' probability distribution, it still is an excellent tool to analyze it. To estimate the entropy, we require a proper probability distribution that sums up to 1. Thus, it is preferred to normalize the full key candidates' probability distribution. This is done by ensuring that the sum of the exponential of log probabilities given by the bins of all keys in all histograms sum to 1. Furthermore, it is recommended to normalize the distribution of the subkeys, prior to the histogram convolution, to avoid that the estimated metrics are weighted by the distributions of the subkeys. The entropy of the full key after a side-channel attack is estimated using the corresponding histogram (H,bin) as:

$$\begin{aligned} \mathrm{H} &= -\sum_{k^* \in \mathcal{K}} \Pr(k^*) \cdot \log \Pr(k^*) \approx \sum_{k^* \in \mathcal{K}} \exp(\mathsf{bin}(k^*)) \cdot \mathsf{bin}(k^*) \\ &\approx \sum_{i=1}^{N_p \cdot N_{\mathrm{bin}} - (N_p - 1)} H(i) \cdot \exp(\mathsf{bin}(i)) \cdot \mathsf{bin}(i) \end{aligned}$$

We denote by $\tilde{\mathrm{H}}$ the estimation of the entropy using the histogram. Note that bounds on this estimation can be computed. The corresponding formulas are given in Appendix A.

5 Adapting the CHES 2017 key-less GE

At CHES 2017, Choudary and Popescu [5] consider a number of information theoretic measures and inequalities that are easy and efficient to estimate in order to bound the guessing entropy of the key. However, in their proposal, they make a distinction between a key-agnostic guessing entropy and one that requires the knowledge of the key[1]. This key-less guessing entropy, that we denote by GE_{kl}, is computed given the sorted vector of the $|\mathcal{K}| = 2^n$ key candidate probabilities $\mathbf{p} = [p_1, p_2, ..., p_{|\mathcal{K}|}]$ obtained after a side-channel attack as:

$$\mathrm{GE}_{kl} = \sum_{i=1}^{|\mathcal{K}|} i \cdot p_i$$

The GE_{kl} is impossible to compute since the sum is over all full key candidates. For that purpose Choudary and Popescu use common measures and bounds described

[1] The framework [5] is actually misleading in this respect as it suggests that the GE is the *actual* key rank while it is the *average* key rank. The keyed and key-less versions are equivalent in case the templates used in the key-less estimation are perfect so the difference between both definitions only lies in the knowledge of the key.

in information theory literature. Since this work provides a key-less rank estimation tool, a natural alternative to our previous proposal is to try to adapt it to our single-attack context. This alternative is again only heuristic since the bounds are only valid on average [5]. To describe this alternative solution, we use again the histogram-based PDF estimation provided by the Glowacz et al. Here, GE_{kl} corresponds to the sum of the position of every key weighted by its probability and it can be estimated as:

$$\mathrm{GE}_{kl} = \sum_{i=1}^{|\mathcal{K}|} i \cdot p_i \approx \sum_{k^* \in \mathcal{K}} \left(\sum_{j=\mathsf{bin}(k^*)}^{N_p \cdot N_{\mathrm{bin}} - (N_p - 1)} H(j) \right) \cdot \Pr[k^*]$$

Then, we sum across all histograms and corresponding log probability bins, yielding the estimation of the key-less guessing entropy $\tilde{\mathrm{GE}}_{kl}$:

$$\tilde{\mathrm{GE}}_{kl} \approx \sum_{i=1}^{N_p \cdot N_{\mathrm{bin}} - (N_p - 1)} \left(\sum_{j=i}^{N_p \cdot N_{\mathrm{bin}} - (N_p - 1)} H(j) \right) \cdot \exp(\mathsf{bin}(i))$$

Again, new bounds on the key-less guessing entropy estimation can be computed from the histogram approximation, which are given in Appendix A. However, these bounds are not directly relevant to our work. Precisely, the ones in [6] require the knowledge of the key (i.e. an evaluation setting) while we aim at key-less rank estimation, and the ones in [5] only become tight on average over many experiments (i.e. also in an evaluation setting) while we aim to estimate the rank of single experiments on-the-fly. For the same reason, bounds on the entropy are also irrelevant in our adversarial setting. Subsequently, the rest of this paper ignores the bounds and only focus on the heuristic evaluation of these metrics.

