REGULAR PAPER

Template attacks versus machine learning revisited and the curse of dimensionality in side-channel analysis: extended version

Liran Lerman[1] · Romain Poussier[2] · Olivier Markowitch[1] · François-Xavier Standaert[2]

Received: 31 January 2016 / Accepted: 11 April 2017

Abstract Template attacks and machine learning are two popular approaches to profiled side-channel analysis. In this paper, we aim to contribute to the understanding of their respective strengths and weaknesses, with a particular focus on their curse of dimensionality. For this purpose, we take advantage of a well-controlled simulated experimental setting in order to put forward two important aspects. First and from a theoretic point of view, the data complexity of template attacks is not sensitive to the dimension increase in side-channel traces given that their profiling is perfect. Second and from a practical point of view, concrete attacks are always affected by (estimation and assumption) errors during profiling. As these errors increase, machine learning gains interest compared to template attacks, especially when based on random forests. We then clarify these results thanks to the bias–variance decomposition of the error rate recently introduced in the context side-channel analysis.

Keywords Side-channel attacks · Template attacks · Machine learning · Curse of dimensionality · Bias–variance decomposition

1 Introduction

In a side-channel attack, an adversary targets a cryptographic device that emits a measurable leakage depending on the manipulated data and/or the executed operations. Typical examples of physical leakages include the power consumption [21], the processing time [20] and the electromagnetic emanation [13].

Evaluating the degree of resilience of a cryptographic implementation is an important concern, e.g., for modern smart cards. In this respect, profiled attacks are handy tools since they can be used to approach their worst-case security level [36]. These attacks rely on the optimal strategy of recovering the key via a maximum likelihood strategy. In practice, they essentially work in two steps: first a leakage model is estimated during a profiling phase, then the leakage model is exploited to extract key-dependent information in an online phase. Several approaches to profiling have been introduced in the literature. Template attacks (TA), e.g., based on a Gaussian assumption [5], are a typical example. The stochastic approach exploiting linear regression (LR) is a frequently considered alternative [34]. More recently, solutions relying on machine learning (ML) have also been investigated [1,3,15–19,22–24,26,27,30,31]. These previous works support the claim that machine learning-based attacks are effective and lead to successful key recoveries. This is natural since they essentially exploit the same discriminating criteria as template attacks and linear regression (i.e., a difference in the mean traces corresponding to different intermediate computations if an unprotected implementation is targeted—a difference in higher-order statistical moments if the device is protected with masking). By contrast, it remains unclear whether machine learning can lead to more efficient attacks, either in terms of profiling or in terms of online key recovery. Previous publications conclude in one or the other direction, depending on the implementation scenario considered, which is inherent to such experimental studies [1,17,19,24,26].

In this paper, we aim to complement these previous works with a more systematic investigation of the conditions under

✉ Liran Lerman
llerman@ulb.ac.be

[1] Département d'informatique, Université Libre de Bruxelles, Brussels, Belgium

[2] ICTEAM/INGI, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Published online: 22 April 2017

which machine learning-based attacks may outperform template attack (or not).[1] For this purpose, we start with the general intuition that machine learning-based approaches are generally useful in order to deal with high-dimensional data spaces. Following, our contributions are twofold. First, we tackle the (theoretic) question whether the addition of useless (i.e., non-informative) leakage samples in leakage traces has an impact on their informativeness if a perfect profiling phase is achieved. We show that the (mutual) information leakage estimated with a template attack exploiting such a perfect model is independent of the number of useless dimensions if the useless leakage samples are independent of the useful ones. This implies that machine learning-based attacks cannot be more efficient than template attacks in the online phase if the profiling is sufficient. Second, we study the practical counterpart of this question and analyze the impact of imperfect profiling on our conclusions. For this purpose, we rely on a simulated experimental setting, where the number of (informative and useless) dimensions is used as a parameter. Using this setting, we evaluate the curse of dimensionality for concrete template attack and compare it with machine learning-based attacks exploiting support vector machines (SVM) and random forests (RF). That is, we considered support vector machine as a popular tool in the field of side-channel analysis, and random forest as an interesting alternative (since its random feature selection makes its behavior quite different than template attack and support vector machine).

