IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2020, No. 2, pp. 26–48. DOI:10.13154/tches.v2020.i2.26-48

A Fast and Accurate Guessing Entropy Estimation Algorithm for Full-key Recovery

Ziyue Zhang[1], A. Adam Ding[1] and Yunsi Fei[2]

[1] Department of Math., Northeastern University, Boston, MA, USA, a.ding@northeastern.edu
[2] Department of ECE, Northeastern University, Boston, MA, USA,

Abstract. Guessing entropy (GE) is a widely adopted metric that measures the average computational cost needed for a successful side-channel analysis (SCA). However, with current estimation methods where the evaluator has to average the correct key rank over many independent side-channel leakage measurement sets, full-key GE estimation is impractical due to its prohibitive computing requirement. A recent estimation method based on posterior probabilities, although scalable, is not accurate.

We propose a new guessing entropy estimation algorithm (GEEA) based on theoretical distributions of the ranking score vectors. By discovering the relationship of GE with pairwise success rates and utilizing it, GEEA uses a sum of many univariate Gaussian probabilities instead of multi-variate Gaussian probabilities, significantly improving the computation efficiency.

We show that GEEA is more accurate and efficient than all current GE estimations. To the best of our knowledge, it is the only practical full-key GE evaluation on given experimental data sets which the evaluator has access to. Moreover, it can accurately predict the GE for larger sizes than the experimental data sets, providing comprehensive security evaluation.

Keywords: Side-channel analysis · guessing entropy · i-th order success rate · multi-variate Gaussian distribution · additive score distinguisher

1 Introduction

The seminal work of differential power analysis [KJJ99] revealed a realistic new threat to crypto-systems: an adversary can learn the secret of crypto-algorithms using side-channel leakage information from a physical implementation. An important question is how to measure the vulnerability of a crypto-implementation against a side channel analysis (SCA). Different metrics, such as mutual information, success rate and Guessing entropy (GE), have been presented [SMY09]. While Mutual Information serves as an intermediate metric to measure the dependency of side-channel leakage on key information, the commonly used success rate measures the probability that an attacker correctly distinguishes the true key candidate given certain number of leakage measurements. With the typical divide-and-conquer approach in SCA of block ciphers, success rate works well for a single key byte but does not scale for the entire key with many bytes (e.g., a full key of AES-128 consists of 16 bytes where each byte is retrieved independently). GE is defined as the average rank of the true key among all key candidates across multiple data sets at certain size. The higher GE, the more wrong key guesses have to be checked before the true key value is evaluated. Thus, GE measures the average computation cost required for a successful SCA and is deemed an appropriate leakage evaluation metric.

Practically useful GE evaluation for full-key recovery is very challenging. Currently the typical GE estimation relies on an empirical method: finding the rank of the true key

Received: 2019-10-15 Accepted: 2019-12-15 Published: 2020-03-02

on many independent sets of side-channel measurements, and calculating the average rank. Such GE estimation has been conducted commonly for a single-byte subkey, while finding the exact rank of the correct full-key over all candidates is beyond available computational capacity. This issue is addressed by a recent work [VCGS13], with an algorithm to estimate the upper and lower bounds for the rank of the correct full-key. More efficient ranking algorithms have been further developed [GGP+15, BLvV15, MOOS15, DW17, Gro18]. As these ranking algorithms require access to a data set (e.g., with q measurement traces) to find the correct full-key rank over the set, the empirical GE estimation needs to average the ranks on N such independent data sets. Since the rank of the correct full-key generally follows a highly skewed distribution with a large variance [MMOS16], a very large number of N is needed for accurate GE estimation. Therefore, currently an evaluator can only assess the vulnerability of a system to SCA with some N and q that are attainable to their computation and collection capacity. The evaluator cannot predict the vulnerability to adversaries with larger q than the acquired measurement sets.

At CHES 2017, Choudary et al. [CP17] proposed estimating GE from an alternative quantity, Massey's guessing entropy (GM), rather than from the correct full-key rank. While they provide very scalable tight bounds for GM, GM calculation is based on the given data set and suffers the same practical problem of needing to average over many data sets for accuracy. It cannot predict the vulnerability of the system to an adversary with more resource capacity. Moreover, since GM is developed by using posterior probabilities of key candidates as the rank probabilities, the accuracy of GE is limited as we will demonstrate later in this paper.

This work proposes an accurate and fast GE estimation method based on the theoretical multivariate Gaussian distribution of the ranking score vectors. Previous work [Riv09, LPR+14, FLD12, FDLZ15] have used the multivariate Gaussian distribution of single-byte subkey score vectors to derive the success rate formula for subkeys. However, this multivariate Gaussian distribution has not been used to calculate either GE or the general i-th order success rate (which is the probability that the SCA ranks the true key as one of the top i candidates) for the multi-byte full-key.

There is a technical difficulty in using such theoretical formula directly, because the multivariate Gaussian distribution probability calculation involves high-dimensional integral whose exact computation is not feasible even for single-byte key evaluation. Therefore, when calculating single-byte subkey success rate, simulated sets of score vectors from this multivariate Gaussian distribution are repeatedly generated, and the proportion of successful subkey recoveries is taken empirically as the success rate and subsequently GE is derived too. We call such estimation the pseudo-theoretical estimator, as it involves simulated data sets from the theoretical distribution for calculating success rate and ranks of correct subkey. While it can give very accurate single-byte subkey GE estimation, as the number of key bytes increases the pseudo-theoretical GE estimator also becomes impractical as it encounters the same obstacles for the empirical GE estimator mentioned above. For a full key of B bytes, score vectors from the B multivariate Gaussian distributions are generated, the rank of the correct key is figured out using a ranking algorithm, and then the average rank is obtained over N simulated data sets. Due to the large variance of the rank, the pseudo-theoretical GE estimator requires a large number of N for accuracy which exceeds the available computational capacity when B is large.

Realizing that the method of averaging the correct key ranks may be the barrier for obtaining accurate GE from the theoretical multivariate Gaussian distribution, we look closely into theoretical properties of GE to facilitate fast and accurate full-key GE estimation. We discover an important property of GE, i.e, GE only depends on the pairwise success probabilities (the probability that the correct key score beats another key candidate score). This enables GE estimation without going through the i-th order success rates. Note that calculation of the i-th order success rates for the full key, even knowing

the B *multivariate* Gaussian distributions, is nontrivial. We express GE as the sum of pairwise success probabilities, which can be calculated directly from *univariate* Gaussian distributions. Since the univariate Gaussian probabilities are much faster to compute and are much less variable than the ranks, we can evaluate the sum through sampling the univariate Gaussian probabilities. This method is much more accurate than sampling the ranks as used in the empirical GE estimator and the pseudo-theoretical GE estimator. A GE estimation algorithm (GEEA) is developed accordingly, which can estimate the full AES-128 key GE accurately in practical computational time.

The paper is organized as follows. Section 2 introduces the concepts and notations about GE and success rates, reviews the practical issues of empirical GE estimation, and reviews related theoretical results on GE and success rates. Section 3 develops our GE estimation algorithm. Section 4 uses numerical experiments to compare our proposed GEEA and other GE estimation methods. We end the paper with conclusions in Section 5.

2 Background and Preliminaries

2.1 Guessing Entropy and i-th Order Success Rates

We consider SCAs that use physical side-channel measurements to learn about a secret cypto key $k \in \mathbb{K}$. The adversary conducts q queries to the device under attack, and for each query records the (plaintext) input $x_i \in \mathbb{X}$ and the leakage measurements $l_i \in \mathbb{L}$, for $i = 1, ..., q$. Based on the data sets $\mathcal{L} = (l_1, l_2, ..., l_q)$ and $\mathcal{X} = (x_1, x_2, ..., x_q)$, SCA derives a *score* $s(k|\mathcal{L}, \mathcal{X})$ reflecting the log-likelihood of each key candidate $k \in \mathbb{K}$ being the correct key. We denote the correct key value as k_c. Sorting the scores from highest to lowest, the position of $s(k_c)$ is the rank of correct key $rank(k_c|\mathcal{L}, \mathcal{X})$. If an adversary tries the i most likely key candidates, he/she can recover the secret key if and only if $rank(k_c|\mathcal{L}, \mathcal{X}) \leq i$. The success probability of such an attack is called the i-th order success rate (SR) of the SCA:

$$SR_i = \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) \leq i]. \tag{1}$$

Generally, the first-order success rate is studied most often in literature and is simply referred to as the success rate.

The average rank of the correct key with a given number of measurements (q) is called the Guessing Entropy [Mas94, SMY09]:

Definition 1. $GE_q = \mathbb{E}_{\mathcal{L},\mathcal{X}:|\mathcal{L}|=q,|\mathcal{X}|=q}[rank(k_c|\mathcal{L}, \mathcal{X})]$.

Since the rank of the correct key indicates the minimum number of key candidates that the SCA adversary has to try to be successful, GE_q reflects the expected amount of computational effort needed for a successful SCA given a number of q side channel measurements. The subscript "q" emphasizes the fact that GE depends on the size q. Throughout the paper, in places where the size q is evident and not essential to the discussion, we drop the subscript for brevity. GE is related to the success rates (first-order and higher-order) as

$$\begin{aligned} GE &= \sum_{i=1}^{|\mathbb{K}|} i \times \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) = i] \\ &= \sum_{i=1}^{|\mathbb{K}|} \sum_{a=1}^{i} \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) = i] \\ &= \sum_{a=1}^{|\mathbb{K}|} \sum_{i=a}^{|\mathbb{K}|} \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) = i] \\ &= \sum_{a=1}^{|\mathbb{K}|} \{\mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) = a] + \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) > a]\} \\ &= \sum_{i=1}^{|\mathbb{K}|} \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) = i] + \sum_{i=1}^{|\mathbb{K}|} \mathbb{P}_{\mathcal{L},\mathcal{X}}[rank(k_c|\mathcal{L}, \mathcal{X}) > i] \\ &= 1 + \sum_{i=1}^{|\mathbb{K}|} (1 - SR_i). \end{aligned} \tag{2}$$

Equation (2) indicates that GE can be calculated when all i-th order success rates are known. However, our analysis finds that GE can be calculated instead from the pairwise success rates, making GE estimation much faster and easier.

2.2 Issues of Empirical Estimation of GE

Typically GE_q is estimated empirically following Definition 1: collect N independent sets of $(\mathcal{L}_1, \mathcal{X}_1)$, ..., $(\mathcal{L}_N, \mathcal{X}_N)$ each with the size q. On each set, the evaluator finds the correct key's rank $rank(k_c|\mathcal{L}_j, \mathcal{X}_j)$. Then GE_q is estimated by:

$$\overline{GE}_q = \frac{1}{N}\sum_{j=1}^{N} rank(k_c|\mathcal{L}_j, \mathcal{X}_j). \tag{3}$$

Here we denote the empirical estimator with the bar on top. However, there are several issues to prevent this estimator $\overline{GE}_q$ from working for full-key evaluation in practice.

