Profiling Side-channel Analysis in the Efficient Attacker Framework

Stjepan Picek[1], Annelie Heuser[2], Guilherme Perin[1], and Sylvain Guilley[3]

[1] Delft University of Technology, The Netherlands
[2] Univ Rennes, Inria, CNRS, IRISA, France
[3] Secure-IC, France

Abstract. Profiling side-channel attacks represent the most powerful category of side-channel attacks. There, we assume that the attacker has access to a clone device to profile its leaking behavior. Additionally, we consider the attacker to be unbounded in power to give the worst-case security analysis. In this paper, we start with a different premise where we are interested in the minimum strength that the attacker requires to conduct a successful attack. To that end, we propose a new framework for profiling side-channel analysis that we call the Efficient Attacker Framework. With it, we require the attackers to use as powerful attacks as possible, but we also provide a setting that inherently allows a more objective analysis among attacks.
We discuss the ramifications of having the attacker with unlimited power when considering the neural network-based attacks. There, we show that the Universal Approximation Theorem can be connected with neural network-based attacks able to break implementations with only a single measurement. Those considerations further strengthen the need for the Efficient Attacker Framework. To confirm our theoretical results, we provide an experimental evaluation of our framework.

1 Introduction

Side-channel analysis (SCA) is a threat that exploits weaknesses in physical implementations of cryptographic algorithms rather than the algorithms themselves [1]. SCA takes advantage of any unintentional leakage observed in physical channels like timing [2], power dissipation [3,4], or electromagnetic (EM) radiation [5]. Profiling SCA performs the worst-case security analysis by considering the most powerful side-channel attacker with access to an open (the keys can be chosen or are known by the attacker) clone device.

Usually, we consider an attacker in the setting where he has unbounded power, e.g., he can obtain any number of profiling or attack traces, and he has unlimited computational power. In the last two decades, besides template attack and its variants [6,7], the SCA community started using machine learning to conduct profiling attacks. Those results proved to be highly competitive when compared to template attack, and, in many scenarios, machine learning methods surpassed template attack performance [8–10]. Unfortunately, in these scenarios, the experimental setup is often arbitrarily limited, and no clear guidelines on the limitation of profiling traces or the hyperparameter tuning phase are offered or discussed. More recently, the SCA community also started to experiment with deep learning where the performance of such methods bested both template attack and other machine learning methods [11, 12]. Again, no clear guidelines on the number of profiling traces or hyperparameter tuning were given or investigated. Consequently, there exists an evident lack of evaluation guidelines/frameworks in the context of profiled analysis to properly understand the performance of various attacks or how they compare.

In the machine learning domain, there is a well-known theorem called the Universal Approximation Theorem [13], which informally states that a feed-forward neural network with a single hidden layer containing a finite number of neurons can approximate a wide range of functions to any desired level of error. With such a theorem, and considering a powerful ("unbounded") side-channel attacker, we must assume he can approximate any function describing the implementation leakage. Since the theorem states that the approximation is made to any desired level of error, this would result in an attacker able to break any implementation that leaks exploitable side-channel information. As such setting may represent an ideal case from the attacker's perspective, our goal is to extend the currently used evaluation techniques to a framework that determines the least powerful attacker that is still able to reveal secret information.

To achieve this, we evaluate the limit on 1) the number of measurements the attacker can collect in the training phase and 2) the number of hyperparameter tuning experiments. It could sound counter-intuitive to make such limitations as one can argue there is no reason why an attacker cannot collect a large number of measurements or run hyperparameter tuning as long as needed (or select algorithm that has no hyperparameters to tune). We claim that there are several reasons for that:

1. By considering a scenario where an unlimited number of measurements are available, we "allow" less powerful attacks. More precisely, the attacker can use a larger set of measurements to compensate for less powerful profiling models.
2. By considering a scenario where a computationally unbounded attacker runs the analysis, we assume he can always find the best possible attack and that seldom happens in practice.
3. The device may include a countermeasure that limits the number of exploitable measurements. The experimental setup can have constraints that limit the allowed length of the hyperparameter tuning phase.
4. Although taking measurements or running more experiments is "cheap", there is always a point where this is more effort than the target/secret is worth.
5. Having more measurements does not guarantee better results, especially in realistic scenarios. Consider the case where one device is used for profiling and the other for attacking (a realistic case that is usually simplified in research works where only a single device is used). Then, adding more measurements to the profiling phase can cause machine learning methods to

overfit [14]. The same issue can happen due to a too long tuning phase. Overfitting occurs when the learning model learns the data too well and cannot adapt to previously unseen data. We illustrate this with a simple example of two AVR Atmega328p 8-bit micro-controllers. There are two devices, where each has a different key. In Figure 1, we depict the results with 10 000, and 40 000 measurements in the training phase. We see that the guessing entropy (i.e., the average key rank of the secret key) is much better for 10 000 measurements as less overfitting occurs. With 40 000 measurements, the machine learning method overfitted on train data and could not adapt to measurements coming from a different device. We emphasize that we do not claim this behavior must occur anytime different devices are used for profiling and attacking. Still, there will be overfitting (even if using only a single device), and additional measurements or hyperparameter tuning may increase this effect.

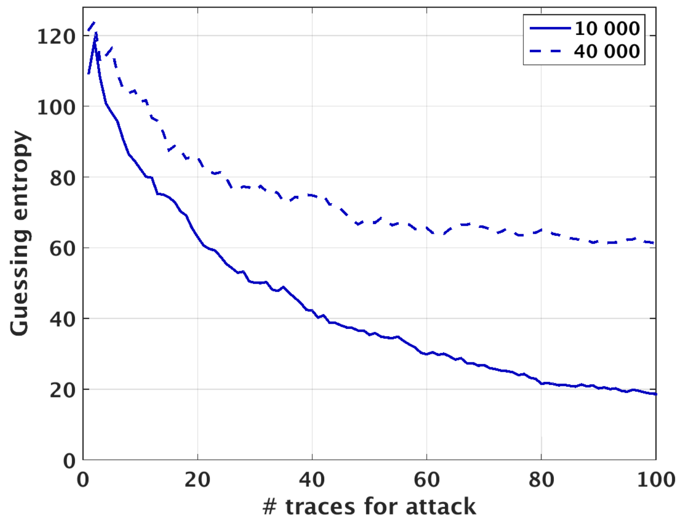


Fig. 1: Multilayer perceptron results for two device settings.

In realistic profiling scenarios, training and testing data do not come from the same distribution (as they come from two devices), so it is clear that the learning model does not learn on the data that it needs to predict, which makes this a problem difficult to circumvent. We refer interested readers to related works for further details on this problem and possible solutions [14, 15].

As far as we know, there are no previous works considering profiled and realistic attacker evaluation frameworks. When the attacker is restricted, it is usually set as one of several tested scenarios (e.g., testing the performance of a classifier with specific hyperparameters or a different number of measurements in the training phase). Alternatively, it is motivated with some limitations in the data acquisition or evaluation process.

In this paper, we present the following main contributions:

1. We propose a new framework for profiling side-channel analysis where we evaluate the minimum strength of an attacker in the profiling phase to still be successful in the testing phase. As such, we also introduce a new threat model that differs from a common one by considering a more realistic attacker. Note, the attacker in our threat model is still powerful from the computational perspective as well as from the perspective of the learning models he can build. In other words, we move from the problem of simply breaking the target (which is well-explored and with strong results, especially when considering machine learning) to a problem where we aim the break the target with a minimal number of measurements and minimal hyperparameter tuning. We consider our framework to be intuitive and easily adaptable to a plethora of realistic scenarios.
2. With some constraints on the type of SCA leakage, we argue that a neural network can break the cryptographic implementation with a single measurement.
3. We strengthen our theoretical results with an experimental evaluation conducted on publicly available datasets protected with masking countermeasures. We explore two commonly used leakage models and two neural network types.

The rest of this paper is organized as follows. In Section 2, we discuss the common methods to conduct profiling SCA. In Section 3, we introduce the currently used frameworks for profiling SCA. Section 4 introduces our new framework – The Efficient Attacker Framework. In Section 5, we further develop on the proposed framework, and we show its relevance for neural network-based approaches. Additionally, by connecting the theoretical results from the machine learning domain and SCA, we show how neural networks can be even more powerful than one would intuitively assume. In Section 6, we discuss the significance of our findings and how the Efficient Attacker Framework fits into scenarios outside of side-channel analysis. Finally, in Section 7, we conclude the paper.

2 On the Methods to Perform Profiling Side-channel Analysis

In this section, we start by introducing the notation we follow. Next, we discuss profiling side-channel analysis and related works. Finally, we briefly introduce multilayer perceptron and convolutional neural networks as the architectures of choice in our experiments.