6 Simulated and real experiments

In the following sections, we investigate whether the previous metrics can be used to approximate the rank of the correct key after a single side-channel attack. In that purpose, we evaluate two average absolute differences (for multiple side-channel attacks): the first between the logarithm of the rank and the estimated entropy $\tilde{\mathrm{H}}$, and the second between the logarithm of the key rank and the logarithm of $\tilde{\mathrm{GE}}_{kl}$. For that purpose, we use both simulated and real experiments. We start by describing our experimental setups and then show practical results.

6.1 Experimental setups

Simulated leakages: We simulated side-channel leakages of the 16 S-box outputs of the first round of an unprotected AES. For each byte x_i out of the 16, we model the leakage of x_i as $\mathsf{HW}(x_i) + b$, where HW denotes the Hamming weight function, and b represents an independent noise distributed according to a normal distribution with mean 0 and standard deviation 10.

Real leakages: We target a custom constant-time C implementation of an unprotected AES with T-tables. The code runs on an ARM Cortex-M3 microcontroller

running at 83 Mhz, mounted on an Arduino Due board. We acquired EM measurements with a Langer near field RF-U 5-2 probe and a Lecroy 610Zi oscilloscope at a sampling rate of 1GHz. We synchronized the traces at the beginning of the AES computation using a trigger signal. As for the simulated case, we target the output of the S-boxes of the first round. For each of the 16 target bytes, we selected the point of interest that exhibit the highest correlation with the corresponding S-box output value, using a first set of 10,000 traces with a known key.

We performed a template attack [4] for both simulated and real leakage experiments. For the real experiments, we use a set of 100,000 traces with known key and plaintext for the template building phase.

6.2 Results

We show here the practical results using the estimation of the entropy $\tilde{\mathrm{H}}$ and the estimation of the key-less guessing entropy $\tilde{\mathrm{GE}}_{kl}$ to approximate the rank of the key R after a side-channel attack. We additionally compare the performance of both metrics. We recall that we estimate the rank using the histogram method of Glowacz et al. and that we used a large enough number of bins so that the bounds are tight enough for us to use the estimated rank and ignore the bounds. In the following, we refer to the approximation of log(R) using the entropy as the $\tilde{\mathrm{H}}$-based approximation and the one using the logarithm of the key-less guessing entropy as the $\tilde{\mathrm{GE}}_{kl}$-based approximation.

As a preliminary experiment, we ran a single attack against both simulated and real traces. We incremented the number of attack traces until the rank reached one. The results are depicted in Figure 1. The left (resp. right) part of the figure shows the results for simulated (resp. real) traces. In each case, the X-axis represents the number of attack traces, and the Y-axis is used to represent the different metrics in $\log_2$ scale. We notice that the entropy-based approximation is closer to the logarithm of the rank than the one based on the key-less guessing entropy. More specifically, for the simulated traces the $\tilde{\mathrm{H}}$-based approximation remains within less than 10 bits of log(R), while the gap with the $\tilde{\mathrm{GE}}_{kl}$ goes up to 25 bits. The real experiments show less optimistic results. Indeed, while $\tilde{\mathrm{H}}$ remains mainly within 5 bits of log(R), we can observe a fairly large gap of 30 bits when the rank is around 2^{40}. Results are worse for $\log(\tilde{\mathrm{GE}}_{kl})$ with a maximal gap of 50 bits.

The previous experiment showed some results for a single attack and that the entropy is neither an upper bound or a lower bound on the logarithm of the rank. While this gives some insight about the interest of the considered metrics, it does not allow deriving any general conclusion. Next, we examine how the $\tilde{\mathrm{H}}$-based approximation and the $\tilde{\mathrm{GE}}_{kl}$-based one perform on average over many independent experiments. For that purpose, we evaluate the average absolute difference in bits, first between $\tilde{\mathrm{H}}$ and log(R), then between $\log(\tilde{\mathrm{GE}}_{kl})$ and log(R). This is performed over 100 independent attacks for the simulated and the real traces. The results on the simulated traces are shown in Figure 2. The X-axis corresponds to log(R). The Y-axis on the left (resp. right) part of the figure gives the mean (resp. standard deviation) of the distance between each measure and the rank. For the left part of the figure, we additionally plot the maximal distance obtained over the 100 attacks