Issue 1. *Leakage measurements collection cost:* A total number of $N \times q$ leakage measurements are needed for one empirical $\overline{GE}_q$. The collection capacity generally limit the available number N, which, if not large enough, would affect the accuracy of full-key GE estimation. Also, with the empirical GE estimation method, $\overline{GE}_q$ cannot be predicted for any q value that exceeds the size of the evaluation data set.

Issue 2. *Large variation of $rank(k_c)$:* As pointed out by the prior work [MMOS16], "the key rank is a random variable with inherently large variation", and the large variance occurs when the GE value falls in "exactly the range in which the assumed enumeration capability of an adversary transitions from realistic to unrealistic". Such range is where GE would be most useful as a security metric and requires accurate estimation. From Equation (3), $Var(\overline{GE}) = \frac{1}{N}Var[rank(k_c)]$, the larger the N, the smaller the empirical $\overline{GE}$ variance. However large N makes the computational cost prohibitive as GE calculation requires running the heavy ranking algorithm for N times.

To address the first issue, we propose to estimate GE based on theoretical multivariate Gaussian distributions instead of specific data set, which will be discussed in Section 2.4 below. To solve the second issue, we further utilize a theoretical relationship between GE and the pairwise success rates derived in Section 3.1. This relationship allows us to calculate GE without finding $rank(k_c)$ by using the ranking algorithms, and the quantities involved have much less variance compared to the ranks.

2.3 Posterior Probability Based GM Bounds

At CHES2017, Choudary and Popescu proposed another estimation method for multi-byte GE [CP17]. As Equation (2) indicates, GE can be estimated if we know the i-th rank probability $\mathbb{P}[rank(k_c|\mathcal{L}, \mathcal{X}) = i]$ for all i. While these rank probabilities are not easy to get, the prior work [CP17] used the i-th largest posterior probability $\mathbb{P}_{post}(k|\mathcal{L}, \mathcal{X})$ as $\mathbb{P}[rank(k_c|\mathcal{L}, \mathcal{X}) = i]$. This estimator of GE is called GM [CP17]. Since the posterior probability for a multi-byte key is the product of the posterior probabilities of each byte subkey (assuming key byte scores are independent), they derived a bound for GM from the subkey posterior probabilities. The bounds are shown to be tight, easy to scale up and can be computed very fast.

However, the GM estimator also has some issues. First, since posterior probabilities are used, $GM(\mathcal{L}, \mathcal{X})$ are data set dependent. GM has to be averaged over N data sets, and hence they suffer the same practical issues of the empirical GE estimator as discussed in Section 2.2. Second, there is no theoretical guarantee that using posterior probabilities in GM gives the correct answer, i.e., $\mathbb{E}_{\mathcal{L}_q,\mathcal{X}_q}[GM_{\mathcal{L}_q,\mathcal{X}_q}] = GE_q$ is not proven. In fact, this equality does not always hold as we demonstrate theoretically in the Appendix, and empirically in Section 4.2.1. Third, the posterior probabilities are only available for SCAs with explicit leakage model, thus GM cannot be used to evaluate the new Deep Learning based SCAs without known leakage model [PSB+18].

2.4 Multivariate Gaussian Distribution of the Score Vector and Pseudo-Theoretical GE Estimation

One property we will use in our GE estimator is that the score vector approximately follows a multivariate Gaussian distribution. This has also been used in literature to derive theoretical formulas for single-byte key success rate [LPR+14, FDLZ15, Riv09]. As in [LPR+14, FDLZ15], we consider a $(|\mathbb{K}|-1)$-sized *comparison vector* $\vec{\Delta} = (\Delta_{k_g})_{k_g \in \mathbb{K}/\{k_c\}}$ used in SCA where

$$\Delta_{k_g} = s(k_g) - s(k_c). \tag{4}$$

Here k_c denotes the correct key value and k_g is any other wrong key guess, and $s(\cdot)$ function is the score vector used in SCA, e.g., the Pearson correlation in CPA and Difference-of-Mean in DPA. In SCA, the distinguisher picks the correct key k_c based on the score: when $s(k_c) > s(k_g)$, the correct key is successfully distinguished.

The comparison vector $\vec{\Delta}$ satisfies the definition of additive distinguisher [LPR+14]. Assuming that the leakage measurements $l_1, l_2, ..., l_q$ are independent of each other and follow the same distribution, then such an additive vector $\vec{\Delta}(\mathcal{X}, \mathcal{L}) = \frac{1}{q}\sum_{j=1}^{q}\vec{\Delta}(x_j, l_j)$ follows a multivariate Gaussian Distribution $\mathcal{N}(\vec{\mu}_\Delta, \frac{1}{q}\vec{\Sigma}_\Delta)$ [LPR+14] due to the Central Limit Theorem. Here $\vec{\mu}_\Delta$ and $\vec{\Sigma}_\Delta$ are respectively the mean vector and variance matrix of $\vec{\Delta}(x_1, l_1)$ with size $|\mathbb{K}| - 1$ and $(|\mathbb{K}| - 1) \times (|\mathbb{K}| - 1)$, respectively. A component of the comparison vector $\vec{\Delta}$ having positive value means that the corresponding k_g is chosen over k_c. Hence $rank(k_c) = Npos(\vec{\Delta}) + 1$, where $Npos(\vec{x})$ denotes the function who counts the number of positive components of input $\vec{x}$. Thus the theoretical GE value becomes the expectation of $Npos(\vec{\Delta}) + 1$ under this multivariate Gaussian distribution. That is,

$$GE \quad = \int_{\vec{\Delta}}[Npos(\vec{\Delta}) + 1] \cdot \phi(\vec{\Delta}; \vec{\mu}_\Delta, \tfrac{1}{q}\vec{\Sigma}_\Delta)\ d\vec{\Delta}, \tag{5}$$

where $\phi(\cdot; \vec{\mu}, \vec{\Sigma})$ denotes the probability density function for the $\mathcal{N}(\vec{\mu}, \vec{\Sigma})$ distribution. For single-byte subkey, $\vec{\mu}_\Delta$ and $\vec{\Sigma}_\Delta$ can be profiled from measurement traces as shown in prior work [Riv09].

There are a couple of practical issues computing GE from formula (5): (a) analytic or numerical evaluation of high-dimensional integral is computational prohibitive so that direct evaluation using (5) is impractical even for single-byte subkey; (b) the dimensions of $\vec{\Delta}$ and $\vec{\Sigma}_\Delta$ for full-key case are very high, since $|\mathbb{K}|$ has to be large enough to prevent cryptanalysis. It is challenging but imperative to find a way to evaluate (5) not from the full-key multi-variate Gaussian distribution but by combining the B subkey multivariate Gaussian distributions. For example, for AES-128, we need a method to evaluate by combining the 16 multivariate Gaussian distributions (each for a single-byte subkey), rather than the impossible direct evaluation from the $2^{128} - 1$ dimensional full-key multivariate Gaussian distribution.

We first evaluate single-byte GE, and we can compute (5) empirically to overcome the issue (a): generate N samples $\vec{\Delta}_1, ..., \vec{\Delta}_N$ from the profiled multivariate Gaussian distribution $\mathcal{N}(\vec{\mu}_\Delta, \frac{1}{q}\vec{\Sigma}_\Delta)$, then estimate the integral using the empirical average of $Npos(\vec{\Delta}_j) + 1$ among $j = 1, ...N$. We define this method as the pseudo-theoretical GE (PS-TH-GE) estimator:

$$\overline{GE}_{TH,q} = \frac{1}{N}\sum_{j=1}^{N}[Npos(\vec{\Delta}_j) + 1], \tag{6}$$

for $\vec{\Delta}_1, ..., \vec{\Delta}_N$ generated from $\mathcal{N}(\vec{\mu}_\Delta, \frac{1}{q}\vec{\Sigma}_\Delta)$. We note that this is also the method used in estimating single-byte success rates before. While [LPR+14, FDLZ15] derives the theoretical multivariate Gaussian distribution, the ith-order success rate formula

$\int_{\vec{\Delta}} \mathbb{1}_{Npos(\vec{\Delta})+1\leq i}\phi(\vec{\Delta};\vec{\mu}_\Delta,\vec{\Sigma}_\Delta)d\vec{\Delta}$ is not directly evaluated with high-dimensional integral. Rather, this integral is also computed using the corresponding empirical quantity on $\vec{\Delta}_1,...,\vec{\Delta}_N$ generated from the theoretical multivariate Gaussian distribution. We define this pseudo-theoretical (PS-TH) method to emphasize that the evaluation uses samples generated from the theoretical distribution. For the full-key case, the usage of empirical quantity on generated samples runs into the issue of large variance for the GE evaluation as we discuss next. Such issue does not affect the first-order success rate evaluation which we will explain afterwards.

To solve issue (b), we make the assumption that the comparison scores for the key bytes are independent. Then comparison vectors can be generated separately for each key byte where the dimension of the multivariate Gaussian distribution is 2^8-1, and then $rank(k_c)=[Npos(\vec{\Delta})+1]$ can be found from the B byte-wise comparison vectors using the ranking algorithm. Averaging this $rank(k_c)$ over N such generations gives the PS-TH-GE estimation for the full-key. However, the PS-TH-GE estimator can run into similar difficulties as the empirical GE estimator. Note that the first issue of leakage measurements collection cost in Section 2.2 is greatly alleviated since PS-TH-GE estimator only requires N *generated* sets of comparison vectors rather than the N sets of real physical measurements. However, the second issue in Section 2.2 remains the same for the PS-TH-GE estimator: the large variance of $rank(k_c)$ [MMOS16] implies that accurate GE estimation will require such a large N value that it becomes computationally impractical. In summary, although the PS-TH-GE estimator explores multivariate Gaussian distributions rather than relies on data sets of real measurements, it still does not scale well for full-key cases. The reason is essentially that it still uses ranks, which are computationally prohibitive and require averaging over large independent simulated data sets.

Here we note that the computational issue for the full-key PS-TH estimator only applies to the GE evaluation but not for first-order success rate since the latter is based on the empirical average of the quantity $\mathbb{1}_{Npos(\vec{\Delta})=0}$ instead of $rank(k_c)$. While the variance of full-key $rank(k_c)$ is very big, the variance of $\mathbb{1}_{Npos(\vec{\Delta})=0}$ is upper bounded by $1/4$. Also, while the determination of $rank(k_c)$ requires a computationally intensive ranking algorithm, the determination of $\mathbb{1}_{Npos(\vec{\Delta})=0}$ is easy and fast since we only need to check that there are no positive components for each byte comparison vector separately.

The above discussion of PS-TH-GE estimator makes the assumption that the comparison scores for the key bytes are independent. This assumption is commonly made in many prior work on ranking algorithms and GE evaluation [VCGS13, GGP+15, BLvV15, MOOS15, DW17, Gro18, CP17]. Our work is addressing an issue of existing methods – inability to provide accurate GE estimation, under the same independent key byte scores assumption. Recently there are interests in extending the security evaluation work to cases where the key byte scores are dependent. Those cases are out of scope of this work and will be our future work.

We describe our theory-based GE estimation in the next section. The stark difference is our method computes the integral in equation (5) using a quantity that is much less variable than $rank(k_c)$ and is much faster to compute than $rank(k_c)$. This results in an accurate and fast GE estimation so as to make full-key GE estimation feasible.