2.1 Notation

Let calligraphic letters ($\mathcal{X}$) denote sets, capital letters (X) denote random variables over $\mathcal{X}$, and the corresponding lowercase letters (x) denote their realizations. Let k^* be the fixed secret cryptographic key (byte), k any possible key hypothesis, and the random variable T the plaintext or ciphertext of the cryptographic algorithm that is uniformly chosen.

2.2 Profiling Side-channel Attacks

A powerful attacker has a device (usually called the clone device) with knowledge about the secret key implemented where he can obtain a set of N profiling traces $X_1, \ldots, X_N$. Using the known secret key and N plaintexts or ciphertexts T_p, he calculates a leakage model $Y(T_p, k^*)$. In this phase, commonly known as the profiling phase, the attacker has available N pairs (X_i, Y_i) with $i = 1, \ldots, N$, which are used to build a profiling model f. The attack can then be carried out on another device by using the mapping f. For this, the attacker measures an additional Q traces $X_1, \ldots, X_Q$ from the device under attack to guess the unknown secret key k_a^*. The leakage model is now calculated for all possible key candidates $k \in \mathcal{K}$:

$$Y(T_a, k_1), \ldots, Y(T_a, k_{|\mathcal{K}|}), \tag{1}$$

given Q plaintexts or ciphertexts T_a.

In Figure 2, we depict the profiling side-channel attack scenario where we distinguish between the profiling phase where a machine learning model f is fitted to N measurements and the attack phase where we use the machine learning model f and Q measurements to predict the secret key on the attacking device. Throughout the paper, we use the terms of the learning model, machine learning model, and profiling model interchangeably.

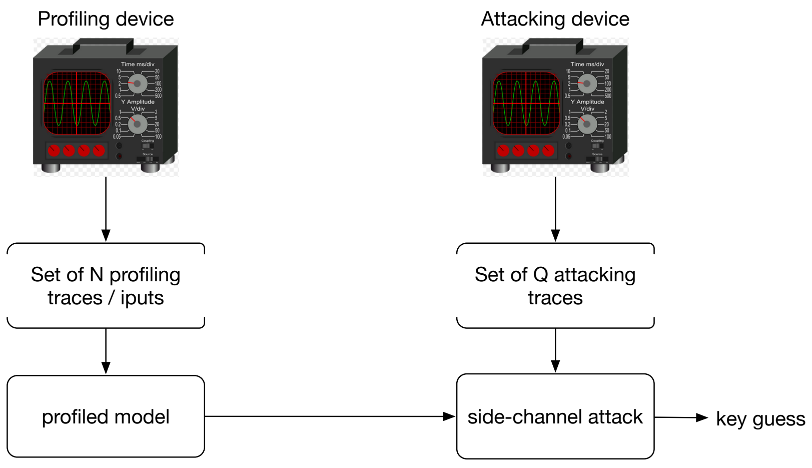


Fig. 2: The profiling side-channel analysis.

The best-known profiling attack is the template attack [16], which is based on the Bayesian rule. It works under the assumption that the measurements are dependent on the D features given the target class. More precisely, given the vector of N observed attribute values for x, the posterior probability for each class value y is computed as:

$$p(Y=y|X=x) = \frac{p(Y=y)p(X=x|Y=y)}{p(X=x)}, \tag{2}$$

where $X=x$ represents the event that $X_1 = x_1 \wedge X_2 = x2 \wedge \ldots \wedge X_N = x_N$.

Note that the class variable Y and the measurement X are not of the same type: Y is discrete while X is continuous. So, the discrete probability $p(Y=y)$ is equal to its sample frequency where $p(X=x|Y=y)$ displays a density function. Mostly in the state-of-the art, $p(X=x|Y=y)$ is assumed to rely on a (multivariate) normal distribution and is thus parameterized by its mean $\bar{x}_y$ and covariance matrix Σ_y:

$$p(X=x|Y=y) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_y|}} e^{-\frac{1}{2}(x-\bar{x}_y)^T \Sigma_y^{-1}(x-\bar{x}_y)}. \tag{3}$$

In practice, the estimation of the covariance matrices for each class value of y can be ill-posed mainly due to insufficient traces for each class. The authors of [7] propose to use only one pooled covariance matrix to cope with statistical difficulties and thus lower efficiency. Accordingly, Eq. (3) changes to:

$$p(X=x|Y=y) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} e^{-\frac{1}{2}(x-\bar{x}_y)^T \Sigma^{-1}(x-\bar{x}_y)}. \tag{4}$$

The works in e.g., [7, 17] showed that the pooled version can be more efficient, in particular for a smaller number of traces in the profiling phase.

Profiling side-channel attacks, especially those based on machine learning, received significant attention in the SCA community in the last decade. There, researchers reported various scenarios where machine learning can achieve top results in attacking cryptographic implementations. In the last few years, deep learning emerged as a powerful alternative where results surpassed template attack and other machine learning methods.

When considering profiling (or supervised) SCA, the SCA community started with a template attack and its variants. Afterward, Support Vector Machines (SVM) [8, 18, 19] and Random Forest (RF) [20, 21] attracted most of the attention. In the last few years, deep learning, and more precisely, multilayer perceptron (MLP) [20, 22–24] and convolutional neural networks (CNNs) [10, 11, 22, 23] are taking the lead in profiled SCAs. The reason for the high popularity of deep learning can be found in the facts that it performs well 1) even without preprocessing (e.g., no need for feature selection) and 2) in the presence of countermeasures (CNNs work well with countermeasures occurring in the time domain (like random delay interrupts) due to their spatial invariance. At the same time, MLP combines features to produce a similar effect as higher-order attacks, thus thwarting masking countermeasures). Interestingly, most of the related works consider only scenarios to evaluate the influence of the number of testing traces in the performance. When there are experiments with different training set sizes, they do not aim to find the minimum training and testing traces for a certain level of performance. For the hyperparameter tuning, commonly, the authors denote the tested ranges and the best-obtained values. Still, it is far from apparent how much computational effort was needed or how sensitive are the algorithms to tuning.

Multilayer Perceptron The multilayer perceptron (MLP) is a feed-forward neural network that maps sets of inputs onto sets of appropriate outputs. MLP consists of multiple layers of nodes in a directed graph, where each layer is fully connected to the next one. The backpropagation algorithm is used to train the neural network [25]. An MLP consists of three or more layers (since input and output represent two layers) of nonlinearly-activating nodes [26].

Convolutional Neural Networks Convolutional neural networks (CNNs) commonly consist of three types of layers: convolutional layers, pooling layers, and fully-connected layers. Convolution layer computes the output of neurons connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. Pooling decrease the number of extracted features by performing a down-sampling operation along the spatial dimensions. The fully-connected layer (the same as in MLP) computes either the hidden activations or the class scores.

3 Existing Frameworks for Side-channel Evaluation

In this section, we discuss the currently used frameworks for profiling side-channel analysis in scientific works and certification.

3.1 Scientific Metrics

The most common evaluation metrics in the side-channel analysis are success rate (SR) and guessing entropy (GE) [27]. GE states the average number of key candidates an adversary needs to test to reveal the secret key after conducting a side-channel analysis. In particular, given Q amount of samples in the attack phase, an attack outputs a key guessing vector $g = [g_1, g_2, \dots, g_{|\mathcal{K}|}]$ in decreasing order of probability with $|\mathcal{K}|$ being the size of the keyspace. So, g_1 is the most likely and $g_{|\mathcal{K}|}$ the least likely key candidate. The guessing entropy is the average position of k_a^* in g over multiple experiments. The success rate is defined as the average empirical probability that g_1 equals the secret key k_a^*.

In practice, one may consider leakage models $Y(\cdot)$ that are bijective functions. Thus, each output probability calculated from the classifiers for $Y(k)$ directly relates to one key candidate k. When $Y(\cdot)$ is not bijective, several key candidates k may get assigned with the same output probabilities, which is why on average, a single trace attack ($Q = 1$) may not be possible in case of non-bijective leakage models. Further, to calculate the key guessing vector g over Q amount of samples, the (log-)likelihood principle is used.

Remark 1. SR and GE are used for practical evaluations in both non-profiling and profiling scenarios. Typically, they are given over a range of traces used in the attack phase (i.e., for $q = 1, 2, \ldots, Q$). In case these metrics are used in profiling scenarios, there are no clear guidelines on how to evaluate attacks. Most of the time, the number of training samples N in the profiling stage is (arbitrary) fixed, making comparisons and meaningful conclusions on profiling side-channel attacks or resistance of implementations hard and unreliable in most scenarios.

Whitnall and Oswald introduced a more theoretical framework that aims at comparing distinguishing powers instead of estimators of attacks [28,29]. Accordingly, the size of the profiling dataset N does not play any role in this framework. The most popular metrics of the framework are the relative and absolute distinguishing margins in which the output score of the correct key and the value for the highest-ranked alternative are compared.

Another approach to compare side-channel attacks uses closed-form expressions of distinguishers [30], which enables conclusions about distinguishers without the requirement of actual measurements. Unfortunately, only a few closed-form expressions of distinguishers have been achieved so far.