Our experiments essentially conclude that template attack outperform machine learning-based attacks whenever the number of dimensions can be kept reasonably low, e.g., thanks to a selection of points of interests (POI), and that machine learning (and random forest in particular) become(s) interesting in "extreme" profiling conditions (i.e., with large traces and a small profiling set)—which possibly arises when little information about the target device is available to the adversary. We then complement these results with an additional analysis based on the bias–variance decomposition of the error rate, which was recently introduced in the side-channel literature [25]. The bias–variance decomposition allows separating the error rate of an attack in three weighted terms, among them the bias and the variance terms. The values of the variance and the bias relate to the attack complexity: A strategy with a high variance means a high sensitivity to the profiling set while an attack with a high bias indicates a high systematic error. This last analysis brings an interesting complement to our results of COSADE 2015 [27], since it adds a sound statistical explanation to our findings. Namely, we can now show that template attacks have a high variance while a random forest represents an interesting approach to reduce this term in high-dimensional data spaces. The bias–variance decomposition also sheds new light on the results obtained in previous(ly listed) papers comparing machine learning algorithms with conventional profiled attacks.

As a side remark, we also observe that most current machine learning-based attacks rate key candidates according to (heuristic) scores rather than probabilities. This prevents the computation of probability-based metrics (such as the mutual/perceived information [32]). It may also have an impact on the efficiency of key enumeration [37], which is an interesting scope for further investigation.

The rest of the paper is organized as follows. Section 2 contains notations, the attacks considered, our experimental setting and evaluation metrics. Section 3 presents our theoretic result on the impact of non-informative leakage samples in perfect profiling conditions. Section 4 discusses practical (simulated) experiments in imperfect profiling conditions. Section 5 analyses our results based on the bias–variance decomposition. Eventually, Sect. 6 concludes the paper and discusses perspectives of future work.

2 Background

2.1 Notations

We use capital letters for random variables and small caps for their realizations. We use sans serif font for functions (e.g., f) and calligraphic fonts for sets (e.g., $\mathcal{A}$). We denote the conditional probability of a random variable A given B with $\Pr[A|B]$ and use the acronym SNR for the signal-to-noise ratio.

2.2 Template attacks

Let $l_{x,k}$ be a leakage trace measured on a cryptographic device that manipulates a target intermediate value $v = \mathsf{f}(x, k)$ associated with a known plaintext (byte) x and a secret key (byte) k. In a template attack, the adversary first uses a set of profiling traces $\mathcal{L}_{\mathrm{PS}}$ in order to estimate a leakage model, next denoted $\hat{\Pr}_{\mathrm{model}}[l_{x,k} \mid \hat{\theta}_{x,k}]$, where $\hat{\theta}_{x,k}$ represents the (estimated) parameters of the leakage probability density function (PDF). The set of profiling traces is typically obtained by measuring a device that is similar to the target, yet under control of the adversary. Next, during the online phase, the adversary uses a set of new attack traces $\mathcal{L}_{\mathrm{AS}}$ (obtained by measuring the target device) and selects the secret key (byte) $\tilde{k}$ maximizing the product of posterior probabilities:

$$\tilde{k} = \underset{k^*}{\operatorname{argmax}} \prod_{l_{x,k} \in \mathcal{L}_{\mathrm{AS}}} \frac{\hat{\Pr}_{\mathrm{model}}\left[l_{x,k} \mid \hat{\theta}_{x,k^*}\right] \cdot \Pr[k^*]}{\hat{\Pr}_{\mathrm{model}}[l_{x,k}]}. \qquad (1)$$

[1] Note that the gain of linear regression-based attacks over template attack is known and has been analyzed, e.g., in [14,35]. Namely, it essentially depends on the size of the basis used in linear regression.

Concretely, the seminal template attack paper suggested to use Gaussian estimations for the leakage PDF [5]. We will follow a similar approach and consider a Gaussian (simulated) experimental setting. It implies that the parameters $\hat{\theta}_{x,k}$ correspond to mean vectors $\hat{\mu}_{x,k}$ and covariance matrices $\hat{\Sigma}_{x,k}$. However, we note that any other probability density function estimation could be considered by the adversary/evaluator [12]. We will further consider two types of template attacks: in the naive template attack (NTA), we will indeed estimate one covariance matrix per intermediate value; in the efficient template attack (ETA), we will pool the covariance estimates (assumed to be equal) across all intermediate values, as previously suggested in [7].