3 Guessing Entropy Estimation Algorithm

In this section, we derive and describe our guessing entropy estimation algorithm (GEEA) based on a key insight on the relationship between GE and pairwise success rates. A SCA succeeds if the score for the correct key is higher than that of any other wrong key guesses. We define $\mathbb{P}[s(k_c)>s(k_g)]$ as the pairwise success rate.

3.1 Estimate Guessing Entropy from Gaussian Distributions

GE relates to the sum of pairwise success rates as stated in the following lemma:

Lemma 1. $GE = 1 + \sum_{k_g \neq k_c}[1 - \mathbb{P}[s(k_c) > s(k_g)]]$

Proof. Let $\mathbb{1}_{s(k_c)>s(k_g)}$ be the indicator that the correct key is successfully distinguished from k_g. Then $rank(k_c) = 1 + \sum_{k_g \neq k_c}[1 - \mathbb{1}_{s(k_c)>s(k_g)}]$. Hence

$$\begin{aligned} GE = \mathbb{E}[rank(k_c)] \quad &= 1 + \textstyle\sum_{k_g \neq k_c} \mathbb{E}[1 - \mathbb{1}_{s(k_c)>s(k_g)}] \\ &= 1 + \textstyle\sum_{k_g \neq k_c}\{1 - \mathbb{P}[s(k_c) > s(k_g)]\}. \end{aligned} \tag{7}$$

□

Denote the incorrect keys in $\mathbb{K}/\{k_c\}$ as $k_1, ..., k_{|\mathbb{K}|-1}$. Recall the definition of comparison vector Δ_{k_g} ($|\mathbb{K}| - 1$ dimension) by Equation (4) in Section 2.4. Then the i-th element $\vec{\Delta}_i$ in the vector $\vec{\Delta}$ follows the univariate Gaussian distribution $\mathcal{N}(\mu_i, \sigma_{ii}^2/q)$, where μ_i denotes the i-th element in $\vec{\mu}_\Delta$ and σ_{ij}^2 denotes the element in i-th row and j-th column of $\vec{\Sigma}_\Delta$. Hence the pairwise success rate is

$$\mathbb{P}[s(k_c|\mathcal{X}, \mathcal{L}) > s(k_i|\mathcal{X}, \mathcal{L})] = \mathbb{P}[\Delta_{k_i}(\mathcal{X}, \mathcal{L}) < 0] = \Phi(\frac{-\mu_i}{\sigma_{ii}/\sqrt{q}}) = 1 - \Phi(\frac{\sqrt{q}\mu_i}{\sigma_{ii}}), \tag{8}$$

where $\Phi(\cdot)$ denotes the cumulative distribution function (CDF) of $\mathcal{N}(0, 1)$. Plugging this theoretical pairwise success rate formula (8) into (7), we have the following theorem.

Theorem 1. *For an additive score distinguisher, assuming independent leakage measurements $l_1, l_2, ..., l_q$, then*

$$GE = 1 + \sum_{i=1}^{|\mathbb{K}|-1} \Phi(\frac{\sqrt{q}\mu_i}{\sigma_{ii}}). \tag{9}$$

In practice, we need to estimate the mean vector μ_i and diagonals of the variance matrix σ_{ii}^2. However, the given evaluation set $(\mathcal{X}_E, \mathcal{L}_E)$ is not an infinite set but has only $|\mathcal{L}_E|$ measurements, we have to estimate μ_i and σ_{ii}^2 using the sample mean and the sample variance of the comparison-scores, where the comparison-score for each key on each leakage measurement is $\Delta_{k,j} = s(k; l_j, x_j) - s(k_c; l_j, x_j)$ for $k \in \mathbb{K}$ and $j = 1, ..., |\mathcal{L}_E|$.

$$\hat{\mu}_i = \frac{1}{|\mathcal{L}_E|}\sum_{j=1}^{|\mathcal{L}_E|} \Delta_{k_i,j}, \quad \hat{\sigma}_{ii}^2 = \frac{1}{|\mathcal{L}_E|}\sum_{j=1}^{|\mathcal{L}_E|}[\Delta_{k_i,j} - \hat{\mu}_i]^2. \tag{10}$$

Then based on evaluation set $(\mathcal{X}_E, \mathcal{L}_E)$, our estimator of GE for any random $(\mathcal{X}, \mathcal{L})$ of size $q = |\mathcal{L}|$ is :

$$\widehat{GE}_q = 1 + \sum_{i=1}^{|\mathbb{K}|-1} \Phi(\frac{\sqrt{q}\hat{\mu}_i}{\hat{\sigma}_{ii}}) \tag{11}$$

Remark 1. *Note here we only need to estimate σ_{ii}^2, the diagonal elements of the variance matrix $\vec{\Sigma}_\Delta$, rather than the entire matrix as done before [Riv09]. This results in a much smaller number of variance parameters to be estimated: $|\mathbb{K}| - 1$ instead of $|\mathbb{K}|(|\mathbb{K}| - 1)/2$* [1].

[1] Using the confusion coefficients work [FDLZ15], we can further reduce the number of variance parameters to estimate from $|\mathbb{K}| - 1$ to 1 for DPA and CPA attack which does require an explicitly assumed leakage function. Generally we recommend the profiling approach of [Riv09] to estimate μ_i and σ_{ii}^2 to avoid the adverse effect of an inaccurately specified leakage model.

GE only depends on the sum of all i-th order success rates, as shown in equation (2). From equations (7), the sum of all i-th order success rates equals to the sum of pairwise success rates $\sum_{k_g \neq k_c} \mathbb{P}[s(k_c) > s(k_g)]$ *which are captured by the diagonal variances. Given the diagonal variances, varying the off-diagonal variance element can only affect how this sum is distributed among different i-th order success rates but not the sum itself. Let us consider the simple case of distinguishing only two wrong key candidates* k_{g1} *and* k_{g2} *from* k_c*. With two fixed diagonal variance elements* $\sigma^2_{k_{g1},k_{g1}}$ *and* $\sigma^2_{k_{g2},k_{g2}}$*, the probabilities of the successful distinction of* k_c *versus* k_{g1} *and the successful distinction of* k_c *versus* k_{g1} *are fixed, thus their sum is fixed. The off-diagonal variance element at* $\{k_{g1}, k_{g2}\}$ *is the covariances between those two successful distinctions, thus affect the probability of joint successful distinction. A bigger positive covariance indicates that the two pairwise distinctions tend to succeed together more often, and the first-order success rate (i.e., the probability of both pairwise distinctions succeed) is bigger. However, this also indicates that the two pairwise distinctions tend to fail together more often, thus the second-order success rate (i.e., one minus the probability of both pairwise distinctions fail) is smaller. The off-diagonal variance element's effect on the first-order success rate and its effect on the second-order success rate cancel each other out, and the sum remains the same. Overall, the off-diagonal variance element affect both the first-order success rate and the second-order success rate individually, but not the summation of them (the guessing entropy).*

Remark 2. *Some SCA involves a profiling stage to estimate some parameter* θ *in the score function* $s(k; l_j, x_j; \theta)$*, e.g., profiling in template attack or training a DNN. In such cases,* μ_i *and* σ^2_{ii} *should not be estimated on the same profiling set used to estimate* θ*, since overfitting can result in inaccurate* $\hat{\mu}_i$ *and* $\hat{\sigma}^2_{ii}$*. Rather, a separate evaluation set* $(\mathcal{X}, \mathcal{L})$ *should be used for estimating* $\hat{\mu}_i$ *and* $\hat{\sigma}^2_{ii}$ *after profiling* θ *is done. This is similar to the common idea of separating training and validation data in machine learning procedures.*

3.2 Guessing Entropy Estimation Algorithm (GEEA) for the Full-Key

As mentioned in earlier section, a big challenge in full-key GE evaluation is that the size $|\mathbb{K}|$ of the whole key space does not allow enumeration over it in practical time. Thus we need to consider how to decompose the calculation of Formula (11) by key bytes.

Assume the secret key of a block cipher has B bytes, with each byte of b-bits. Generally SCA is conducted separately on each byte following the divide-and-conquer principle. Let $k = (k^1, k^2, ..., k^B)$ with k^m denoting the m-th subkey byte, $m = 1, ..., B$.

The mean and variance of the subkey comparison-score $\Delta^m_{k_i^m} = s(k_i^m) - s(k_c^m)$ are estimated as before using Equation (10), separately for each $m = 1, ..., B$. Here we consider the typical case where the full key comparison score is the sum of byte comparison scores. This is under the assumption of independent key bytes and the byte raw scores are proportional to their log-likelihoods. Therefore the mean and variance of the full key score is respectively the sum of mean and variance of byte scores. Consequently, the comparison-score of the whole key k_g versus k_c is computed as $\Delta_{k_g} = \sum_{m=1}^{B} \Delta^m_{k_g^m}$ whose mean and variance estimators become:

$$\hat{\mu}_{k_g} = \sum_{m=1}^{B} \hat{\mu}^m_{k_g^m}, \qquad \hat{\sigma}^2_{k_g,k_g} = \sum_{m=1}^{B} (\hat{\sigma}^m_{k_g^m,k_g^m})^2. \tag{12}$$

Notice that for $k_g \neq k_c$, it is still possible that one of its bytes agrees with the correct key byte: $k_g^m = k_c^m$ for some m. When $k_g^m = k_c^m$ happens, since the byte comparison score of k_c^m with itself is always zero, $\hat{\mu}^m_{k_g^m}$ and $\hat{\sigma}^m_{k_g^m,k_g^m}$ take the value zero in (12).

According to Theorem 1, and using the mean and variance estimator expressions in

equation (12), GEEA calculates

$$\widehat{GE}_q = 1 + \sum_{k_g \in \mathbb{K}/\{k_c\}} \Phi(\frac{\sqrt{q}\hat{\mu}_{k_g}}{\hat{\sigma}_{k_g,k_g}}) = 1 + \frac{1}{|\mathbb{K}|-1} \sum_{k_g \in \mathbb{K}/\{k_c\}} f(k_g), \tag{13}$$

where $f(k_g) = (|\mathbb{K}|-1)\Phi(\frac{\sqrt{q}\hat{\mu}_{k_g}}{\hat{\sigma}_{k_g,k_g}})$. Comparing equations (7) and (13), we see that the $f(k_g)$ here is an estimator for the quantity $(|\mathbb{K}|-1)\{1-\mathbb{P}[s(k_c) > s(k_g)]\}$, a scaled version of (1− pairwise success rate).

For a full key with large B value, the enumeration through the key space (whose size is 2^{bB}) can be computationally prohibitive. In this case, we sample M guessed key k_g from the discrete uniform distribution on $\mathbb{K}/\{k_c\}$. Denote the set of sampled keys by $\mathcal{S}$, a sample version of $\widehat{GE}_q$ becomes

$$\widetilde{GE} = 1 + \frac{1}{M} \sum_{k_g \in \mathcal{S}} f(k_g). \tag{14}$$

where $M = |\mathcal{S}|$.

Note that $Var(\widetilde{GE}) = Var[f(k_g)]/M$. Hence for accurate $\widetilde{GE}$, the sampling rate M needs to be large enough so that $Var[f(k_g)]/M$ is reduced to achieve a specified accuracy.