In the case of masking countermeasures, Duc et al. defined information-theoretical bounds on the success rate depending on the number of measurements, shares, and independent on the concrete estimated side-channel attack [31]. In [32], the authors provide information-theoretic tools to bound the model errors in side-channel evaluations concerning the choice of the leakage model.

Typically, to assess the performance of the machine learning classifiers, accuracy is used [33]. A detailed comparison between accuracy (but also other machine learning metrics like precision, recall, F1) and guessing entropy/success rate is given in [10], which details that such metrics may not always be a proper choice for side-channel analysis.

3.2 Practical Evaluation Testing

While most of these previous metrics are relevant in some contexts and scenarios, a different approach is required to make research statements in the context of profiling attacks. This issue becomes even more evident when looking at practical evaluation used in standardization processes. In practice, there are two main practical schemes:

1. Test-based schemes, such as NIST FIPS 140 [34] and its application to the mitigation of other attacks (part of Appendix F, in particular non-invasive attacks ISO/IEC 17825 [35]).
2. Evaluation-based schemes, such as Common Criteria (CC, ISO/IEC 15408 [36]).

Interestingly, both FIPS 140 and CC pay attention to the limited amount of resources spent. When considering FIPS 140 / ISO/IEC 17825, the requirement is more on the attack traces, but regarding CC, the evaluation of attacks is considered under two phases: identification (which matches with the training phase in the context of profiling side-channel attacks) and exploitation (which matches with the attack phase in the context of profiling side-channel attacks).

Strictly speaking, the distinction is for CC version 2, but it still implicitly holds for version 3. Several factors are considered for the quotation of attacks, namely: elapsed time, expertise, knowledge of the Target Of Evaluation (TOE), access to TOE, equipment, open samples. The first factor, elapsed time, has a direct connection with the acquisition of traces in the profiling phase and the hyperparameter tuning. Indeed, according to the guidance "Application of Attack Potential to Smartcards" [37]), the score is considered:

- 0 if the profiling of the traces can be performed in less than one hour,
- 1 if the profiling of the traces can be performed in less than one day,
- 2 if the profiling of the traces can be performed in less than one week,
- 3 if the profiling of the traces can be performed in less than one month,
- 5 if the profiling of the traces cannot be performed in less than one month.

Accordingly, we see that the CC guidance favors attacks, which are realized with as little profiling effort as possible. This profiling effort can go into the direction of the number of required measurements, the number of experiments in the hyperparameter tuning phase, or both.

3.3 Practical Observations and Effects of Aging

Besides overfitting (see details in the introduction and Figure 1), another difficulty for profiling attacks is that the collection of side-channel traces becomes less reliable after a long period. Indeed, some trend noise must be added to the side-channel traces (due to temperature and environmental conditions evolution over time). This has, for instance, been characterized by Heuser et al. in [38], where it is proven that trend noise drastically impedes SCA. Similar findings are confirmed by Cao et al. [39]. Very practical distinguishing situations, such as that depicted in Figure 3 shows that the best number of traces to estimate a distinguisher is not always "the maximal". This is illustrated on a simple "difference of means" attack representing side-channel attack on DPA contest 4.2 traces.

4 The Efficient Attacker Framework

In this section, we first introduce the threat model we follow. Next, we discuss the core assumptions for our framework as well as the components of a successful attack. Finally, we give a formal description of the Efficient Attacker Framework.

4.1 Threat Model

The adversary has access to a clone device running the target cryptographic algorithm. This device can be queried with a known key and plaintext, while corresponding leakage measurement is stored. Commonly, the adversary can have infinite queries to characterize a precise profiling model. There are no limits on how many experiments he can do to find such a profiling model. In our threat model, the adversary has a limited number of queries to characterize a profiling

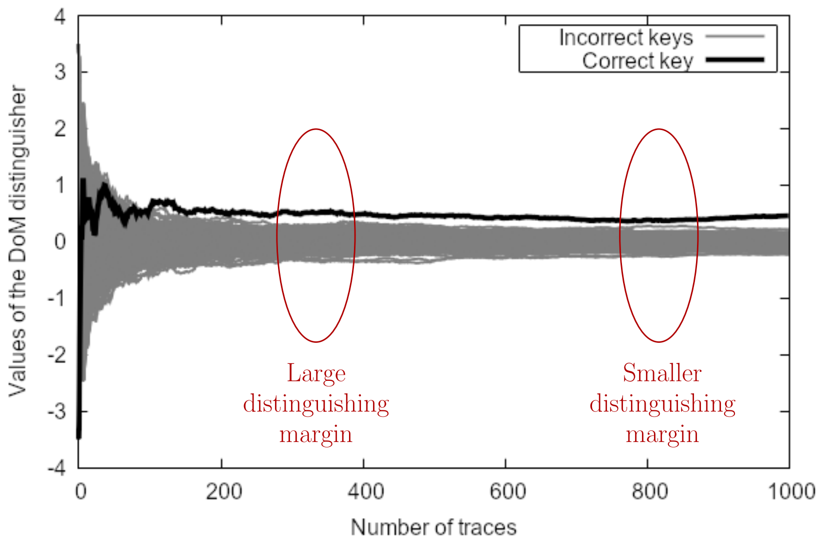


Fig. 3: Difference of Means (DoM) distinguisher estimation for all key bytes (the correct one and all incorrect ones).

model. Additionally, he has a limited number of experiments to conduct hyper-parameter tuning. Next, the adversary queries the attack device with known plaintext to obtain the unknown key. The corresponding side-channel leakage measurement is compared to the characterized profiling model to recover the key. Note, while our framework allows various machine learning tasks, in this paper, we concentrate on the classification task, as it is common in the profiling SCA.

4.2 Components of a Successful Attack

Current evaluations for profiling SCA mostly assume that the attacker is unbounded in his power. This assumption aims to provide the worst-case scenario for the designer, which should help in the proper assessment of the risk. Although the attacker is considered unbounded, he is always bounded, with bounds set ad-hoc, and there are no clear directions one should follow when modeling the realistic attacker.

First, we discuss two core assumptions we make in this research. These need to be fulfilled so that general meaningful comparisons between profiled attacks can be made, and our framework can provide exploitable results:

1. Attack must be possible. While our framework does not require the attacker to be always successful, the attack must be possible. For instance, having measurements completely uncorrelated with the labels will make our framework not useful. Still, if there is no connection between the measurements

and labels, no side-channel attack can succeed. Consequently, this is not a drawback of our framework.

2. We consider only profiling (supervised) attacks, and therefore, profiling measurements need to allow learnability about the problem. Profiling measurements that are completely uncorrelated with the attack measurements would make our framework not usable. The hyperparameter tuning (if possible) must allow reaching the useful profiling model. Again, the profiling attacks cannot work if the previous conditions are not fulfilled, so this does not represent a disadvantage of our framework.

Next, we examine the three components of a successful attack. The worst-case (strongest) attacker will be unbounded in all three components. At the same time, fulfilling only one or two of them accounts for more realistic settings one encounters in practice:

1. Quantity - there must be sufficient measurements in the profiling/testing phase to conduct the attack, i.e., to build a reliable profiling model that generalizes to the unseen data. This criterion is a natural one and already well-known in SCA as researchers usually report the performance of the attack concerning a different number of measurements. At the same time, there is much less research to determine the minimum number of measurements for a successful attack.
2. Quality - the measurements need to be of sufficient quality to conduct the attack. This condition could be translated into the requirement that the SNR should be sufficiently high, or that the data need to have all information required to model the leakage correctly. Finally, this component includes the leakage model's quality, i.e., that the considered leakage model provides sufficient information and the distribution of leakages. Again, like the previous component, this one is well addressed in the SCA community as researchers usually conduct various pre-processing steps, e.g., to select/transform features or align traces.
3. Learnability - the attacker needs to be able to learn the profiling model. This perspective also accounts for finding the best possible hyperparameters for the profiling model. The learnability is naturally connected with the quantity and quality parameters. This component is significantly less addressed in related work. While the researchers usually conduct various tuning procedures, they rarely report how difficult it was to find the hyperparameters used in the end.

We should not limit the quality component: if the attacker can obtain measurements, those measurements should be of the best possible quality. When discussing the quantity and learnability components, we can (and we must) evaluate the limit of the number of profiling measurements and experiments in the tuning phase since:

1. If always considering the extreme case of unbounded measurements in the profiling phase, we "allow" to utilize weaker attack, which may only work in this extreme scenario. On the other hand, if we consider the minimum

number of traces that are available in the profiling phase while still being able to succeed in the attack phase, we promote efficient attacks.

2. The attacker who is unbounded in his capabilities could break cryptographic implementations even with a single measurement (under certain assumptions) as he can always find the optimal attack. This reasoning suggests that ultimately, there is nothing the designer can do to stop the attack.