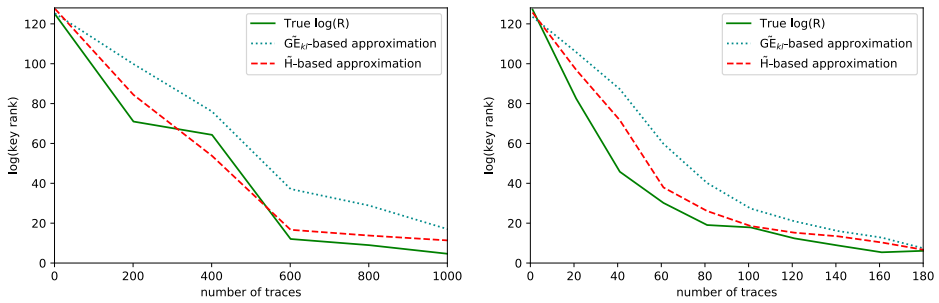

Fig. 1. Comparison of $\log(\tilde{GE}_{kl})$, $\tilde{H}$ and log(R) for a single random side-channel attack on the AES S-box output. Simulated traces (left) and real EM traces (right).

in corresponding dashed curves. We notice that the entropy clearly seems to outperform the key-less guessing entropy when estimating the rank for a single attack. First focusing on the average distance on the left part of the figure, the entropy stays within less than 10 bits of R. The maximal difference we obtained among the 100 experiments is slightly above 20 bits. However, the mean distance of log(R) to $\log(\tilde{GE}_{kl})$ is in most cases above 10 bits and below 25 bits, with a maximum above 45 bits. Also considering the standard deviations of these distances on the right part of the figure, it is reasonable to consider that the entropy provides a more reliable estimation of R than the key-less guessing entropy: since the standard deviation of the distance to $\tilde{H}$ is lower than the one to $\log(\tilde{GE}_{kl})$, the first one is less likely to deviate from its mean value which is below 10 bits.

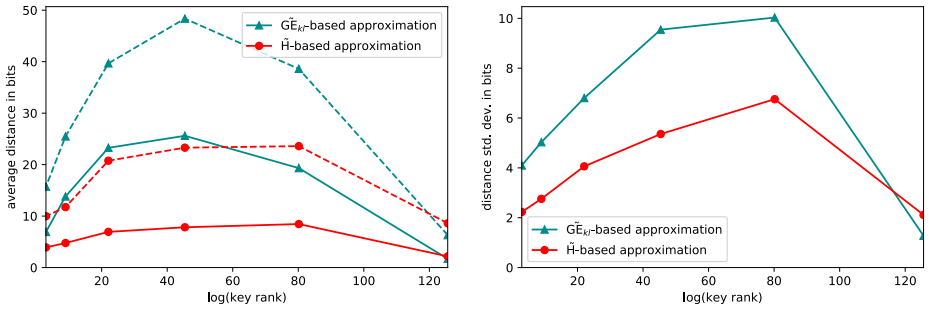

Fig. 2. Simulated HW leakages: (Left) The average distance from the rank to both the $\tilde{H}$-based approximation and the $\tilde{GE}_{kl}$-based approximation as function of the logarithm of the rank and the maximal distance observed in corresponding dashed lines. (Right) the distances' standard deviations.

The results for real traces are shown in Figure 3. Again, the X-axis represents log(R) and the Y-axis on the left (resp. right) part of the figure represents the

mean (resp. standard deviation) of the distances, computed over 100 independent attacks. The experiments on real traces coincide with the simulated ones. Accordingly, the entropy-based approximation provides a better estimate of log(R) than the approximation based on the key-less guessing entropy. The average distance between log(R) and $\tilde{\mathrm{H}}$ is below 12 bits while the average distance to $\log(\tilde{\mathrm{GE}}_{kl})$ is always above the average distance to $\tilde{\mathrm{H}}$. The right part of Figure 3 is similar to the right part of Figure 2, as the standard deviation of the distance to the entropy is lower than the one to the key-less guessing entropy, confirming that for a real side-channel attack, $\tilde{\mathrm{H}}$ provides a better approximation of log(R) than $\log(\tilde{\mathrm{GE}}_{kl})$.