In the following, we will keep the $l_{x,k}$ and v notations for leakage traces and intermediate values, and sometimes omit the subscripts for simplicity.

2.3 Support vector machines

In their basic (two-classes) context, support vector machine essentially aims at estimating Boolean functions [8]. For this purpose, it first performs a supervised learning with labels (e.g., $v = -1$ or $v = 1$), annotating each sample of the profiling set. The binary support vector machine estimates a hyperplane $y = \hat{w}^{\top} l + \hat{b}$ that separates the two classes with the largest possible margin, in the geometrical space of the vectors. Then in the attack phase, any new trace l will be assigned a label $\tilde{v}$ as follows:

$$\tilde{v} = \begin{cases} 1 & \text{if } (\hat{w}^{\top} l + \hat{b}) \geq 1, \\ -1 & \text{otherwise.} \end{cases} \tag{2}$$

Mathematically, support vector machine finds the parameters $\hat{w} \in \mathbb{R}^{n_s}$ (where n_s is the number of time samples per trace) and $\hat{b} \in \mathbb{R}$ by solving the convex optimization problem:

$$\begin{aligned} &\min_{w,b} && \frac{1}{2}(w^{\top} w), \\ &\text{subject to} && v(w^{\top} \phi(l_v) + b) \geq 1, \end{aligned} \tag{3}$$

where ϕ denotes a projection function that maps the data into a higher (sometimes infinite) dimensional space usually denoted as the feature space. Our experiments considered a radial basis kernel function ϕ (RBF), which is a commonly encountered solution, both in the machine learning field and the side-channel communities. The radial basis kernel function maps the traces into an infinite dimensional Hilbert space in order to find a hyperplane that efficiently discriminate the traces. It is defined by a parameter γ that essentially relates to the "variance" of the model. Roughly, the variance of a model is a measure on the variance of its output in function of the variance of the profiling set. The higher the value of γ, the lower the variance of the model is. Intuitively, the variance of a model therefore relates to its complexity (e.g., the higher the number of points per trace, the higher the variance of the model). We always selected the value of γ as the inverse of the number of points per trace, which is a natural choice to compensate the increase in the model variance due to the increase in the number of points per trace. Future works could focus on other strategies to select this parameter, although we do not expect them to have a strong impact on our conclusions.

When the problem of Eq. 3 is feasible with respect to the constraints, the data are said to be linearly separable in the feature space. As the problem is convex, there is a guarantee to find a unique global minimum. Support vector machine can be generalized to multi-class problems (which will be useful in our context with typically 256 target intermediate values) and produce scores for intermediate values based on the distance to the hyperplane. In our experiments, we considered the "one-against-one" approach. In a one-against-one strategy, the adversary builds one support vector machine for each possible pair of target values. During the attack phase, the adversary selects the target value with a majority vote among the set of support vector machines. We refer to [9] for a complete explanation.

2.4 Random forests

Decision trees are classification models that use a set of binary rules to calculate a target value. They are structured as diagrams (tree) made of nodes and directed edges, where nodes can be of three types: root (i.e., the top node in the tree), internal (represented by a circle in Fig. 1) and leaf (represented by a square in Fig. 1). In our side-channel context, we typically consider decision trees in which (1) the value associated with a leaf is a class label corresponding to the target to be recovered, (2) each edge is associated with a test on the value of a time sample in the leakage traces, and (3) each internal node has one incoming edge from a node called the parent node, as also represented in Fig. 1.