Remark 2. Having a limited number of measurements or time to conduct hyperparameter tuning is a realistic occurrence in practical scenarios, as the attacker may be limited by time, resources, and also face implemented countermeasures, which prevent him from taking an arbitrarily large amount of side-channel measurements, while knowing the secret key of the device.

To conclude, we need to consider an attacker who can perform a successful attack with the smallest possible number of measurements N, where success is defined over a performance metric ρ with a threshold of δ. To reach that success, the attacker should use the smallest possible number of tuning experiments h.

Example 1. Consider ρ being the guessing entropy < 20, which is a common threshold value in the side-channel analysis, see, e.g., [10]. Then, the measure of the attacker's power is 1) the number of profiling traces he needs to train a profiling model, which is then used on attack traces to break the implementation, 2) the number of experiments conducted before finding the hyperparameters resulting in a strong attack, or 3) both the number of profiling traces and hyperparameter tuning experiments.

4.3 Framework Description

Recall, the goal for machine learning is to learn a mapping (model) f from $\mathcal{X}$ to $\mathcal{Y}$, i.e., $Y \leftarrow f(X, \theta)$ where X are samples drawn i.i.d. from set $\mathcal{X}$ and where the cardinality of X equals N. Let $theta$ be the parameters of the profiling model that result in the best possible approximation (and for which we needed to select one of h hyperparameter combinations). X_p is the input to the profiling model (measurements), Y_a are labels associated with X_a, and $c(\theta, X_a, Y_a)$ is the cost used to train the profiling model. Additionally, let $\mathbf{g}_{Q,f} = [g_1, g_2, \ldots, g_{|\mathcal{K}|}]$ be the guessing vector from the profiling side-channel attack using Q traces in the attack phase, and the profiling model f built in the profiling phase as an input. Then, $\rho(g_Q, k_a^*)$ represents the performance metric of the profiling side-channel attack using the secret key k_a^* to evaluate the success.

The Efficient Attacker Framework aims at minimizing the number of profiling traces N and hyperparameter tuning experiments h to model the function f, such that the performance metric is still below (or above) a certain threshold δ:

$$\min\{N, h \ : \ \rho(g_{Q,f}, k_a^*) < \delta\}, where\ N, h \geq 1. \tag{5}$$

Algorithm 1 gives the pseudocode of the evaluation in the Efficient Attacker Framework, and a motivating example is given in Example 2. Note that the framework allows conducting experiments in parallel to the data acquisition phase. Indeed, one can start with evaluating the performance regardless of the number of already acquired measurements. For example, the attacker can assume the regime where he downloads the new measurements every hour and repeats the experiments with an always-increasing number of measurements.

```
   Input  : Profiling and attacking device to collect traces from
   Output : Minimum number of profiling traces $N$, the minimum number of
            hyperparameter experiments $h$
 1 Capture a testing dataset (with secret key $k_a^*$). Its size $Q$ depends on the
     expected performance of the attack. For instance, this test dataset can be as
     small as one trace!
 2 Select a performance metric $\rho$ and a threshold value $\delta$, e.g., GE $< 20$
 3 Training_set $\leftarrow \varnothing$
 4 while True do
 5     Capture one trace        // A speed-up can be obtained by advancing
         faster, e.g., 10 by 10 traces
 6     Append them to Training_set, $N = N + 1$
 7     Perform Training with specific hyperparameters (which yields a model $f$)
 8     Make a key guess $k$ from the Testing_set with $Q$ measurements
 9     if $\rho < \delta$ then
10         break                                   // model is good enough
11 return Minimum number of profiling traces $N$, minimum number of
     hyperparameter experiments $h$
```

Algorithm 1: The evaluation procedure in the Efficient Attacker Framework.

Remark 3. The Algorithm 1 considers both the number of profiling traces and hyperparameter tuning experiments, but this can be easily adjusted for only one of those options. For instance, if using a template attack, there are no hyperparameters to tune, which means that only the number of profiling traces is relevant. On the other hand, if facing a setting where one cannot obtain enough measurements to reach δ, then the natural choice is not to limit the number of measurements even more, but to consider the number of hyperparameter tuning experiments. While we consider the number of hyperparameter tuning experiments for the learnability perspective in this paper, this could be easily cast, for instance, to the selection of points of interest with template attack.

Example 2. A standard performance metric used in the side-channel analysis is guessing entropy with, e.g., a threshold $\delta = 20$. Therefore, in the Efficient Attacker Framework, one would find the minimum number of profiling traces N and hyperparameter experiments h to reach a guessing entropy below 20 for a fixed number of Q attack traces. This setting ensures that key enumeration algorithms [40] (when attacking not one but several key bytes, as in the AES-128 where there are 16 bytes of the key that needs to be recovered simultaneously for a full key recovery attack) are efficient. Typically, Q is ranging over a set of values. Experimental results are discussed in Section 6.

Remark 4. In practice, Algorithm 1 shall be evaluated several times to get an empirical estimation $\hat{\mathbb{E}}(N, h)$ of the minimum number of profiling traces/hyperparameter tuning experiments. This can be achieved by averaging several evaluations of Algorithm 1 (as done in non-profiled side-channel attack-oriented frameworks, see [27, §3.1]).

Remark 5. The Efficient Attacker Framework is attack-oriented and aims at unleashing profiled attacks even with frugal learning constraints. This reflects some situations where the number of interactions with the device is limited:
- by design, e.g., owing to enforcement of countermeasures such as limited number of cryptographic executions until system end-of-life, or
- by certification constraints such as limited "elapsed time" in the Common Evaluation Methodology (CEM [41, B.4.2.2]) of the Common Criteria.

Remark 6. If two profiling models exhibit very similar performance but require a radically different amount of resources, then a Pareto front of solutions (i.e., a set of non-dominated solutions) needs to be given where the designer can decide on a proper trade-off.

We reiterate that our framework is not designed to force the attacker to use a small number of measurements in the profiling phase nor to limit the number of experiments in the hyperparameter tuning phase. Instead, it forces the attacker to find the smallest number of traces and tuning experiments to conduct a successful attack.

5 The Efficient Attacker Framework in Neural Network-based Side-channel Attacks

In this section, we start by discussing the Universal Approximation theorem. Afterward, we connect the Universal Approximation Theorem and the optimal side-channel attack.

5.1 Universal Approximation Theorem

The Universal Approximation Theorem proves that for any Borel measurable function f (where Borel measurable mapping $f : \mathcal{X} \to \mathcal{Y}$ between two topological spaces has the property that $f^{-1}(\mathcal{A})$ is a Borel set[4] for any open set $\mathcal{A}$), there

[4] A Borel set is any set in a topological space that can be formed from open sets (a set S is open if every point in S has a neighborhood lying in the set) through the operations of countable union, countable intersection, and relative complement.

exists a feed-forward neural network, having only a single hidden layer with a finite number of neurons, which uniformly approximates f within an arbitrary nonzero amount of error ϵ [13, 42].

For this theorem to hold, we require only mild assumptions on activation functions (such that they saturate for both very negative and very positive arguments), and naturally, the network needs to have enough hidden units. Note, the theorem was also proved for a broader class of activation functions, including rectified linear unit [43].

We know that a multilayer perceptron with enough nodes can represent any Borel measurable function as a consequence of the Universal Approximation Theorem. Naturally, there is no guarantee that the machine learning algorithm will be able to learn such a function. Indeed, if this theorem is correct, the question is why it is still difficult (in many practical applications) to obtain a decent performance of a classifier, let alone approximation to an arbitrary ϵ. For instance, numerous works used a multilayer perceptron (which is a feed-forward network), where more than a single hidden layer is used, and yet, the results are far from optimal.

The main problem is that the Universal Approximation Theorem does not consider the algorithmic learnability of feed-forward networks. The theorem says that the number of nodes in the network is finite, but does not specify that number. There are some results on bounds on the number of nodes, but unfortunately, in the worst-case scenario, an exponential number of nodes is needed [44]. Additionally, from a practical perspective, the learnability of the profiling model also heavily depends on the quality and quantity of data at disposal. Recall, by quality, we consider that the data need to have all the information needed to model the function f correctly. With the notion of quantity, we assume to have sufficient information to build a reliable profiling model that will generalize to the unseen data.

Up to now, we mentioned only a multilayer perceptron and how it fits the Universal Approximation Theorem. At the same time, we stated in Section 1 that convolutional neural networks were recently used to achieve state-of-the-art performance in the SCA domain. Consequently, the natural question is to ask whether the Universal Approximation Theorem is also valid for convolutional neural networks. We give a small example.