We summarize GEEA in Algorithm 1 below [2]

Algorithm 1: GE Estimation Algorithm

Input : Key byte score distinguisher $s_{k^m;x,l}$;
Evaluating Set $(\mathcal{L}_E, \mathcal{X}_E)$;
Size of leakage measurement set q desired for GE prediction;

Intermediate : Key byte score matrices $S^m = (s^m_{i,j})_{2^b \times |\mathcal{L}|}$;
Estimation for $\hat{\tilde{\mu}}^m_\Delta$ and diagonal elements of $\hat{\tilde{\Sigma}}^m_\Delta$, $m = 1, ..., B$

Output : $\widetilde{GE}$ for $[\mathcal{L}, \mathcal{X}]$ of size q

ProfilingStage : Profile means and variances for *univariate* Gaussian distributions (the complexity is only $B \times (2^b - 1)$)

for $m \leftarrow 1$ **to** B **do**
  (for each key byte)
  for $i \leftarrow 1$ **to** $2^b - 1$ **do**
    (for key byte m, calculate the comparison vector (of size $2^b - 1$) based on the data set)
    $\hat{\mu}^m_i = \frac{1}{|\mathcal{L}_E|} \sum_{j=1}^{|\mathcal{L}_E|} [s(k^m_i; l_j; x_j) - s(k^m_c; l_j; x_j)]$
    $(\hat{\sigma}^m_{i,i})^2 = \frac{1}{|\mathcal{L}_E|} \sum_{i=1}^{|\mathcal{L}_E|} [s(k^m_i; l_j; x_j) - s(k^m_c; l_j; x_j) - \hat{\mu}^m_i)]^2$
  end
end

EvaluationStage : Calculate GE from univariate Gaussian probabilities
Create a random subset $\mathcal{S} = (k_1, ..., k_{|\mathcal{S}|}) \subset \mathbb{K}/\{k_c\}$ with size $M = |\mathcal{S}|$;

$$\widetilde{GE}_q = 1 + \frac{|\mathbb{K}|-1}{M} \sum_{k_i \in \mathcal{S}} \Phi(\frac{\sqrt{q}\sum_{m=1}^B \hat{\mu}^m_{k^m_i}}{\sqrt{\sum_{m=1}^B (\hat{\sigma}^m_{k^m_i,k^m_i})^2}})$$

Comparing to the empirical GE estimation and the pseudo-theoretical GE estimation both of which sample $rank(k_c)$, our GEEA samples $f(k_g)$ instead. The $rank(k_c)$ generally follows a highly skewed distribution with large variance. Our experiments in Section 4

[2] The GEEA estimator $\widetilde{GE}_q$ uses estimated means $\hat{\mu}^m_{k_g}$ and variances $(\hat{\sigma}^m_{k_g,k_g})^2$, thus requires accurate profiled values of these quantities. Since for each byte, there are 255 such means and 255 such variances, we recommend to profile them with number of measurement traces at least two order of magnitudes higher (i.e., $> 51K$ traces).

show that $f(k_g)$ follows a symmetric distribution with much smaller variance than that of $rank(k_c)$. Hence, with the same sampling rate $M = N$, our GEEA gives a much more accurate estimation $\widetilde{GE}$ than the empirical GE estimation $\overline{GE}$.

The two samplings are distinctively different: $\widetilde{GE}$ is sampling k_g over possible key candidates while $\overline{GE}$ is sampling $\mathcal{L}$ over possible sets of leakage measurements. Inspection of the formulas provides hints on why $\widetilde{GE}$ has much smaller variance than $\overline{GE}$. The reasons are twofold. First, $f(k_g)$ uses the expected value of $\mathbb{1}_{s(k_c)>s(k_g)}$ while $rank(k_c)$ uses the realization of $\mathbb{1}_{s(k_c)>s(k_g)}$. Variance of $\mathbb{P}(\mathbb{1}_{s(k_c)>s(k_g)} = 1)$ is generally much lower than the variance of $\mathbb{1}_{s(k_c)>s(k_g)}$. Second, $rank(k_c)$ is affected by the correlation between the comparison scores $\mathbb{1}_{s(k_c)>s(k_{g1})}$ and $\mathbb{1}_{s(k_c)>s(k_{g2})}$ for two different guessed key candidates k_{g1} and k_{g2}. These correlations are reflected by the off-diagonal elements of $\vec{\Sigma}_\Delta$, which cause clusters of guessed key candidates k_g to be simultaneously distinguished correctly (or incorrectly) against k_c. The empirical GE estimation $\overline{GE}$ and the pseudo-theoretical GE estimation $\overline{GE}_{TH}$ are both essentially calculating GE using all empirical i-th order success rates. Our GEEA uses only the pairwise success rates removing the correlation between key candidates k_{g1} and k_{g2}, which also leads to less variable estimation. These two reasons make the variance of $f(k_g)$ to be much smaller than the variance of $rank(k_c)$. Therefore, when both use the same number of sampling $N = M$, $\widetilde{GE}$ is much more accurate than $\overline{GE}$.

Furthermore, when using the same sampling rate $N = M$, our GEEA is faster by several orders of magnitude. In one sample($k_i \in \mathcal{S}$), GEEA only needs to do one univariate Gaussian CDF evaluation(Φ). In contrast, the empirical and pseudo-theoretical GE estimation both need to collect q traces and calculate (or generate) the score vectors to find the $rank(k_c)$. Just the running time of the ranking algorithm, to find $rank(k_c)$, is five orders of magnitude longer than the time for the univariate Gaussian CDF evaluation as measured in Section 4 below. The faster computation of GEEA allows it to use a sampling rate M several orders of magnitude higher than the sampling rate N of the empirical and pseudo-theoretical GE estimation with comparable computing time. In the next section, we conduct detailed numerical comparison of GEEA with other GE estimation methods on two real data sets. The numerical studies show that GEEA can achieve reliable full-key GE estimation while none of the state-of-the-art methods can.

4 Experimental Results

In this section, we conduct detailed numerical comparison of our GEEA $\widehat{GE}$ (for single-byte) and $\widetilde{GE}$ (for full key) with other GE estimation methods, namely the empirical GE (EMP-GE) estimation $\overline{GE}$, the pseudo-theoretical GE (PS-TH-GE) estimation $\overline{GE}_{TH}$ and GM [CP17] on two real power measurement data sets for AES implementations. We first show the performance in the one-byte case where $\overline{GE}$ does give reasonable estimates, to show its agreement with $\overline{GE}_{TH}$ and $\widehat{GE}$, while GM may not provide correct estimations. Then we study the performance as the key byte number increases, and the results demonstrate the advantage of GEEA for full-keys with multiple bytes.

4.1 Experimental Setup

4.1.1 Experiment Databases

For the first data set(SGII-1M), we implemented an unmasked AES-128 on a Sasebo-GII board [3] and collected 1 million power measurements with random plaintexts following uniform distribution. We estimate GE of the profiled template attacks on the AES SBox

[3] http://www.risec.aist.go.jp/project/sasebo/

lookups in the first round. The first 700,000 power traces are used to profile a Gaussian templates $GauTemp700k$ based on leakage measurements at the most leaky time-point. The template targets the SBox output $y = SBox[x \bigotimes k_c]$ where x and k_c respectively denote the plaintext and the secret key. Then the remaining 300,000 power traces are used to evaluate GE.

The second data set is the public benchmark ASCAD database [4] [PSB+18]. It provides a set of power traces from a first-order protected software AES running on an ATMega8515 board, as well as benchmarks for common SCAs such as the template attack and some Deep Learning-based SCAs in the first AES round. ASCAD data set contains three types of leakage measurement: aligned measurements, measurements with misalignment caused by insertion of randomly generate perturbations with two different bounds. Those measurements are labeled in the database as: leakage without desychronization; leakage with 50 maximum desychronization; leakage with 100 maximum desychronization. Each type of measurement contains 50,000 traces to train the Deep Learning Models and another 10,000 traces for testing. We use the test set to evaluate GE of those trained DL models.

4.1.2 GE Estimations

In this section, we review how the four GE estimations are calculated and outline some comparison results with details presented later in Section 4.2 and 4.3.

Given a profiled template or pre-trained ML-model, the evaluator collects another leakage measurement set $\mathcal{L}$ with size $|\mathcal{L}|$ to evaluate how powerful the attack is. To calculate GE of the attack using q leakage measurements, the two empirical method **EMP-GE** and **GM** first separates $\mathcal{L}$ into $\frac{|\mathcal{L}|}{q}$ independent subsets, and compute $rank_{k_c}$ and GM on each independent subset. Then those numbers are averaged over the $\frac{|\mathcal{L}|}{q}$ subsets to yield the estimations $\overline{GE}$ and GM. The two theoretical based estimators **PS-TH-GE** and **GEEA** calculate the GE value based on the multivariate Gaussian distributions whose parameters are profiled from the entire $\mathcal{L}$. For PS-TH-GE, score vectors are generated from those multivariate Gaussian distributions for each subset, $rank_{k_c}$ is calculated on each subset and then averaged across those subsets to get $\overline{GE}_{TH}$. For GEEA, $\widehat{GE}$ is calculated directly from (11) through summation over the key space when the key space is small (e.g., one key byte). When the key space is too big to enumerate, GEEA takes M samples of $f(k_g)$ and averages them to get $\widetilde{GE}$.

Note that there are uncertainties involved when the above methods use sampling. To quantify and show these uncertainties, we will draw the confidence bounds for those estimators. As there are at most $\frac{|\mathcal{L}|}{q}$ independent subsets, the estimators $\overline{GE}$ and GM becomes unreliable as q increases. In one byte case, we can generate many sets from the multivariate Gaussian distributions and get accurate $\overline{GE}_{TH}$ to agree with the exact theoretical value $\widehat{GE}$. For multiple bytes case, if we sample enough times, $\overline{GE}_{TH}$ and $\widetilde{GE}$ both converge to the theoretical GE value. However, our numerical studies will show that the computational cost of $\overline{GE}_{TH}$ is much higher than $\widetilde{GE}$, and only GEEA can give accurate useful GE estimation while other methods cannot.

4.2 Comparison of GE Estimations on a Single Key Byte

For the single key byte here, the $rank_{k_c}$ ranges from 1 to $2^8 = 256$ and thus has limited variance. This means that the state-of-art EMP-GE estimator $\overline{GE}$ can be reasonably

[4] https://github.com/ANSSI-FR/ASCAD

accurate, and we can check the correctness of the theoretical GE estimators from the comparison with $\overline{GE}$.

4.2.1 Comparison of GE Estimations on SGII-1M dataset

Figure 1 plots the estimations of GE_q, on the first data set SGII-1M, against the number of traces q for the four methods: EMP-GE, PS-TH-GE, GEEA, and GM. We also plotted the 95% confidence intervals for the EMP-GE $\overline{GE}$, which is based on $rank(k_c)$s from $N = \frac{300k}{q}$ independent subsets for each q value.