In the profiling phase, learning data are used to build the model. For this purpose, the learning set is first associated with the root. Then, this set is split based on a time sample that most effectively discriminates the sets of traces associated with different target intermediate values. Each subset newly created is associated with a child node. The tree generator repeats this process on each derived subset in a recursive manner, until the child node contains traces associated with the same target value or the gain to split the subset is less than some threshold. That is, it essentially determines at which time sample to split, the value of the split, and the decision to stop or to split again. It then assigns terminal nodes to a class (i.e., intermediate value). Next, in the attack phase, the model simply predicts the target intermediate value by applying the

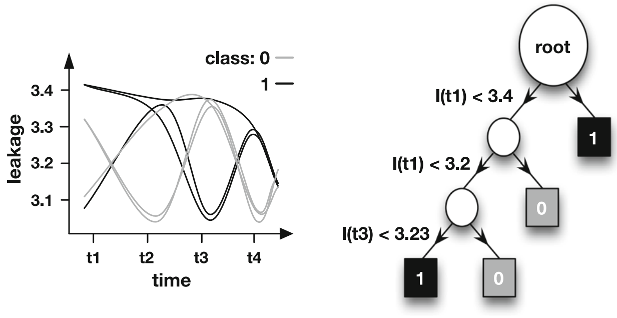

Fig. 1 Decision tree with two classes [$l(t1)$ is the leakage at time $t1$]

classification rules to the new traces to classify. We refer to [33] for more details on decision trees.

The random forests (RF) introduced by Breiman can be seen as a collection of classifiers using many (unbiased) decision trees as models [4]. It relies on model averaging (aka bagging) that leads to have a low variance of the resulting model. After the profiling phase, random forest returns the most consensual prediction for a target value through a majority vote among the set of trees. Random forests are based on three main principles. First, each tree is constructed with a different learning set by re-sampling (with replacement) the original dataset. Secondly, the nodes of the trees are split using the best time sample among a subset of randomly chosen ones (by contrast to conventional trees where all the time samples are used). The size of this subset was set to the square of the number of time samples (i.e., $\sqrt{n_s}$) as suggested by Breiman. These features allow obtaining decorrelated trees, which improves the accuracy of the resulting random forest model. Finally, and unlike conventional decision trees as well, the trees of a random forest are fully grown and are not pruned, which possibly leads to overfitting (i.e., each tree has a low bias but a high variance) that is reduced by averaging the trees. The main (meta-) parameters of a random forest are the number of trees. Intuitively, increasing the number of trees reduces the instability (aka variance) of the models. We set this number to 500 by default, which was sufficient in our experiments in order to show the strength of this model compared to template attack. We leave the detailed investigation of these parameters as an interesting scope for further research.

2.5 Experimental setting

Let $l_{p,k}(t)$ be the t-th time sample of the leakage trace $l_{p,k}$. We consider contexts where each trace $l_{p,k}$ represents a vector of n_s samples, that is:

$$l_{p,k} = \left\{ l_{p,k}(t) \in \mathbb{R} \mid t \in [1; n_s] \right\}. \tag{4}$$

Each sample represents the output of a leakage function. The adversary has access to a profiling set of N_p traces per target intermediate value, in which each trace has d informative samples and u uninformative samples (with $d + u = n_s$). The informative samples are defined as the sum of a deterministic part representing the useful signal (denoted as δ) and a random Gaussian part representing the noise (denoted as ϵ), that is:

$$l_{p,k}(t) = \delta_t(p,k) + \epsilon_t, \tag{5}$$

where the noise is independent and identically distributed for all t's. In our experiments, the deterministic part δ corresponds to the output of the AES S-box, iterated for each time sample and sent through a function G, that is:

$$\delta_t(p,k) = \mathsf{G}\left(\mathsf{SBox}^t(p \oplus k)\right), \tag{6}$$

where:

$$\mathsf{SBox}^1(p \oplus k) = \mathsf{SBox}(p \oplus k),$$
$$\mathsf{SBox}^t(p \oplus k) = \mathsf{SBox}\left(\mathsf{SBox}^{t-1}(p \oplus k)\right).$$

Concretely, we considered a function G that is a weighted sum of the S-box output bits. However, all our results can be generalized to other functions (preliminary experiments did not exhibit any deviation with highly nonlinear leakage functions—which is expected in a first-order setting where the leakage informativeness essentially depends on the SNR [29]). We set our signal variance to 1 and used Gaussian distributed noise variables ϵ_t with mean 0 and variance σ^2 (i.e., the SNR was set to $\frac{1}{\sigma^2}$). Eventually, uninformative samples were simply generated with a noisy part. This simulated setting is represented in Fig. 2, and its main parameters can be summarized as follows:

- number of informative points per trace (denoted as d),
- number of uninformative points per trace (denoted as u),
- number of profiling traces per intermediate value (denoted as N_p),
- number of traces in the attack step (noted N_a),
- noise variance (denoted as σ^2) and SNR.