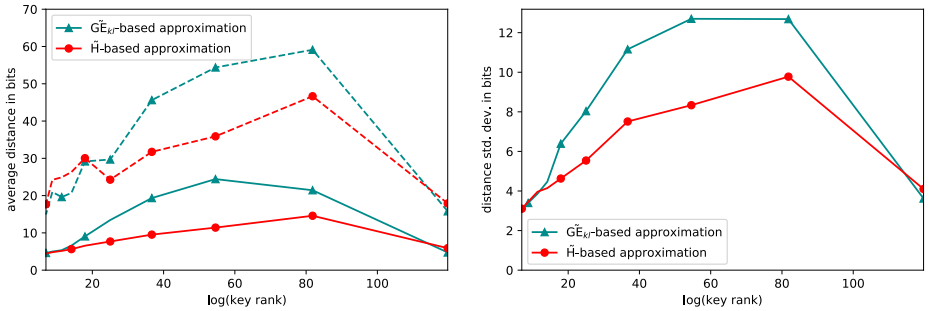

Fig. 3. EM traces: (Left) The average distance from the rank to both the $\tilde{\mathrm{H}}$-based approximation and the $\tilde{\mathrm{GE}}_{kl}$-based approximation as function of the logarithm of the rank and the maximal distance observed in corresponding dashed lines. (Right) the distances' standard deviations.

We highlight the fact that all predictions have a higher variance/standard deviation for middle ranks, which are the most interesting for both evaluators and attackers, as it is typically the range of ranks where enumeration turns from being unfeasible to practically feasible. This has been previously observed by Martin et al. in an evaluation setup [8] with the possibility to perform multiple attacks with the knowledge of the key. Our results and the ones of Martin et al. show that the interesting ranks are the hardest to estimate for an evaluator, and especially in the context of a real attack with the purpose of recovering the key.

Our proposed metric cannot mathematically approximate nor bound the true rank of the key after a single side-channel attack. However, we experimentally show that the entropy tends to stay within reasonable limits from the logarithm of the rank (provided that the attack does not suffer from errors, for e.g. due to a wrong model assumption). As a result, we believe it can be used in a more efficient strategy by trading data complexity for computational effort as illustrated by Algorithm 2 by enumerating key candidates up to (or slightly above) $2^{\tilde{\mathrm{H}}}$.

7 Discussion and limitations

Any attempt to predict the rank from one single attack without the knowledge of the key suffers from a specific caveat. It is possible that the attack is not carried out correctly and is converging towards a wrong key (due for example to wrong intermediates, wrong assumptions about the leakage or unknown countermeasures). The entropy and the key-less guessing entropy would then decrease as the attack tends towards the wrong key candidate, while the rank of the correct key would not. This behavior does not only affect the entropy and the key-less guessing entropy but most probably any metric estimated without the knowledge of the correct key.

Another aspect to consider that affects the considered metrics is the key size. This can be pictured through a simple example: let's consider two attacks that both aim at recovering a bit b whose value is 1, and output two probability distributions $\mathbf{p_1} = [\Pr_1[b = 0] = 0\,, \Pr_1[b = 1] = 1]$ and $\mathbf{p_2} = [\Pr_2[b = 0] = 0.45\,, \Pr_2[b = 1] = 0.55]$. Both attacks achieve a rank of one since the correct value of b has the highest probability. On the other hand, the entropy values are quite different. The entropy of the first attack is equal to 0, which is equal to the logarithm of the rank. For the second attack, the entropy of b is higher and equal to 0.99277, albeit the correct value of b is ranked first. These discrepancies can be observed for small keys, but vanish for larger key sizes. This illustrates how independent conclusions on subkeys can be quite misleading when trying to infer conclusions on full key recovery.