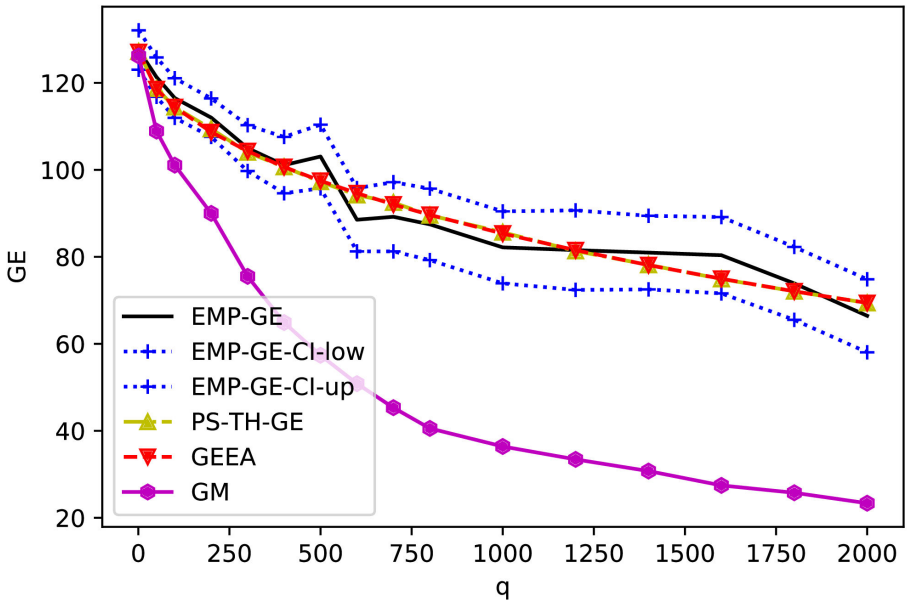


Figure 1: Comparison of four one-byte GE estimations (namely EMP-GE, PS-TH-GE, GEEA, and GM) of the template attack on SGII-1M data set. We also plot the 95% confidence intervals for EMP-GE, whose upper bound and lower bound are labeled respectively by EMP-GE-CI-up and EMP-GE-CI-low.

We can see that the two theoretical based estimators PS-TH-GE and GEEA are very close, and both fall within the confidence intervals of EMP-GE. This indicates good agreement among the three methods in this case. In contrast, the GM estimation GM is consistently much lower than other three estimations. As discussed in section 2.3, GM estimator is based on using posterior probabilities as the correct key ranking probabilities. Since these probabilities theoretically are not the same, GM may differ from GE as occurred in this case. Such inaccuracy will affect the security evaluation. For example, if security assessment requires $GE \geq 80$ when $q = 500$, the GM estimator will falsely claim that the AES implementation fails to pass the security specification with an GE estimation of 60 while the true GE value is much higher.

Furthermore, we can observe that the confidence interval of EMP-GE $\overline{GE}$ widens as q value increases. The reason is that there are only $N = \frac{300k}{q}$ independent subsets available, thus $\overline{GE}$ can become less accurate as q grows. The accuracy of $\overline{GE}$ will become a more serious issue in the multi-byte key case discussed in Section 4.3 below, since the variance of $rank(k_c)$ will be much larger in the range where the enumeration capability of an adversary becomes unrealistic.

Since the size of the key space is only 256, we can enumerate over it in Equation (11) and get the exact GEEA estimator $\widehat{GE}$. The PS-TH-GE estimator $\overline{GE}_{TH}$ is calculated using the average of $rank(k_c)$s based on $N = 10k$ samples of scores generated from the multivariate Gaussian distributions. We can see that PS-TH-GE curve overlaps with the GEEA curve, as their values agree with each other. The confidence intervals (omitted from the Figure) of $\overline{GE}_{TH}$ are very narrow and visually overlap with $\widehat{GE}$ if plotted. Here we used a large $N = 10k$ value so that PS-TH-GE $\overline{GE}_{TH}$ indeed converges to the theoretical value of GEEA estimator $\widehat{GE}$. The N value needed for $\overline{GE}_{TH}$ to accurately converge will increase as variance of $rank(k_c)$ increases, and the computational cost becomes an issue for $\overline{GE}_{TH}$ in the multi-byte cases studied later.

4.2.2 Comparison of GE Estimations on ASCAD dataset

We further compare the estimation methods of GE for Deep Learning-based SCAs using ASCAD database. Note that for the Deep Learning-based SCAs, there are no explicit leakage models and therefore no posterior probabilities. The GM estimation cannot be calculated here. In addition to empirical results in the previous subsection that shown GM is a very biased estimator for GE, the Appendix provides a theoretical explanation why GM is often more biased than GEEA.

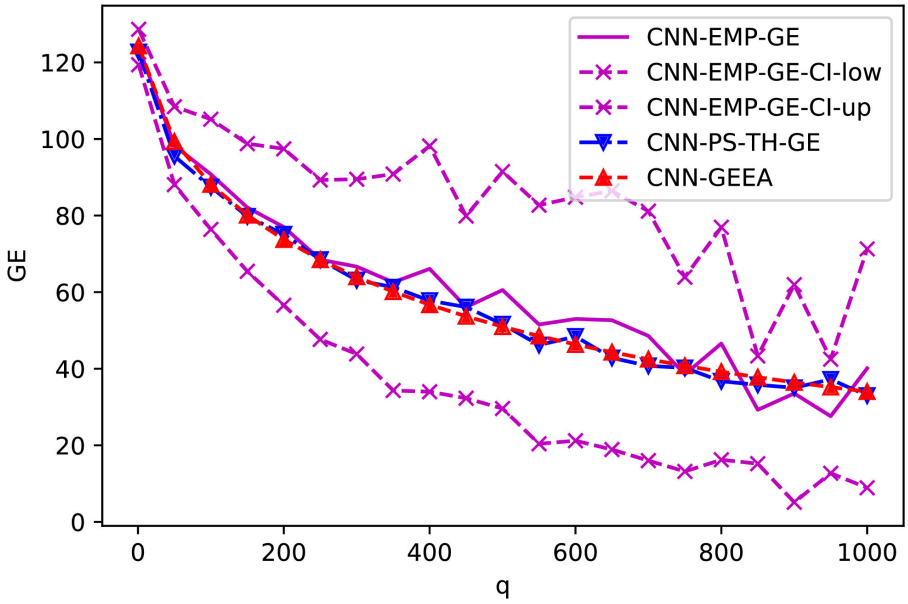
We plot the GE estimations for two types of DNN models attacks on the leakage measurements with 50 maximum desychronization : one using a multi-layer perceptron (MLP) network and the other using a convolutional neural network (CNN). We use the pre-trained networks provided in the database [PSB+18] which target the 3rd byte in the first round of a protected AES-128. With desychronization, the pre-trained MLP model cannot recover k_c while pre-trained CNN model can successfully identify k_c with enough traces. We evaluate GE of both attacks on the test set of leakage traces.

Figure 2(a)-(b) plot GEEA, PS-TH-GE and EMP-GE estimations, together with confidence intervals for EMP-GE. Again we see that GEEA and PS-TH-GE curves track each other, and both are well within the confidence intervals for EMP-GE. Here EMP-GE estimation is calculated using $N = \frac{10k}{q}$ independent subsets, while PS-TH-GE estimation is calculated using $N = 10k$ generated score sets. Note that the pre-trained MLP model attack fails on the desychronized data, which is reflected as the GEEA and PS-TH-GE curves actually increase as q grows and exceed $|\mathbb{K}|/2$. The theory based GEEA and PS-TH-GE can provide accurate GE estimations both when SCA succeeds and when SCA fails.

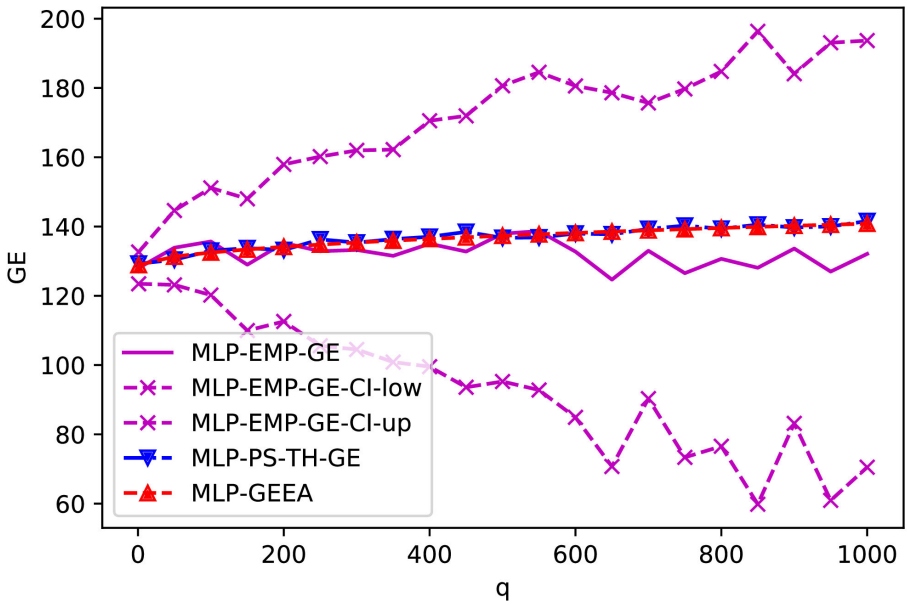
Here we have seen that, with only $N = \frac{10k}{q}$ independent subsets, $\overline{GE}$ becomes unreliable with very wide confidence intervals. $\overline{GE}_{TH}$ can converge to the theoretical value in both two cases by using a large $N = 10k$ value.

4.3 Comparison of GE Estimations in Multi-byte Cases

Since GE reflects the average computational effort needed for a successful SCA, it is most useful when evaluating the full-key attack. However, as mentioned above, large variance of $rank(k_c)$ in a large key space often means that the limited N independent subsets available is not enough to result in accurate EMP-GE $\overline{GE}$. This large variance happens particularly when the GE value ranges from 2^{40} to 2^{100} where the adversarial enumeration capacity changes from realistic to unrealistic, and accurate GE evaluation is most needed for security evaluation. When we calculate 95% confidence intervals of EMP-GE for the template attack on the full 16-byte AES key in the SGII-1M data set, the intervals are too wide (generally even include negative values) to be useful for security assessment.



(a) Comparison of three GE Estimators on CNN Attacks (EMP-GE, PS-TH-GE, and GEEA)



(b) Comparison of three GE Estimators on MLP Attacks (EMP-GE, PS-TH-GE, and GEEA)

Figure 2: Comparison of One byte GE estimations on two types of DNN-SCA using ASCAD database. We also plot the 95% confidence intervals for EMP-GE, whose upper bound and lower bound are labeled respectively by EMP-GE-CI-up and EMP-GE-CI-low.

Similar to EMP-GE, PS-TH-GE $\overline{GE}_{TH}$ is also based on $rank(k_c)$ and can use N generated sets where N is larger than the $|\mathcal{L}|/q$ independent sets available to EMP-GE. This allows PS-TH-GE to provide much tighter confidence intervals than those of EMP-GE. In the single-byte cases studied above, we see that PS-TH-GE does provide very accurate GE estimation. However, PS-TH-GE becomes much more computationally intensive as the byte number increases. In the following, we concentrate on comparing PS-TH-GE to our GEEA. The main focus is to study when and how the PS-TH-GE also becomes impractical due to its huge computational cost. Since PS-TH-GE can always use a larger N value than what is available to EMP-GE, whenever PS-TH-GE is impractical, it implies that EMP-GE should be impractical too.