The rationale of this simulator is that, in practice, the quantity of information in each sample varies, leading to uninformative samples and informative samples containing different quantities of information (from very low to very high) on the target value. This results to an open problem in practice: which sample should be removed. In this context, the main purpose of our simulator is to exhibit the impact of an increase in the number of dimensions for profiled attacks.

2.6 Evaluation metrics

The efficiency of side-channel attacks can be quantified according to various metrics. We will use information theoretic and security metrics advocated in [36].

2.6.1 Success rate (SR) and error rate (ER)

For an attack targeting a part of the key (e.g., a key byte) and allowing to sort the different candidates, we define the success rate of order o as the probability that the correct subkey is ranked among the first o candidates. The error rate represents the probability that the correct subkey is *not* ranked among the first o candidates. The success rate and the error rate are generally computed in function of the number of attack traces N_a (given a model that has been profiled using N_p traces). In the rest of this paper, we focus on the success rate of order 1 (i.e., the correct key rated first).

2.6.2 Perceived/mutual information (PI/MI)

Let X, K, L be random variables representing a target key byte, a known plaintext and a leakage trace. The perceived information $\hat{\mathrm{PI}}(K; X, L)$ between the key and the leakage is defined as [32]:[2]

$$H(K) + \sum_{k \in \mathcal{K}} \Pr[k] \sum_{x \in \mathcal{X}} \Pr[x] \sum_{l \in \mathcal{L}} \Pr_{\text{chip}}[l|x,k] \cdot \log_2 \hat{\Pr}_{\text{model}}[k|x,l].$$

The perceived information measures the adversary's ability to interpret measurements coming from the true (unknown) chip distribution $\Pr_{\text{chip}}[l|x,k]$ with an estimated model $\hat{\Pr}_{\text{model}}[l|x,k]$ while $\Pr_{\text{chip}}[l|x,k]$ is generally obtained by sampling the chip distribution (i.e., making measurement). Of particular interest for the next section will be the context of *perfect profiling*, where we assume that the adversary's model and the chip distribution are identical (which, strictly speaking, can only happen in simulated experimental settings since any profiling based on real traces will at least be imperfect because of small estimation errors [12]). In this context, the estimated perceived information will exactly correspond to the (worst-case) estimated mutual information.

Information theoretic metrics such as the mutual information and the perceived information are especially interesting for the comparison of profiled side-channel attacks as we envision here. This is because they can generally be estimated based on a single plaintext (i.e., with $N_a = 1$), whereas the success rate is generally estimated for varying N_a's. In other words, their scalar value provides a very similar intuition as the success rate curves [35]. Unfortunately, the estimation of information theoretic metrics requires distinguishers providing probabilities, which is not the case of machine learning-based attacks.[3] As a result, our concrete experiments comparing template attack, support vector machine and random forest will be based on estimations of the success rate for a number of representative parameters.

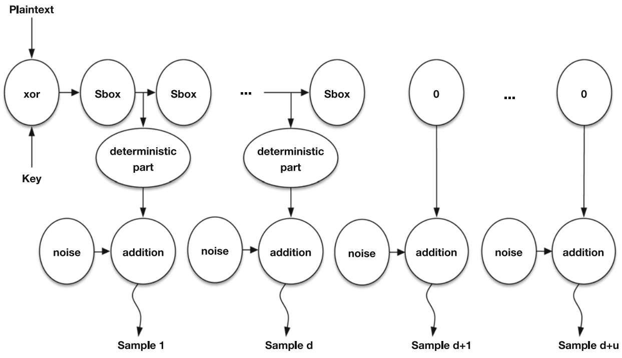

Fig. 2 Simulated leaking implementations

3 Perfect profiling

In this section, we study the impact of useless samples in leakage traces on the performances of template attack with perfect profiling (i.e., the evaluator perfectly knows the leakages' probability density function). In this context, we will use Pr for both $\Pr_{\text{model}}$ and $\Pr_{\text{chip}}$ (since they are equal) and omit subscripts for the leakages l to lighten notations.