To demonstrate this effect, we performed the same experiments as described in the previous section. We estimated the average distance between log(R) and the entropy and then between log(R) and the key-less guessing entropy, but across different key sizes. For each key size, we performed 100 attacks. We normalized the distance with respect to the size of the key in bytes. Indeed, normalizing makes the distances comparable for different key sizes, and allows to infer conclusions based on the distance per byte. As an example, a minor distance for one key byte between R and its key-less prediction is critical, but not so relevant for the full key. The results are given in Figure 4, for both the simulated traces on the left and real traces on the right. The dashed line indicates the maximum values observed. For the simulated experiments, it was possible to perform experiments on large keys of up to 64 bytes, and up to the AES-128 key size for the real traces we measured. We used 400 attack traces for the simulations, leading to a rank of approximately 2^{55}, and 70 traces for the real attack to achieve a rank around 2^{30} for a 128-bit key (with proportional ranks for smaller key sizes). This was chosen to focus on the interesting ranks and we did not notice any considerable differences for other ranks, when it comes to the effect of the key size on the distance to log(R) of either the $\tilde{\mathrm{H}}$-based approximation or the $\tilde{\mathrm{GE}}_{kl}$-based one. As we can see, the normalized difference indeed decreases when the key size increases, confirming our intuition. Moreover, the trend starts to settle for both simulated and real traces once realistic full key sizes are reached. First, for a 1-byte key size, we can observe on average a two-bit difference between log(R) and $\tilde{\mathrm{H}}$ and a lower difference between log(R) and log($\tilde{\mathrm{GE}}_{kl}$). On the other hand, for a 16-byte key size, the distance between the rank and the entropy-based prediction drops to around 0.5 bits of error per byte for both the simulated and the real traces, while the distance between the rank and $\tilde{\mathrm{GE}}_{kl}$ seems to settle at an average distance of 1.5 bits of error per byte

even for larger key sizes. For the maximal value, we observe the same trend as previous experiments. The distance to the key-less guessing entropy is higher than the one to the entropy in most cases. Overall, two conclusions can be drawn from this experiment. First, it confirms that the entropy-based estimation seems to be a better tool to approximate the rank than the key-less guessing entropy once real key sizes are reached. Second, it shows that as expected, it is better to estimate the security level in an adversarial scenario on the full key than on a small part of the key, such as a subkey.

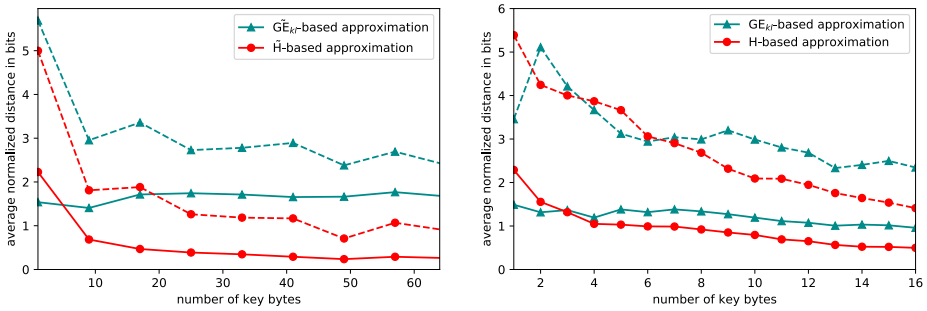

Fig. 4. Average distances between the log of the rank, the entropy and log(GM) as function of the number of key bytes. Maximal distances observed in corresponding dashed lines. Simulated traces on the left and real traces on the right.

8 Conclusion

In this paper, we described a heuristic way to infer an approximation of the key rank for one single attack without the knowledge of the key. This corresponds to a realistic attack scenario, where the adversary aims at figuring out if the correct key can be reached through enumeration. Our proposal helps to devise an optimal attack strategy to trade data complexity for computational effort when possible. For that purpose, we showed that the remaining entropy of the full key can be estimated using the histogram built with the rank estimation method from Glowacz et al. without the knowledge of the key. We showed experimentally that the entropy of the full key distribution after a side-channel attack is close to the logarithm of the rank on both simulated data and real EM side-channel measurements of an AES implementation. We compared this entropy-based approximation of the rank, to a single-attack adaptation of the key-less rank estimation method of Choudary and Popescu [5]. We additionally discussed factors that may affect the accuracy of the entropy (and any measure that lacks knowledge of the key or its probability) as a predictor of the logarithm of the rank. Further research might investigate if the behavior observed in this paper is common to different side-channel datasets. Moreover, it would be interesting to investigate if the tool described in this work

can help to identify possible wrong assumptions about the implementation or device that can possibly hinder the success of the attack. Alternatively, an interesting direction is to propose a more precise technique or metric to approximate the rank of the correct key in the single attack scenario.