As the byte number increases, the key space $\mathbb{K}$ grows so large that GEEA also needs to sample M terms of $f(k_g)$. We study the computational costs and accuracies between the sampling of $rank(k_c)$ by PS-TH-GE and sampling of $f(k_g)$ by GEEA. These studies will show that, as the key byte number increases, only our GEEA is able to provide useful GE assessment for the full-key case.

We first consider the computational cost comparison. In each sample, PS-TH-GE generates the scores from multivariate Gaussian distributions and then use the ranking algorithm to find one $rank(k_c)$. In contrast, GEEA only calculates one scaled probability $f(k_g)$ in each sample. To simplify, we ignore the generation costs of PS-TH-GE, and compare only the part of its computational times used by the ranking algorithm with that used by GEEA to calculate $f(k_g)$. Here we use the ranking algorithm FSE [GGP+15] in the PS-TH-GE implementation. In Table 1, we list the computational time for one sample of both PS-TH-GE and GEEA when the number of bytes in the full-key ranges from one to sixteen. Here we calculate GE by assuming all byte comparison vectors follow the multivariate Gaussian distribution with parameters estimated in the SGII-1M data above. The time reported is measured on High Performance Computing Cluster in our institute, whose computing nodes have dual Intel E5 2650 CPU's at 2.00 GHz or higher and 128 GByte of RAM or higher.

Table 1: Computational Comparison of PS-TH-GE and GEEA

Num of bytes	Time per sample(seconds)		
	PS-TH-GE	GEEA	Slowdown of PS-TH-GE
1	1.0		$9.1 * 10^3$
2	1.4		$1.3 * 10^4$
3	2.8		$2.5 * 10^4$
4	3.4		$3.1 * 10^4$
5	3.7		$3.4 * 10^4$
6	4.4		$4.0 * 10^4$
7	5.9		$5.4 * 10^4$
8	6.7	$\mathbf{1.1 * 10^{-4}}$	$6.1 * 10^4$
9	7.2		$6.5 * 10^4$
10	8.4		$7.6 * 10^4$
11	9.7		$8.9 * 10^4$
12	12.2		$1.1 * 10^5$
13	14.0		$1.3 * 10^5$
14	15.5		$1.4 * 10^5$
15	16.7		$1.5 * 10^5$
16	**17.4**		$\mathbf{1.6 * 10^5}$

We see that the computational time required by the ranking algorithm in PS-TH-GE approximately linearly increases as the number of bytes increases. The computational time for $f(k_g)$ in GEEA has no discernable increases since almost all its time is consumed by the one evaluation of the univariate Gaussian CDF, which remains the same for all key

length. There should be a linear increase in number of addition terms in the mean and the variance, but the time used by simple addition is negligible compared to the Gaussian CDF calculation.

The third column lists the ratio of the computational time for one sample in PS-TH-GE versus GEEA. The computational advantage of GEEA grows to 10^5 order for the full 16 bytes key, assuming the same sampling rate $M = N$.

However, PS-TH-GE cannot achieve the same accuracy as GEEA when using the same sampling rate $M = N$. To compare the accuracy, we evaluate the variance of the quantities $rank(k_c)$ and $f(k_g)$, used by each method respectively. Figure 3 plots the variances of $rank(k_c)$ and $f(k_g)$ in logarithm scale, versus the key length (from one to sixteen bytes).

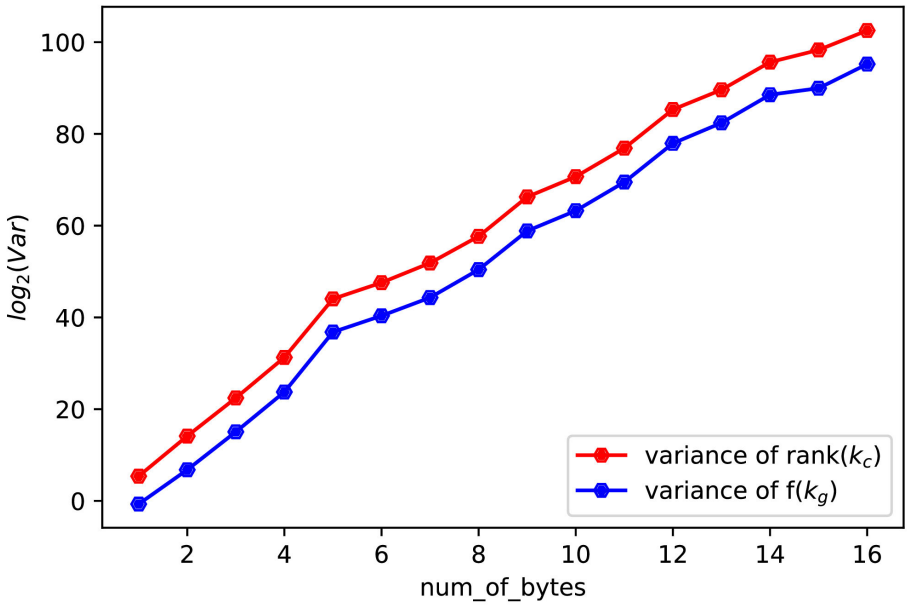


Figure 3: Comparison between variance of $rank(k_c)$ and variance of $f(k_g)$.

Figure 3 shows that the variances of both $rank(k_c)$ and $f(k_g)$ grow approximately linearly in log scale as the number of key bytes increases, and the variance of $rank(k_c)$ is about two order of magnitude larger than the variance of $f(k_g)$. To achieve the same accuracy with GEEA, PS-TH-GE needs to use a sampling rate of $N = M \cdot Var[rank(k_c)]/Var[f(k_g)]$. Joined with the comparison in Table 1, we plot the ratio of computation time needed for PS-TH-GE to achieve same accuracy as GEEA versus the key length in Figure 4. For the same accuracy, Figure 4 shows that GEEA will be seven orders of magnitude faster than PS-TH-GE in the full 16 bytes key case.

While the variance does provide a metric on the number of samples to get accurate estimation, the shape of probability distribution of the quantity also can affect the accuracy. For the same variance, a skewed distribution would require more samples to get good mean estimation than the samples required under symmetric bell-shaped distribution. As pointed out by [MMOS16], $rank(k_c)$ generally has a very skewed distribution. We find that the distribution of $f(k_g)$, on the other hand, is more symmetric. To see this, we plot the distributions of $rank(k_c)$s and $f(k_g)$s in Figure 6 when using $q = 1000$ traces to attack the first three bytes in the SGII-1M data set.

We choose the 3-bytes key case here because the key space $\mathbb{K}$ is small enough to

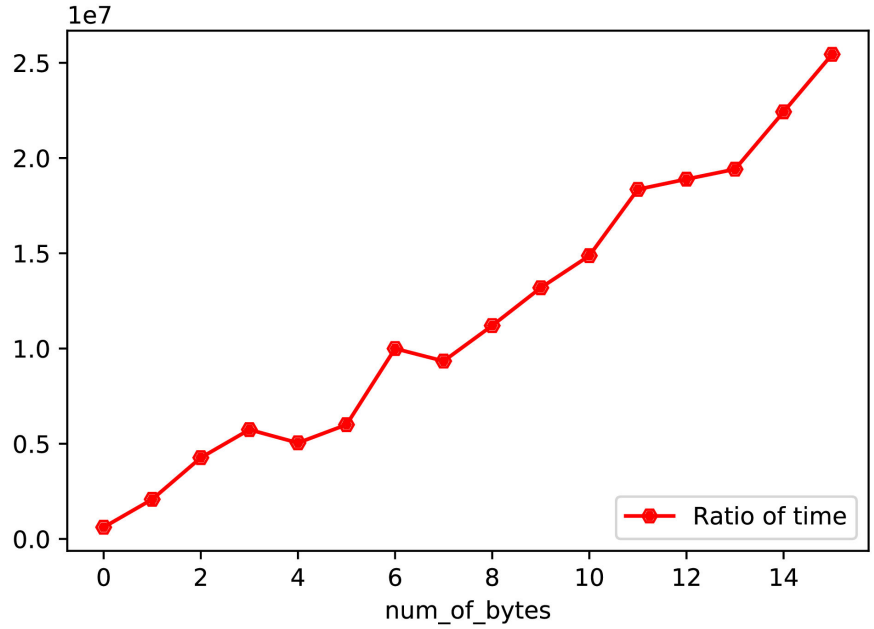


Figure 4: Ratio of time for PS-TH-GE and GEEA to reach the same accuracy.

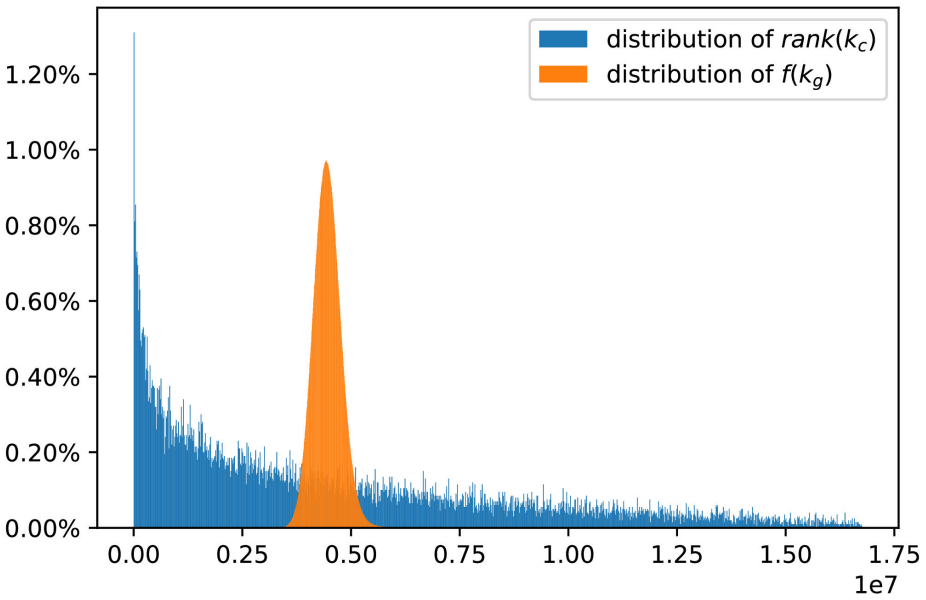


Figure 6: Distributions of $rank(k_c)$(blue) and $f(k_g)$(orange).

enumerate, thus giving us the exact distribution of $f(k_g)$. It is also large enough to start showing the highly skewed distribution of $rank(k_c)$ typical in multi-byte cases. We can see that the distribution of $f(k_g)$ is symmetric and bell shaped, while the distribution of $rank(k_c)$ is highly skewed. Thus to arrive at the same accuracy, PS-TH-GE would need even more than the seven order of magnitudes time indicated in Figure 4.