In Proposition 1, we aim to show that in case of perfect profiling, the detection of points of interests is not necessary for a template attack, since useless points will not have any impact on the attack's success. Since template attacks are optimal from an information theoretic point of view, it also means that the machine learning-based approaches cannot be more efficient in this context.

Proposition 1 *Let us assume two template attacks with perfect models using two different attack traces l_1 and l_2 associated with the same plaintext x: l_1 is composed of d samples providing information and $l_2 = [l_1 || \epsilon]$ (where $\epsilon = [\epsilon_1, \ldots, \epsilon_u]$ represents noise variables independent*

[2] In [32] the equation representing the perceived information has a minus sign, whereas the correct sign is positive.

[3] There are variants of SVM and RF that aim to remedy to this issue. Yet, the "probability-like" scores they output are not directly exploitable in the estimation of information theoretic metrics. For example, we could exhibit examples where probability-like scores of one do not correspond to a success rate of one. More recently, Choudary et al. [6] showed that key enumeration based on scores and based on probabilities provide different results, which highlights the difference between score-based and probability-based profiled attacks.

of l_1 and the key). Then, the mutual information leakage $\mathrm{MI}(K; X, L)$ *estimated with their (perfect) leakage models is the same.*

Proof As clear from the definitions in Sect. 2.6, the mutual/perceived information estimated thanks to template attack only depend on $\Pr[k|l]$. So we need to show that these conditional probabilities $\Pr[k|l_2]$ and $\Pr[k|l_1]$ are equal. Let k and k' represent two key guesses. Since ϵ is independent of l_1 and k, we have:

$$\begin{aligned}\frac{\Pr[l_2|k']}{\Pr[l_2|k]} &= \frac{\Pr[l_1|k'] \cdot \Pr[\epsilon|k']}{\Pr[l_1|k] \cdot \Pr[\epsilon|k]}\\ &= \frac{\Pr[l_1|k'] \cdot \Pr[\epsilon]}{\Pr[l_1|k] \cdot \Pr[\epsilon]}\\ &= \frac{\Pr[l_1|k']}{\Pr[l_1|k]}. \end{aligned} \tag{7}$$

This directly leads to:

$$\begin{aligned}\frac{\sum_{k'\in\mathcal{K}} \Pr[l_2|k']}{\Pr[l_2|k]} &= \frac{\sum_{k'\in\mathcal{K}} \Pr[l_1|k']}{\Pr[l_1|k]},\\ \frac{\Pr[l_2|k]}{\sum_{k'\in\mathcal{K}} \Pr[l_2|k']} &= \frac{\Pr[l_1|k]}{\sum_{k'\in\mathcal{K}} \Pr[l_1|k']},\\ \Pr[k|l_2] &= \Pr[k|l_1], \end{aligned} \tag{8}$$

which concludes the proof. □

Quite naturally, this proof does not hold as soon as there are dependencies between the d first samples in l_1 and the u latter ones. This would typically happen in contexts where the noise at different time samples is correlated (which could then be exploited to improve the attack).

Note that the main reason why we need a perfect model for the result to hold is that we need the independence between the informative and non-informative samples to be reflected in these models as well. For example, in the case of Gaussian templates, we need the covariance terms that corresponds to the correlation between informative and non-informative samples to be null (which will not happen for imperfectly estimated templates). In fact, the result would also hold for imperfect models, as long as these imperfections do not suggest significant correlation between these informative and non-informative samples. But of course, we could not state that template attacks necessarily perform better than machine learning-based attacks in this case. Overall, this conclusion naturally suggests a more pragmatic question. Namely, perfect profiling never occurs in practice. So how does this theoretic intuition regarding the curse of dimensionality for template attack extends to concrete profiled attack (with bounded profiling phases)? We study it in the next section.

4 Experiments with imperfect profiling

We now consider examples of template attack, support vector machine and random forest-based attacks in order to gain intuition about their behavior in concrete profiling conditions. As detailed in Sect. 2, we will use a simulated experimental setting with various number of informative and uninformative samples in the leakage traces for this purpose.