Acknowledgement. François-Xavier Standaert is a senior research associate of the Belgian Fund for Scientific Research. This work has been funded in part by the European Commission through the H2020 project 731591 (acronym REASSURE) and by the ERC Consolidator Grant 724725 (acronym SWORD). The authors acknowledge the support from the 'National Integrated Centre of Evaluation' (NICE), a facility of Cyber Security Agency, Singapore (CSA).

References

1. Daniel J. Bernstein, Tanja Lange, and Christine van Vredendaal. Tighter, faster, simpler side-channel security evaluations beyond computing power. *IACR Cryptology ePrint Archive*, 2015:221, 2015.
2. Andrey Bogdanov, Ilya Kizhvatov, Kamran Manzoor, Elmar Tischhauser, and Marc Witteman. Fast and memory-efficient key recovery in side-channel attacks. In Orr Dunkelman and Liam Keliher, editors, *Selected Areas in Cryptography – SAC 2015*, pages 310–327, Cham, 2016. Springer International Publishing.
3. Eric 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*, pages 16–29, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
4. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski, çetin K. Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, pages 13–28, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
5. Marios O. Choudary and P. G. Popescu. Back to massey: Impressively fast, scalable and tight security evaluation tools. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems – CHES 2017*, pages 367–386, Cham, 2017. Springer International Publishing.
6. Cezary Glowacz, Vincent Grosso, Romain Poussier, Joachim Schüth, and François-Xavier Standaert. Simpler and more efficient rank estimation for side-channel security assessment. In Gregor Leander, editor, *Fast Software Encryption*, pages 117–129, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.
7. Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 388–397, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.
8. Daniel P. Martin, Luke Mather, Elisabeth Oswald, and Martijn Stam. Characterisation and estimation of the key rank distribution in the context of side channel evaluations. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 548–572, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
9. Daniel P. Martin, Jonathan F. O'Connell, Elisabeth Oswald, and Martijn Stam. Counting keys in parallel after a side channel attack. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015*, pages 313–337, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

10. Romain Poussier, François-Xavier Standaert, and Vincent Grosso. Simple key enumeration (and rank estimation) using histograms: An integrated approach. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems – CHES 2016*, pages 61–81, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
11. Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, pages 30–46, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
12. François-Xavier Standaert, Tal G. Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, pages 443–461, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
13. Nicolas Veyrat-Charvillon, Benoît Gérard, Mathieu Renauld, and François-Xavier Standaert. An optimal key enumeration algorithm and its application to side-channel attacks. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography*, pages 390–406, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
14. Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Security evaluations beyond computing power. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 126–141, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
15. Xin Ye, Thomas Eisenbarth, and William Martin. Bounded, yet sufficient? how to determine whether limited side channel information enables key recovery. In Marc Joye and Amir Moradi, editors, *Smart Card Research and Advanced Applications*, pages 215–232, Cham, 2015. Springer International Publishing.

A Error bounds on the histogram estimations

The bounds on the estimation of the entropy and the key-less guessing entropy using the Glowacz et al. full key distribution histogram and based on its quantization error are given by:

$$\text{H\_upper\_bound} = \sum_{i=1}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H(i).\exp(\mathsf{bin}(i + N_p)).\mathsf{bin}(i + N_p)$$

$$\text{H\_lower\_bound} = \sum_{i=1}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H(i).\exp(\mathsf{bin}(i - N_p)).\mathsf{bin}(i - N_p)$$

$$\text{GE}_{kl}\text{\_upper\_bound} = \sum_{i=1}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} \left(\sum_{j=i-N_p}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H(j) \right) . \exp(\mathsf{bin}(i + N_p))$$

$$\text{GE}_{kl}\text{\_lower\_bound} = \sum_{i=1}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} \left(\sum_{j=i+N_p}^{N_p \cdot N_{\text{bin}} - (N_p - 1)} H(j) \right) . \exp(\mathsf{bin}(i - N_p))$$