Example 3. Let us consider a feed-forward network with a single hidden layer with A inputs and B outputs. To realize such an architecture, we require a weight matrix $W \in \mathbb{R}^{B \times A}$. If we assume that the convolution is applied only to the input and there is no padding, it is rather straightforward to see that we can simulate this feed-forward network with only two convolutional layers. In the first layer, we have $B \times A$ filters of shape A. The element a of filter b, a is equal to $W_{b,a}$ with the rest being zeros. This layer transforms the input into a BA-dimensional intermediate space where every dimension represents a product of weight and its corresponding input. The second layer contains B filters of shape BA. The elements $bA \ldots (b+1)A$ of filter b are ones while the rest are zeros. This layer performs the summation of products from the previous layer. Naturally, for this construction, we assumed some conditions that are not realistic, but we show this as a motivating example that the Universal Approximation Theorem can be applied for other types of neural networks.

We emphasize that various functions can be more efficiently approximated by architectures with greater depth, which is why deep learning can exhibit strong performance. More formally, D. Yarotsky showed that any translation equivariant function can be approximated arbitrarily well by a convolutional neural network given that it is sufficiently wide [45]. This result has a direct analogy to the Universal Approximation Theorem.

5.2 From the Universal Approximation Theorem to the Optimal Side-channel Attack

Here, we present results that connect the Universal Approximation Theorem and side-channel attacks with optimal performance, i.e., those where the attacker needs only a single measurement to break the implementation.

Conjecture 1. A side-channel leakage can be modeled by Borel measurable function.

Recall, Borel measurable function is a mapping $f : \mathcal{X} \to \mathcal{Y}$ between two topological spaces with the property that $f^{-1}(\mathcal{A})$ is a Borel set. A Borel set is any set in a topological space that can be formed from open sets (a set $\mathcal{S}$ is open if every point in $\mathcal{S}$ has a neighborhood lying in the set) through the operations of countable union, countable intersection, and relative complement for any open set $\mathcal{A}$.

Clearly, all continuous functions (i.e., functions defined on $\mathbb{R}$) are Borel functions (but not all Borel functions are continuous). Unfortunately, in SCA, one uses an oscilloscope in the acquisition process, and they have a finite precision, which makes the resulting function a discrete one.

Let us consider power or electromagnetic side-channel. As mentioned, the oscilloscope samples only a discrete-time and quantifies the measurements. Such measurements are a series of finite values, which may not be Borel measurable as such. However, before sampling and quantization, the signal was a physical quantity, which is Borel measurable. Indeed, it is obtained from the RLC-filtering of some physical quantity [46, Figure 2], itself obtained as the resolution of differential equations of electronics/mechanisms. It is, therefore, possible, as a pre-processing step, to interpolate and smooth the SCA measurements to make them continuous, hence eligible to be Borel measurable. More intuitively, there are infinitely many continuous functions that can describe a finite number of samples. Additionally, we can make use of Lusin's theorem, which states that every measurable function is continuous on nearly all its domain [47]. More formally, a function $f : \mathcal{X} \to \mathbb{R}$ is measurable if for every real number a, the set $x \in \mathcal{X} : f(x) > a$ is measurable. Practically, this means that any function that can be described is measurable.

Lemma 1. *If a side-channel leakage is Borel measurable (see Conjecture 1)), then a feed-forward neural network with a single hidden layer consisting of a finite number of neurons can approximate any side-channel leakage to a desired nonzero error.*

Proof. Straightforward from the Universal Approximation Theorem.

Lemma 2. *A profiling side-channel attack where the Universal Approximation Theorem holds (i.e., where Lemma 1 holds), can succeed in breaking an implementation with only a single measurement.*

Proof. Trivial. If SCA leakage can be approximated to a desired (nonzero) level of error, it means (provided that we use the appropriate leakage model) that we need only a single measurement to obtain the key information.

While the proof is trivial, we note that in practice, we do not expect this situation to happen often. Since, in practical settings, we need to account for environmental factors like noise, expecting to break the implementation with a single measurement is difficult. Nevertheless, this does not contradict our results. The best possible attack (that an unbounded attacker can always find) needs a single measurement, while realistic attacks need more measurements (as we do not know the best possible attack). The core idea is that we always must aim to find as powerful as possible attack. Such an attack uses the smallest number of measurements in the profiling phase and requires the smallest hyperparameter tuning phase to deliver that level of performance.

We note that the breaking of cryptographic implementations in a single measurement is not something possible only in theory, see, e.g., [12, 48] where convolutional neural networks and template attack can break different implementations in a single measurement. Additionally, in Section 6, we give several experiments where we require only a single measurement in the attack phase to break the target (provided that we allowed for good learnability).

There is also a simple alternative proof for Lemma 2. Since we know that in the ideal case, template attack can break an implementation with a single measurement, then it is enough for neural networks to approximate such a profiling model (template) built by the template attack. If a neural network can approximate a template with a desired nonzero level of error, then such a network can simulate a template attack. We emphasize that for the template attack to be able to break an implementation in a single measurement, some conditions must be met. Similar as we discussed the quality and quantity components for machine learning, we can extend it to a template attack. There, in quality, we need to account for the level of noise, leakage model, and leakage distribution. In quantity, we assume to have a sufficient number of measurements for template attack to work (and break the implementation with a single measurement). Recently, Masure et al. conducted a study of deep learning for SCA, where they also discussed the Universal Approximation Theorem [49]. While they discuss the problem in a slightly different way and use a different definition for the Universal Approximation Theorem, their results do not oppose ours.

Remark 7. If neural networks can (theoretically and under some assumptions) break the implementation in a single measurement, how is that aligned with what we know about template attack? The template attack is the most powerful one from the information-theoretic point of view, yet we claim that neural networks can reach the same performance. We believe this not to be in contradiction due to heavy assumptions on both methods. We require an unlimited number of traces for a template attack, which is naturally impossible to have. On the other side, for neural networks, there is the algorithmic learnability, where the learning process can fail for several reasons [25].

In this paper, we do not discuss machine learning methods except neural networks, but other methods would also benefit from the Efficient Attacker Framework. Indeed, we do not require the theoretical promise of being able to break implementation with a single measurement. Instead, we require a setting that limits the attacker's power in the profiling phase, which is independent of the considered attack.

6 Discussion

In this section, we start by experimentally evaluating our framework in the context of profiling SCA. Afterward, we discuss how our framework can be used in other security applications that use a supervised learning paradigm. Finally, we give several possible future research directions for profiled SCA.

6.1 Datasets

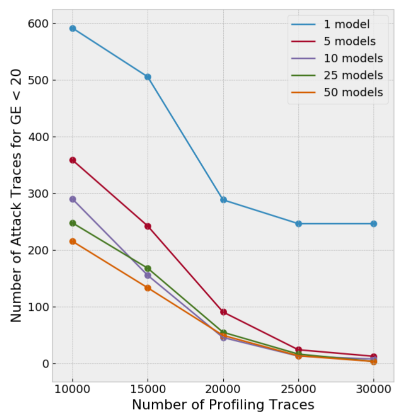
DPAv4 is a publicly available trace set obtained from a masked AES software implementation [50]. We ignore the knowledge of the masks for labeling the traces in the profiling phase. Every trace contains 2 000 features, and they correspond to the interval representing the processing of the first S-box in the first AES encryption round. The full trace set (including profiling and attack traces) contains a fixed key. There are 30 000 traces for the profiling phase, and 1 000 for the attack phase.

The second implementation refers to the ASCAD database, which is commonly used in the side-channel community for deep learning research [51][5]. This dataset corresponds to a masked AES implementation where the masks are known but also ignored in our analysis. Each trace contains 1 400 features, which represent the interval corresponding to the processing of the third S-Box in the first AES encryption round. Profiling traces contain random keys, and the attack traces have a fixed key. There are 200 000 traces for the profiling phase, and 100 000 for the attack phase.

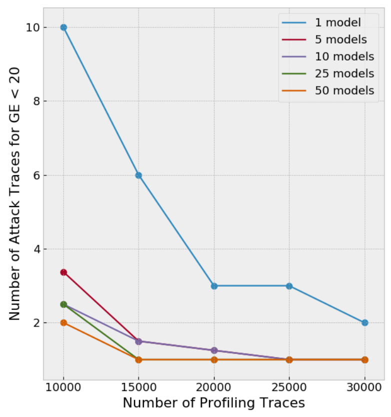
[5] Trace set publicly available at `https://github.com/ANSSI-FR/ASCAD`

6.2 Efficient Attacker Framework Evaluation

The Efficient Attacker Framework enables us to compare side-channel attacks but also gives a fair comparison between leakage models. For profiling side-channel attacks, it is often assumed to consider the most accurate leakage model, i.e., using the intermediate value as class variables (the identity leakage model), which results in 2^b classes where b is the number of considered bits. In an unsupervised setting (i.e., non-profiled attacks), using the Hamming weight or the Hamming distance leakage model is a common choice, which results in $b+1$ classes only. Using $b+1$ Hamming weight/distance classes to guess a key value in $\{0, \ldots, 2^b-1\}$ cannot result in a single trace attack on average. However, it is often overlooked that using the Hamming weight/distance leakage models may require fewer traces in the profiling phase to gain good quality estimates of the leakage models. It is, therefore, not straightforward to determine what leakage model is most suitable. Consequently, to fairly give a comparison, one should include a dependency on the number of traces in the profiling phase, as done in the Efficient Attacker Framework.