We now show the results of applying PS-TH-GE and GEEA for the template attack on the full 16-bytes AES key in the SGII-1M data set. Figure 7 plots PS-TH-GE and GEEA estimation, and corresponding 95% confidence intervals at several q values ($q = 5K, 10K, 30K, 50K, 80K$). For the faster GEEA, we averaged over $M = 6 \times 10^7$ $f(k_g)$s which takes about 1.8 hours per q values. The PS-TH-GE averages over $N = 2000$ $rank(k_c)$s which takes about five-fold computing time than GEEA. The narrow confidence intervals of GEEA demonstrate that they provide accurate useful GE estimation to evaluate the side-channel security of the device against adversaries with access to q measured traces. In contrast, PS-TH-GE only gives useful narrow confidence intervals at two ends of the range of q: when the attack always fails ($GE \approx 2^{112}$ when $q = 5000$) or always succeeds ($GE \approx 1$ when $q = 80,000$). For the cases with GE ranging from 2^{40} to 2^{100}, the confidence intervals of PS-TH-GE are too wide to be useful. Actually, the lower confidence bounds of PS-TH-GE are negative and set to 0 in log-scale shown in the figure. When $q = 30,000$, the 95% confidence interval of GEEA is $(2^{65.20}, 2^{65.54})$. In contrast, based on the estimated variance of $rank(k_c)$s, for PS-TH-GE to achieve a confidence interval of one-bit length(i.e. $log_2(upperbound) - log_2(lowerbound) \leqslant 1$), $N = 4.4 * 10^6$ samples are needed. This will require $2.1 * 10^4$ hours of computation, far beyond our computation limit.

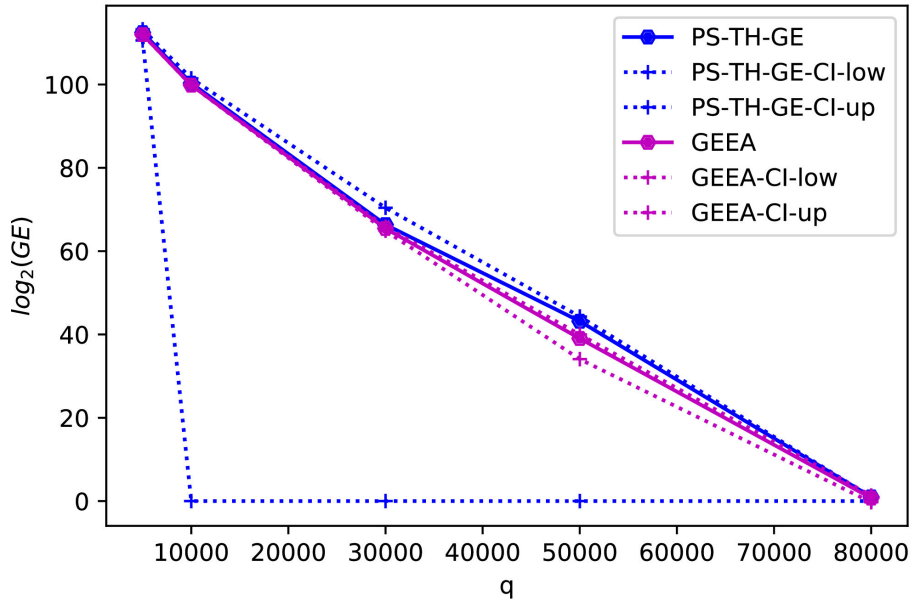


Figure 7: Comparison of PS-TH-GE and GEEA for full-key GE on the SGII-1M database.

4.3.1 Robustness of GEEA Estimator with respect to estimated $\hat{\mu}_{k_g}$ and $\hat{\sigma}^2_{k_g,k_g}$

As described in Algorithm 1, the computation of our GEEA estimator uses the estimated means $\hat{\mu}^m_{k_g}$ and the estimated variances $(\hat{\sigma}^m_{k_g,k_g})^2$, $m = 1, ..., B$. Hence, for reliable GEEA

estimator we want it to be robust to variations in $\hat{\mu}_{k_g}^m$ and $(\hat{\sigma}_{k_g,k_g}^m)^2$. For each byte, since we are estimating $255 + 255 = 510$ quantities using a number of traces $(300k)$ that is more than two orders of magnitude larger, we expect them to be estimated very accurately and the resulting GEEA estimator should be reliable.

To empirically verify the robustness of GEEA estimator, we simulate sets of score vectors of the same size as the validation data set (i.e. 300k) from multivariate Gaussian distributions of each subkey byte. Then we re-estimate means and variances of the score vectors on the simulated data sets, and calculate GEEA estimations accordingly. The resulting simulated GEEA estimations are very close to the original GEEA estimation on the SGII-1M data. Figure 8 plots five such simulated GEEA estimations together with the confidence bounds of original GEEA estimation reproduced from Figure 7. We can see that all simulated GEEA estimations fall well within the confidence bounds, indicating that robustness of GEEA estimator to the variations in re-estimated means $\hat{\mu}_{k_g}^m$ and variances $(\hat{\sigma}_{k_g,k_g}^m)^2$.

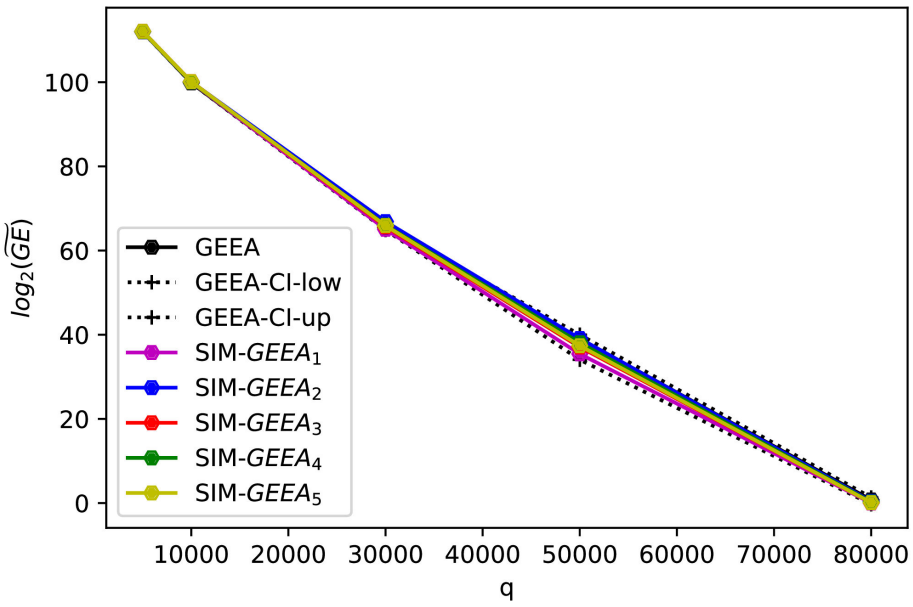


Figure 8: Comparison of Simulated GEEA estimations with confidence bounds of original GEEA estimation on the SGII-1M data.

5 Conclusions

GE is an important metric for evaluating SCA security commonly adopted in practice. However, it has only been widely used in evaluating attacks on single-byte subkeys, and reliable GE estimation is often unavailable for a long key such as the full 16-byte AES key. Since GE measures the average amount of computation that is required for a successful SCA, it is most relevant for full-key attacks. In this paper, we propose a novel approach of estimating GE, not through averaging ranks nor through posterior probabilities, but from theoretical pairwise success rates. This approach produces the first reliable GE estimation for the full AES-128 key. This enables accurate practical assessment of SCA security using GE.

Acknowledgments

This work was supported in part by National Science Foundation under grant SaTC-1563697. We are grateful to Dr. Annelie Heuser for her shepherding and helpful comments.

References

[BLvV15] 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.

[CP17] 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.

[DW17] Liron David and Avishai Wool. A bounded-space near-optimal key enumeration algorithm for multi-subkey side-channel attacks. In Helena Handschuh, editor, *Topics in Cryptology – CT-RSA 2017*, pages 311–327, Cham, 2017. Springer International Publishing.

[FDLZ15] Yunsi Fei, A. Adam Ding, Jian Lao, and Liwei Zhang. A statistics-based success rate model for dpa and cpa. *Journal of Cryptographic Engineering*, 5(4):227–243, Nov 2015.

[FLD12] Yunsi Fei, Qiasi Luo, and A. Adam Ding. A statistical model for dpa with novel algorithmic confusion analysis. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, pages 233–250, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[GGP+15] 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.

[Gro18] Vincent Grosso. Scalable key rank estimation (and key enumeration) algorithm for large keys. *IACR Cryptology ePrint Archive*, 2018:175, 2018.

[KJJ99] 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.

[LPR+14] Victor Lomné, Emmanuel Prouff, Matthieu Rivain, Thomas Roche, and Adrian Thillard. How to estimate the success rate of higher-order side-channel attacks. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems – CHES 2014*, pages 35–54, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[Mas94] J. L. Massey. Guessing and entropy. In *Proceedings of 1994 IEEE International Symposium on Information Theory*, pages 204–, June 1994.

[MMOS16] 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.

[MOOS15] Daniel P. Martin, Jonathan F. O'Connell, Elisabeth Oswald, and Martijn Stam. Counting keys in parallel after a side channel attack. In *Proceedings, Part II, of the 21st International Conference on Advances in Cryptology — ASIACRYPT 2015 - Volume 9453*, pages 313–337, Berlin, Heidelberg, 2015. Springer-Verlag.

[PSB+18] Emmanuel Prouff, Remi Strullu, Ryad Benadjila, Eleonora Cagli, and Cécile Dumas. Study of deep learning techniques for side-channel analysis and introduction to ASCAD database. *IACR Cryptology ePrint Archive*, 2018:53, 2018.

[Riv09] Matthieu Rivain. On the exact success rate of side channel analysis in the gaussian model. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography*, pages 165–183, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[SMY09] 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.

[VCGS13] 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.

A Appendix

In this section, we study when GM is an unbiased estimator of Guessing Entropy, for the simple case where there are only two key candidate k_g and k_c. For GM calculation, we need the posterior probabilities of the key given data, denoted as $\mathbb{P}_{post}(k|\mathcal{L},\mathcal{X})$. Notice that the posterior probability is proportional to the likelihood, and we consider the score $s(k)$ used by the adversary as the log-likelihood of each key given data. Assuming uniform prior distribution (no prior knowledge of key without side channel measurement), the posterior probability for a key value is $\mathbb{P}_{post}(k|\mathcal{L},\mathcal{X}) = \frac{e^{s(k)}}{e^{s(k_c)}+e^{s(k_g)}}$. We consider the case where the score vector follows a Gaussian distribution as commonly occurs with the additive distinguishers discussed above. For such a case, we have an explicit condition of GM to be unbiased for Guessing Entropy as stated in the following Theorem.