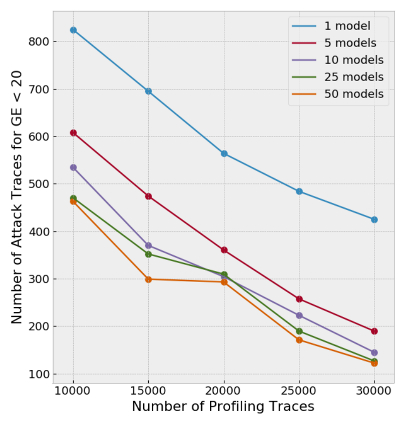
(a) Results for MLP with the Hamming weight model.



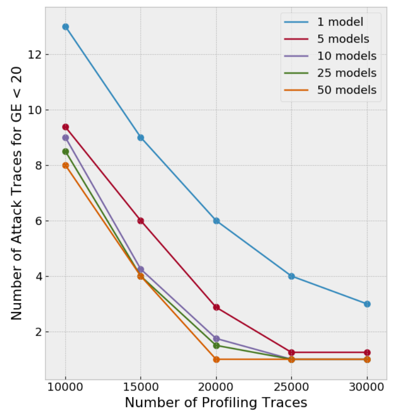
(b) Results for MLP with the identity model.

Fig. 4: Profiled SCA on the DPAv4 dataset with MLP.

As a metric, we consider guessing entropy (GE), and in particular, we give the minimum number of profiling and attack traces to reach GE < 20. For every training procedure, we randomly define hyperparameters for multilayer perceptron and convolutional neural networks according to the hyperparameter ranges provided in Tables 1 and 2. This scenario represents an optimized random hyperparameter search since the hyperparameters ranges are chosen based on the optimized minimum and maximum values (the minimal and maximal values are selected based on related works). The number of epochs is set to 50 (we observed that the models tend to overfit and degrade the generalization after 50 epochs),



(a) Results for CNNs with the Hamming weight model.



(b) Results for CNNs with the identity model.

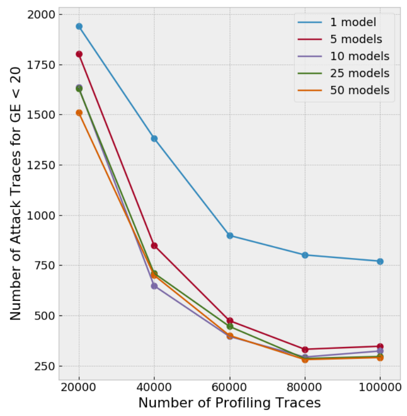
Fig. 5: Profiled SCA on the DPAv4 dataset with CNNs.

and the backpropagation algorithm optimizer is *Adam*. The weights and biases are initialized in a randomly uniform way. To avoid the overfitting, we use the batch normalization layer, which normalizes the input layer by adjusting and scaling the activations.

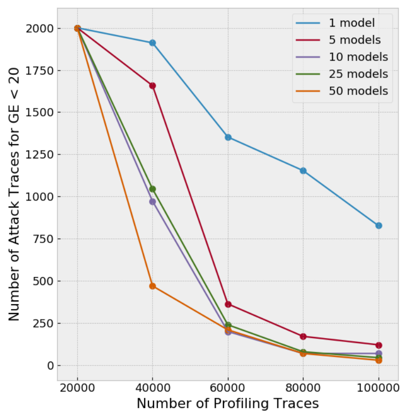
We do not discuss the time perspective here (e.g., the number of hours or days needed to conduct the experiments). We leave this out as we consider it to be hardware dependent. Still, comparing the number of tuning experiments gives a fair evaluation, regardless of the time needed to run those experiments. We note that the number of tuning experiments up to 50 is low, although we manage to break the target. There is no constraint on the number of experiments one can use with our framework. Additionally, as we work with guessing entropy, each attack is repeated 100 times, which gives much higher computational complexity than one could conclude solely based on the number of the tuning experiments.

Figures 4 to 7 illustrate examples using DPAv4 and ASCAD datasets for profiled attacks with multilayer perceptron (MLP) and convolutional neural networks (CNN). Every figure contains results for the Hamming weight and identity (i.e., intermediate value) leakage models, as AES operates on $b = 8$ bits. For each leakage model and a different number of profiling traces, we select the best neural network model out of 1, 5, 10, 25, or 50 trained profiling models. Here, the main idea is to demonstrate that the *learnability* also represents an important dimension in our framework. In Figure 4a, we examine the orange line, which depicts the best of 50 different tuning experiments. While the attack performance stays the same for scenarios between 25 000 and 30 000 profiling traces, our framework indicates that the more powerful attack is the one that uses 20 000 traces in the profiling phase (as it requires less training traces while having the same attack performance). Next, we compare the performance between a different number of hyperparameter tuning experiments. Let us consider the case with 20 000 in the profiling phase. We can see the performance to be similar if we have ten or more profiling models (tuning experiments). As such, the best attack is the one that needed the least tuning experiments. For CNN and the Hamming weight leakage model (Figure 5a, the results are similar, but we can observe some more influence of the hyperparameter tuning. For most of the considered settings, we observe the performance difference for both hyperparameter tuning and the number of profiling traces. An interesting point is 20 000 measurements in the profiling phase, and the number of models 10 or 25. As there is no difference in the attack performance, our framework indicates that the better attack is the one with ten hyperparameter tuning experiments.

Considering the identity leakage model, we observe we need much fewer attack traces to reach GE < 20. For MLP, we need 15 000 profiling traces to reach GE equal to one with a single attack measurement, and for CNN, for the same attack performance, we require 20 000 profiling traces. From Figure 4b, the advantages of the Efficient Attacker Framework are clear. The attack performance remains the same when using 15 000 profiling traces, or more. Consequently, solely judging the attack based on the number of attack traces does not give any differences, while our framework indicates that the most powerful attack is the one with 15 000 profiling traces. Similar observations can be made, for instance, for the number of tuning experiments (more than five) and 25 000 or more profiling traces. The experiments for CNNs point to the same conclusions, but with somewhat more differences in performance concerning a different number of profiling traces or tuning experiments.

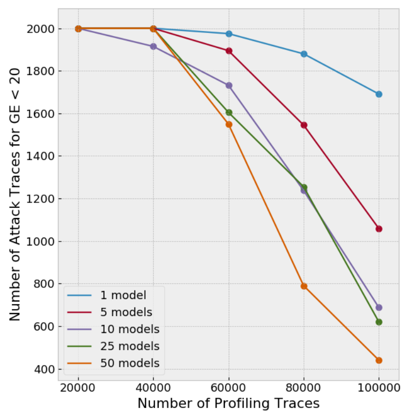


(a) Results for MLP with the Hamming weight model.

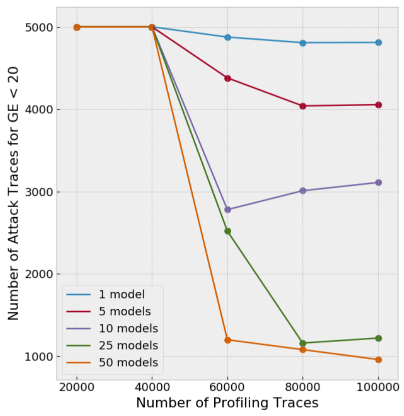


(b) Results for MLP with the identity model.

Fig. 6: Profiled SCA on the ASCAD dataset with MLP.



(a) Results for CNNs with the Hamming weight model.



(b) Results for CNNs with the identity model.

Fig. 7: Profiled SCA on the ASCAD dataset with CNNs.

Hyperparameter	**min**	**max**	**step**
Learning Rate	0.0001	0.001	0.0001
Mini-batch	100	1 000	100
Dense (fully-connected) layers	2	8	1
Neurons (for dense or fc layers)	500	1 000	100
Activation function (all layers)	ReLU, Tanh, ELU, or SELU		

Table 1: Hyperparameter search space for multilayer perceptron.

Next, in Figures 6a and 7a, we depict results for the ASCAD dataset and the Hamming weight leakage model. Again, we can observe the influence of both profiling traces and the number of tuning experiments on the attack performance. Note the interesting results for CNN with a smaller number of profiling traces as the attack performance is relatively bad, and then, with 60 000 profiling traces or more, it quickly improves for up to five times.