Theorem 2. *Given two key hypotheses k_g and k_c, assume that the scores $s(k_g)$ and $s(k_c)$ represent the log-likelihood of each key given data. Also we assume that the comparison score $\Delta_{k_g} = s(k_g) - s(k_c)$ follows the Gaussian distribution with mean μ_{k_g} and variance $\sigma^2_{k_g,k_g}$. Then $\mathbb{E}(GM) = GE$ if and only if*

$$\int_{t<0} \frac{e^t}{1+e^t}\phi_{\mu_{k_g},\sigma_{k_g,k_g}}(t)dt = \int_{t\geq 0} \frac{e^t}{1+e^t}\phi_{\mu_{k_g},\sigma_{k_g,k_g}}(t)dt,$$

where $\phi_{\mu_{k_g},\sigma_{k_g,k_g}}(t)$ denotes the probability density function of the Gaussian distribution with mean μ_{k_g} and variance $\sigma^2_{k_g,k_g}$.

Proof. For two keys only, the comparison vector only has one element $\Delta_{k_g} = s(k_g) - s(k_c)$ as in equation (4). As in equation (7), we have

$$GE = \mathbb{E}[rank(k_c)] = \mathbb{E}[1 + \mathbb{1}_{s(k_g)>s(k_c)}] \quad = 1 + \mathbb{E}[\mathbb{1}_{s(k_g)>s(k_c)}] = 1 + \mathbb{P}(\Delta_{k_g} \geq 0). \tag{15}$$

For the GM estimator, [CP17] used the i-th largest posterior probability $\mathbb{P}_{post}(k|\mathcal{L},\mathcal{X})$ as $\mathbb{P}[rank(k_c|\mathcal{L},\mathcal{X})=i]$. Since there are only two keys, GM considers $\mathbb{P}[rank(k_c|\mathcal{L},\mathcal{X})=1]$ the same as $max[\mathbb{P}_{post}(k_g|\mathcal{L},\mathcal{X}),\mathbb{P}_{post}(k_c|\mathcal{L},\mathcal{X})]$, and considers $\mathbb{P}[rank(k_c|\mathcal{L},\mathcal{X})=2]$ the same as $min[\mathbb{P}_{post}(k_g|\mathcal{L},\mathcal{X}),\mathbb{P}_{post}(k_c|\mathcal{L},\mathcal{X})]$. Since the posterior probability of k_g and k_c are respectively $\frac{e^{s(k_g)}}{e^{s(k_c)}+e^{s(k_g)}}$ and $\frac{e^{s(k_c)}}{e^{s(k_c)}+e^{s(k_g)}}$, we

$$\begin{aligned}
&\mathbb{E}(GM)\\
=\ &\mathbb{E}\{1*max[\mathbb{P}_{post}(k_g|\mathcal{L},\mathcal{X}),\mathbb{P}_{post}(k_c|\mathcal{L},\mathcal{X})]+2*min[\mathbb{P}_{post}(k_g|\mathcal{L},\mathcal{X}),\mathbb{P}_{post}(k_c|\mathcal{L},\mathcal{X})]\}\\
=\ &\mathbb{E}\{1*max[\tfrac{e^{s(k_c)}}{e^{s(k_c)}+e^{s(k_g)}},\tfrac{e^{s(k_g)}}{e^{s(k_c)}+e^{s(k_g)}}]+2*min[\tfrac{e^{s(k_c)}}{e^{s(k_c)}+e^{s(k_g)}},\tfrac{e^{s(k_g)}}{e^{s(k_c)}+e^{s(k_g)}}]\}\\
=\ &\mathbb{E}\{[\tfrac{e^{s(k_c)}}{e^{s(k_c)}+e^{s(k_g)}}+\tfrac{e^{s(k_g)}}{e^{s(k_c)}+e^{s(k_g)}}]+min[\tfrac{e^{s(k_c)}}{e^{s(k_c)}+e^{s(k_g)}},\tfrac{e^{s(k_g)}}{e^{s(k_c)}+e^{s(k_g)}}]\}\\
=\ &\mathbb{E}\{1+min[\tfrac{1}{1+e^{\Delta_{k_g}}},\tfrac{e^{\Delta_{k_g}}}{1+e^{\Delta_{k_g}}}]\}\\
=\ &1+\textstyle\int_{t\in\mathbb{R}} min[\tfrac{1}{1+e^t},\tfrac{e^t}{1+e^t}]*p_{\Delta_{k_g}}(t)dt\\
=\ &1+\textstyle\int_{t<0}\tfrac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt+\int_{t\geq 0}\tfrac{1}{1+e^t}*p_{\Delta_{k_g}}(t)dt\\
=\ &1+\textstyle\int_{t<0}\tfrac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt+\mathbb{P}(\Delta_{k_g}\geq 0)-\int_{t\geq 0}\tfrac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt.
\end{aligned}\tag{16}$$

Thus we have

$$GE-\mathbb{E}(GM)=\int_{t\geq 0}\frac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt-\int_{t<0}\frac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt.$$

Using the fact that Δ_{k_g} follows the Gaussian distribution with mean μ_{k_g} and variance $\sigma^2_{k_g,k_g}$, we have

$$\begin{aligned}
GE-\mathbb{E}(GM)\ &=\textstyle\int_{t\geq 0}\tfrac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt-\int_{t<0}\tfrac{e^t}{1+e^t}*p_{\Delta_{k_g}}(t)dt\\
&=\textstyle\int_{t\geq 0}\tfrac{e^t}{1+e^t}*\phi_{\mu_{k_g},\sigma_{k_g,k_g}}(t)dt-\int_{t<0}\tfrac{e^t}{1+e^t}*\phi_{\mu_{k_g},\sigma_{k_g,k_g}}(t)dt\\
&=-G(\mu_{k_g},\sigma_{k_g,k_g}).
\end{aligned}\tag{17}$$

where we denote the function

$$G(\mu,\sigma)=\int_{t<0}\frac{e^t}{1+e^t}\phi_{\mu,\sigma}(t)dt-\int_{t\geq 0}\frac{e^t}{1+e^t}\phi_{\mu,\sigma}(t)dt$$

From equation (17), $GE=\mathbb{E}(GM)$ if and only if $G(\mu_{k_g},\sigma_{k_g,k_g})=0$. □

Given Theorem 2, we can study the bias of GM versus GE. We plotted the function of $G(\mu,\sigma)$ versus the mean μ for several variance values $\sigma^2=1$, $\sigma^2=4$ and $\sigma^2=25$. As μ increases from $-\infty$ to ∞, $G(\mu,\sigma)$ starts from 0 and becomes increasingly negative before turns up and crosses zero at $\mu=-\sigma^2/2$, then it becomes increasingly positive and approaches 1 in as $\mu\to\infty$.

It is easy to understand the behavior when $\mu\geq 0$: in such cases, the SCA is using a wrong model which prefers the wrong key k_g over the correct key k_c, thus the average $rank(k_c)$ is approaching 2 as $\mu\to\infty$. However, as $\mu\to\infty$, the posterior probability increasingly concentrate on k_g as the adversarial becomes increasingly confident while selecting the wrong key, which results in GM estimator converging to 1. Using posterior probability as the ranking probability can not distinguish the 'wrongly' high confidence from the 'correctly' high confidence, and will provide an underestimate for GE.

Also, this graph provides an explanation on why an agreement was observed between $\mathbb{E}(GM)$ and GE on their simulations [CP17]. The posterior probability (likelihood) is an additive score distinguisher on the trace set $\mathcal{L}=\{l_1,...,l_q\}$ with $\Delta_{k_g}=s(k_g)-s(k_c)=\sum_{i=1}^{q}[s(k_g|l_i)-s(k_c|l_i)]$ where the $s(k|l_i)$ is the (log-likelihood) score based on only one trace l_i. Let μ_1 and σ_1^2 denote the mean and variance of $s(k_g|l_1)-s(k_c|l_1)$, we have $\mu_{k_g}=q\mu_1$ and $\sigma^2_{k_g,k_g}=q\sigma_1^2$. For the leakage simulated from Hamming weights plus additive Gaussian

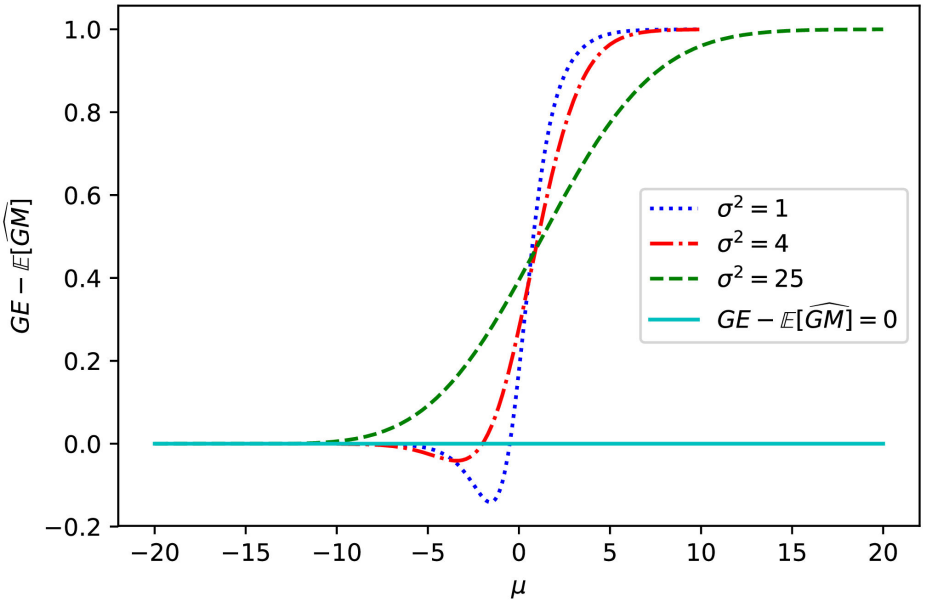


Figure 9: Plot of the GE-$\mathbb{E}(GM)$ versus mean μ for several variance values.

noise [CP17], we have $\mu_1 = -SNR^2\kappa(k_g, k_c)/2$ and $\sigma_1^2 = SNR^2\kappa(k_g, k_c)$ when ignoring higher order terms in the formulas [FDLZ15]. Here SNR is the Signal-Noise-Ratio and $\kappa(k_g, k_c)$ is the confusion coefficient as defined in prior work [FDLZ15]. This means that the condition $\mu_{k_g} = -\sigma^2_{k_g,k_g}/2$ is approximately satisfied for large SNR.

Also, both $\mu_{k_g} = q\mu_1$ and $\sigma^2_{k_g,k_g} = q\sigma_1^2$ are large for large values of q. Since the valley becomes very shallow for large σ_{k_g,k_g} in Figure 9, any overestimation of GE by $\mathbb{E}(GM)$ is generally not observable in numerical studies. Only the underestimation of GE by $\mathbb{E}(GM)$ may become obvious for the weak distinguisher when $\mu_{k_g} > -\sigma^2_{k_g,k_g}/2$, as occurred in the real data experiments of Choudary and Popescu's work [CP17] and here in Section 4 in the one byte key example. The simulation from a known leakage model with moderate to low noise level leads to easy correct key distinction, and may not show observable difference between $\mathbb{E}(GM)$ and GE since it falls into the left tail of the curves shown in the Figure 9. However, for practical certification purpose, the device may already be equipped with countermeasures and remaining leakage is weak. In such cases, the underestimation by $\mathbb{E}(GM)$ prevents an accurate assessment of SCA resistance.