Figures 6b and 7b depict results for the ASCAD dataset in the identity leakage model. For MLP, we can observe a strong influence of the number of the tuning experiments and profiling traces up to 60 000 traces. Afterward, the benefit of more profiling measurements is considerably smaller, where for more than 80 000 profiling traces, we do not see much improvement. For CNN, an insufficient profiling traces number significantly influences the performance, and more tuning experiments cannot circumvent this. What is more, we observe how for a specific number of profiling traces, the number of tuning experiments plays

Hyperparameter	min	max	step
Learning Rate	0.0001	0.001	0.0001
Mini-batch	100	1 000	100
Convolution layers	1	2	1
Filters	8	32	4
Kernel Size	10	20	2
Stride	5	10	5
Dense (fully-connected) layers	2	3	1
Neurons (for dense or fc layers)	500	1 000	100
Activation function (all layers)	ReLU, Tanh, ELU or SELU		

Table 2: Hyperparameters search space for convolutional neural network.

a significant role. More precisely, we can recognize settings where adding more profiling traces does not help.

On a general level, while not the core research point in this work, we note that the identity leakage model requires fewer attack traces to reach GE < 20, which is expected. MLP exhibits somewhat better performance than CNN for a smaller number of profiling traces, which is again in line with results in related works. It is important to observe how the learnability constraint directly influences the required combination of the number of profiling and attack traces to reach a low guessing entropy. Moreover, one can choose a trade-off between profiling traces N and attack traces Q while still being able to perform a successful attack.

While our framework aims to find the minimal number of profiling traces and tuning experiments to mount a successful attack, we never state what those numbers should be. Indeed, the experiments with only two datasets already depict a radically different number of profiling and attack traces (coupled with the influence of the number of tuning experiments). Providing actual values makes sense only when the whole experimental environment is considered (datasets, algorithms, environmental settings, etc.), and, even more importantly, when one compares experiments on the same targets but with different settings. All our experiments strongly confirm that the number of profiling traces and the number of experiments (complexity) play a paramount role and should be included in proper performance analysis.

6.3 Advantages of Efficient Attacker Framework

Normally, it is expected that an attacker would make use of the maximum possible amount of profiling traces to build a model (templates, deep neural networks, etc.). Similarly, the amount of attack traces tends to be maximized to estimate the model exploitation capability better. In cases when the learning model is inefficient (i.e., unable to fit existing leakage) and all available side-channel measurements are used, the attacker or evaluator has a limited view of which component has a significant impact on the attack results, which can lead to overestimating the security of the target.

In this case, the reference metric would be the guessing entropy of a single experiment (see, for example, the two lines in Figure 1), which say nothing about the influence of the number of measurements and tuning experiments for the security of the assessed target. Therefore, the usage of the Efficient Attacker Framework provides a better representation of the influence of the number of profiling traces, attack traces, and tuning experiments. Here, we provided analysis about the efficiency of an attack having GE < 20 as a reference metric. Of course, the framework can be adapted to any metric that describes the attack's efficiency.

While the benefits of depicting the results with our framework are evident, one can ask whether we lost some information when compared to the traditional result depiction (e.g., cf. Figures 1 and 4a). We claim this not to be true due to two reasons. First, all the relevant information is kept so the attacker can still depict traditional results. Second, once the appropriate performance level is set (e.g., guessing entropy value equal to δ), it is less relevant to observe how is that value reached (as values above the threshold are out of the attacker's reach).

6.4 The Efficient Attacker Framework Beyond Profiling SCA

We discussed our framework for profiling SCA where side-channels are power or EM radiation. There is no reason not to extend to other types of side-channels or even entirely different security applications. Indeed, as long as the two core assumptions for the framework are followed, it can be applied (depending on the application, the extension to the optimal attack with neural networks may not be possible). We briefly discuss a few applications where the Efficient Attacker Framework could be useful. Zhang, Zhang, and Lee analyzed cache side-channels with deep neural networks. There, they aim to quantitatively evaluate the effectiveness of such attacks and defenses [52]. Continuing with side-channels, Genkin et al. used machine learning to detect screen content through remote acoustic side-channels [53]. Sirinam et al. proposed a website fingerprinting attack with convolutional neural networks [54]. The authors investigated the influence of the training set size but did not try to find the minimal training and testing set sizes that result in a specific performance. Shen et al. used recurrent neural networks to predict the specific steps taken by an adversary when performing an attack [55]. We see a place for our framework in the domain of adversarial attacks (e.g., in poisoning attacks) and defenses where it is interesting to consider the minimum effort (number of measurements, tuning experiments, etc.) needed for successful attacks and defenses [56, 57].

6.5 Future Directions

Besides using the Efficient Attacker Framework to compare profiling side-channel attacks and learning models on various implementations, we emphasize the comparison of attacks in the presence of various side-channel countermeasures. Such a study will highlight if different types of side-channel countermeasures differ in their complexity of profiling, where it may be of particular interest to increase the profiling phase's complexity more than the attack phase or find a suitable trade-off to protect against powerful attackers.

Several papers have recently proposed to conduct data augmentation to construct additional synthetic measurements to be used in the profiling phase [10, 11]. Our framework does not limit the use of such synthetic examples in the sense that those measurements are not counted in the profiling set N since they are built from the original measurements. It would be interesting to investigate the limits of data augmentation in SCA, i.e., can we construct good, synthetic examples from a limited number of real measurements.

In this paper, we consider only the profiling SCA settings, but there is no reason not to extend our framework to other scenarios that use profiling (supervised learning) methods. We are confident that it would provide similar insights and fairness in the performance comparison.

7 Conclusions

In this paper, we discuss how to evaluate attacks when considering the profiling side-channel analysis. We argue that considering only an unbounded attacker can have negative effects on the way how side-channel analysis is performed in addition to not being realistic. We propose a new framework, denoted as the Efficient Attacker Framework, where we explore the number of measurements and hyperparameter tuning experiments required in the profiling phase such that an attacker is still successful.

Next, we connect the notion of the unbounded attacker with the Universal Approximation Theorem, and we show that because of it, the attacker could be able to break implementations with only a single measurement, provided that some conditions are met. While this does not often occur in practice, we still consider the "race" for the most powerful attacks meaningless in the unbounded scenario if the theory indicates that breaking an implementation in a single measurement is possible.

We consider our new framework more realistic but also more adept for experimental evaluations since it allows us to compare different results in a more unified way. In particular, our framework will trigger more research that is meaningful not only for academia but also for evaluation labs. Finally, our framework is relevant beyond profiling side-channel analysis and can be used in any supervised learning setting.

References

1. Mangard, S., Oswald, E., Popp, T.: Power Analysis Attacks: Revealing the Secrets of Smart Cards. Springer (December 2006) ISBN 0-387-30857-1, `http://www.dpabook.org/`.
2. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Proceedings of CRYPTO'96. Volume 1109 of LNCS., Springer-Verlag (1996) 104–113
3. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology. CRYPTO '99, London, UK, UK, Springer-Verlag (1999) 388–397
4. Zhao, M., Suh, G.E.: FPGA-Based Remote Power Side-Channel Attacks. In: 2018 IEEE Symposium on Security and Privacy (SP). (May 2018) 229–244
5. Quisquater, J.J., Samyde, D.: ElectroMagnetic Analysis (EMA): Measures and Counter-measures for Smart Cards. In Attali, I., Jensen, T., eds.: Smart Card Programming and Security, Berlin, Heidelberg, Springer Berlin Heidelberg (2001) 200–210
6. Schindler, W., Lemke, K., Paar, C.: A Stochastic Model for Differential Side Channel Cryptanalysis. In LNCS, ed.: CHES. Volume 3659 of LNCS., Springer (Sept 2005) 30–46 Edinburgh, Scotland, UK.
7. Choudary, O., Kuhn, M.G.: Efficient template attacks. In Francillon, A., Rohatgi, P., eds.: Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers. Volume 8419 of LNCS., Springer (2013) 253–270
8. Heuser, A., Zohner, M.: Intelligent Machine Homicide - Breaking Cryptographic Devices Using Support Vector Machines. In Schindler, W., Huss, S.A., eds.: COSADE. Volume 7275 of LNCS., Springer (2012) 249–264
9. Lerman, L., Poussier, R., Bontempi, G., Markowitch, O., Standaert, F.: Template Attacks vs. Machine Learning Revisited (and the Curse of Dimensionality in Side-Channel Analysis). In: COSADE 2015, Berlin, Germany, 2015. Revised Selected Papers. (2015) 20–33
10. Picek, S., Heuser, A., Jovic, A., Bhasin, S., Regazzoni, F.: The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(1) (2019) 209–237
11. Cagli, E., Dumas, C., Prouff, E.: Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures - Profiling Attacks Without Pre-processing. In: Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings. (2017) 45–68
12. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(3) (May 2019) 148–179
13. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems **2**(4) (Dec 1989) 303–314
14. Bhasin, S., Chattopadhyay, A., Heuser, A., Jap, D., Picek, S., Shrivastwa, R.R.: Mind the portability: A warriors guide through realistic profiled side-channel analysis. Cryptology ePrint Archive, Report 2019/661 (2019) `https://eprint.iacr.org/2019/661`.

15. Das, D., Golder, A., Danial, J., Ghosh, S., Raychowdhury, A., Sen, S.: X-deepsca: Cross-device deep learning side channel attack*. In: 2019 56th ACM/IEEE Design Automation Conference (DAC). (June 2019) 1–6
16. Chari, S., Rao, J.R., Rohatgi, P.: Template Attacks. In: CHES. Volume 2523 of LNCS., Springer (August 2002) 13–28 San Francisco Bay (Redwood City), USA.
17. Picek, S., Heuser, A., Jovic, A., Ludwig, S.A., Guilley, S., Jakobovic, D., Mentens, N.: Side-channel analysis and machine learning: A practical perspective. In: 2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017. (2017) 4095–4102
18. Hospodar, G., Gierlichs, B., De Mulder, E., Verbauwhede, I., Vandewalle, J.: Machine learning in side-channel analysis: a first study. Journal of Cryptographic Engineering **1** (2011) 293–302 10.1007/s13389-011-0023-x.
19. Lerman, L., Bontempi, G., Markowitch, O.: Power analysis attack: An approach based on machine learning. Int. J. Appl. Cryptol. **3**(2) (June 2014) 97–115
20. Heuser, A., Picek, S., Guilley, S., Mentens, N.: Side-channel analysis of lightweight ciphers: Does lightweight equal easy? In: Radio Frequency Identification and IoT Security - 12th International Workshop, RFIDSec 2016, Hong Kong, China, November 30 - December 2, 2016, Revised Selected Papers. (2016) 91–104
21. Heuser, A., Picek, S., Guilley, S., Mentens, N.: Lightweight ciphers and their side-channel resilience. IEEE Transactions on Computers **PP**(99) (2017) 1–1
22. Picek, S., Samiotis, I.P., Kim, J., Heuser, A., Bhasin, S., Legay, A.: On the performance of convolutional neural networks for side-channel analysis. In Chattopadhyay, A., Rebeiro, C., Yarom, Y., eds.: Security, Privacy, and Applied Cryptography Engineering, Cham, Springer International Publishing (2018) 157–176
23. Maghrebi, H., Portigliatti, T., Prouff, E.: Breaking cryptographic implementations using deep learning techniques. In: Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings. (2016) 3–26
24. Hettwer, B., Gehrer, S., Güneysu, T.: Profiled power analysis attacks using convolutional neural networks with domain knowledge. In Cid, C., Jr., M.J.J., eds.: Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers. Volume 11349 of Lecture Notes in Computer Science., Springer (2018) 479–498
25. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016) `http://www.deeplearningbook.org`.
26. Collobert, R., Bengio, S.: Links Between Perceptrons, MLPs and SVMs. In: Proceedings of the Twenty-first International Conference on Machine Learning. ICML '04, New York, NY, USA, ACM (2004) 23–
27. Standaert, F.X., Malkin, T., Yung, M.: A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In: EUROCRYPT. Volume 5479 of LNCS., Springer (April 26-30 2009) 443–461 Cologne, Germany.
28. Whitnall, C., Oswald, E.: A Fair Evaluation Framework for Comparing Side-Channel Distinguishers. J. Cryptographic Engineering **1**(2) (2011) 145–160
29. Whitnall, C., Oswald, E.: A Comprehensive Evaluation of Mutual Information Analysis Using a Fair Evaluation Framework. In Rogaway, P., ed.: CRYPTO. Volume 6841 of Lecture Notes in Computer Science., Springer (2011) 316–334
30. Guilley, S., Heuser, A., Rioul, O.: A Key to Success - Success Exponents for Side-Channel Distinguishers. In Biryukov, A., Goyal, V., eds.: Progress in Cryptology - INDOCRYPT 2015 - 16th International Conference on Cryptology in India, Bangalore, India, December 6-9, 2015, Proceedings. Volume 9462 of Lecture Notes in Computer Science., Springer (2015) 270–290

31. Duc, A., Faust, S., Standaert, F.: Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Oswald, E., Fischlin, M., eds.: Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I. Volume 9056 of Lecture Notes in Computer Science., Springer (2015) 401–429
32. Bronchain, O., Hendrickx, J.M., Massart, C., Olshevsky, A., Standaert, F.: Leakage certification revisited: Bounding model errors in side-channel security evaluations. In Boldyreva, A., Micciancio, D., eds.: Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I. Volume 11692 of Lecture Notes in Computer Science., Springer (2019) 713–737
33. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)
34. publication 140-3, N.F.F.I.P.S.: Security Requirements for Cryptographic Modules (Draft, Revised). (09/11 2009) 63 `http://csrc.nist.gov/groups/ST/FIPS140_3/`.
35. ISO/IEC JTC 1/SC 27 IT Security techniques: ISO/IEC 17825:2016 Information technology – Security techniques – Testing methods for the mitigation of non-invasive attack classes against cryptographic modules (January 2016) `https://www.iso.org/standard/60612.html`.
36. ISO/IEC JTC 1/SC 27 IT Security techniques: ISO/IEC 15408-1:2009 Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model (January 2014) `https://www.iso.org/standard/50341.html`.
37. Common Criteria: Supporting Document Mandatory Technical Document Application of Attack Potential to Smartcards (2013) `https://www.commoncriteriaportal.org/files/supdocs/CCDB-2013-05-002.pdf`.
38. Heuser, A., Kasper, M., Schindler, W., Stöttinger, M.: A New Difference Method for Side-Channel Analysis with High-Dimensional Leakage Models. In Dunkelman, O., ed.: CT-RSA. Volume 7178 of Lecture Notes in Computer Science., Springer (2012) 365–382
39. Cao, Y., Zhou, Y., Yu, Z.: On the negative effects of trend noise and its applications in side-channel cryptanalysis. Chinese J. Electron. **23** (2014) 366–370
40. Veyrat-Charvillon, N., Gérard, B., Renauld, M., Standaert, F.X.: An Optimal Key Enumeration Algorithm and Its Application to Side-Channel Attacks. In Knudsen, L.R., Wu, H., eds.: Selected Areas in Cryptography. Volume 7707 of Lecture Notes in Computer Science., Springer (2012) 390–406
41. Common Criteria Management Board: Common Methodology for Information Technology Security Evaluation Evaluation methodology, Version 3.1, Revision 4, CCMB-2012-09-004 (September 2012) `https://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R4.pdf`.
42. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5) (July 1989) 359–366
43. Leshno, M., Schocken, S.: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks **6** (1993) 861–867
44. Barron, A.R.: Universal approximation bounds for superpositions of a sigmoidal function. IEEE Transactions on Information Theory **39**(3) (May 1993) 930–945

45. Yarotsky, D.: Universal approximations of invariant maps by neural networks. CoRR **abs/1804.10306** (2018)
46. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining Smart-Card Security under the Threat of Power Analysis Attacks. IEEE Trans. Computers **51**(5) (2002) 541–552
47. Feldman, M.B.: A proof of lusin's theorem. The American Mathematical Monthly **88**(3) (1981) 191–192
48. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Methodology for Efficient CNN Architectures in Profiling Attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems **2020**(1) (Nov. 2019) 1–36
49. Masure, L., Dumas, C., Prouff, E.: A comprehensive study of deep learning for side-channel analysis. Cryptology ePrint Archive, Report 2019/439 (2019) `https://eprint.iacr.org/2019/439`.
50. TELECOM ParisTech SEN research group: DPA Contest (4^{th} edition) (2013–2014) `http://www.DPAcontest.org/v4/`.
51. Prouff, E., Strullu, R., Benadjila, R., Cagli, E., Dumas, C.: Study of deep learning techniques for side-channel analysis and introduction to ASCAD database. IACR Cryptology ePrint Archive **2018** (2018) 53
52. Zhang, T., Zhang, Y., Lee, R.B.: Analyzing cache side channels using deep neural networks. In: Proceedings of the 34th Annual Computer Security Applications Conference. ACSAC '18, New York, NY, USA, Association for Computing Machinery (2018) 174–186
53. Genkin, D., Pattani, M., Schuster, R., Tromer, E.: Synesthesia: Detecting screen content via remote acoustic side channels. In: 2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019, IEEE (2019) 853–869
54. Sirinam, P., Imani, M., Juarez, M., Wright, M.: Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. CCS '18, New York, NY, USA, Association for Computing Machinery (2018) 1928–1943
55. Shen, Y., Mariconti, E., Vervier, P.A., Stringhini, G.: Tiresias: Predicting Security Events Through Deep Learning. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. CCS '18, New York, NY, USA, Association for Computing Machinery (2018) 592–605
56. Vorobeychik, Y., Kantarcioglu, M., Brachman, R.: Adversarial Machine Learning. Morgan & Claypool Publishers (2018)
57. Ling, X., Ji, S., Zou, J., Wang, J., Wu, C., Li, B., Wang, T.: DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model. In: 2019 IEEE Symposium on Security and Privacy (SP). (May 2019) 673–690