4.1 Nearly perfect profiling

As a first experiment, we considered the case where the profiling is "sufficient"—which should essentially confirm the result of Proposition 1. For this purpose, we analyzed simulated leakage traces with 2 informative points (i.e., $d = 2$), $u = 0$ and $u = 15$ useless samples, and an SNR of 1, in function of the number of traces per intermediate value in the profiling set N_p. As illustrated in Fig. 3, we indeed observe that (e.g.) the perceived information is independent of u if the number of traces in the profiling set is "sufficient" (i.e., all attacks converge toward the same perceived information in this case). By contrast, we notice that this "sufficient" number depends on u (i.e., the more useless samples, the larger N_p needs to be). Besides, we also observe that the impact of increasing u is stronger for naive template attack than efficient template attack, since the first one has to deal with a more complex estimation. Indeed, the efficient template attack has 256 times more traces than the naive template attack to estimate the covariance matrice. So overall, and as expected, as long as the profiling set is large enough and the assumptions used to build the model capture the leakage samples sufficiently accurately, template attacks are indeed optimal, independent of the number of samples they actually profile. So there is little gain to expect from machine learning-based approaches in this context.

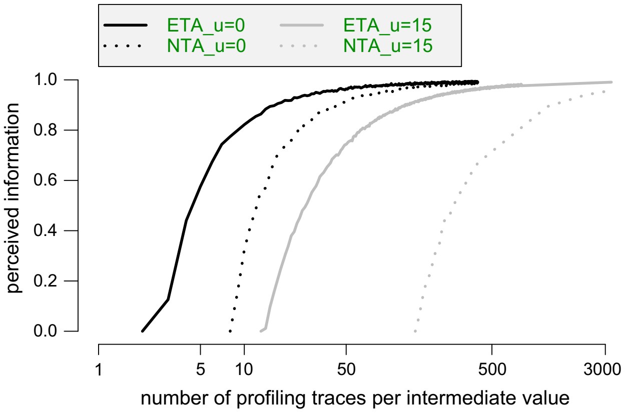

Fig. 3 Perceived information for naive template attack (NTA) and efficient template attack (ETA) in function of N_p with SNR = 1

4.2 Imperfect profiling

We now move to the more concrete case where profiling is imperfect. In our simulated setting, imperfections naturally arise from limited profiling (i.e., estimation errors): We will investigate their impact next and put forward some useful intuitions regarding the curse of dimensionality in (profiled) side-channel attacks. Yet, we note that in general, assumption errors can also lead to imperfect models, that are more difficult to deal with (see, e.g., [12]). For example, in a first-order masking implementation, when the adversary does not know the mask values during the profiling step, the leakages associated with a key value follow a multimodal distribution. This context leads to assumption errors whether the adversary exploits Gaussian template attacks. Note however that, in our context, template attacks have no assumption error.

Besides, and as already mentioned, since we now want to compare template attack, support vector machine and random forest, we need to evaluate and compare them with security metrics (since the two latter ones do not output the probabilities required to estimate information theoretic metrics).

In our first experiment, we set again the number of useful dimensions to $d = 2$ and evaluated the success rate of the different attacks in function of the number of non-informative samples in the leakages traces (i.e., u), for different sizes of the profiling set. As illustrated in Fig. 4, we indeed observe that for a sufficient profiling, efficient template attack is the most efficient solution. Yet, it is also worth observing that naive template attack provides the worst results overall, which already suggests that comparisons are quite sensitive to the adversary/evaluator's assumptions. Quite surprisingly, our experimental results show that up to a certain level, the success rate of random forest increases with the number of points without information. The reason is intrinsic to the random forest algorithm in which the trees need to be as decorrelated as possible. As a result, increasing the number of points in the leakage traces leads to a better independence between trees and improves the success rate. Besides, the most interesting observation relates to random forest in high dimensionality, which remarkably resists the addition of useless samples (compared to support vector machine and template attack). The main reason for this behavior is the random feature selection embedded into this tool. That is, for a sufficient number of trees, random forest eventually detects the informative points of interests in the traces, which makes it less sensitive to the increase in u. By contrast, template attack and support vector machine face a more and more difficult estimation problem in this case.

Another noticeable element of Fig. 4 is that support vector machine and random forest seem to be bounded to lower success rates than template attack. But this is mainly an artifact of using the success rate as evaluation metric. As illustrated in Fig. 5, increasing either the number of informative dimensions in the traces d or the number of attack traces N_a leads to improved success rates for the machine learning-based

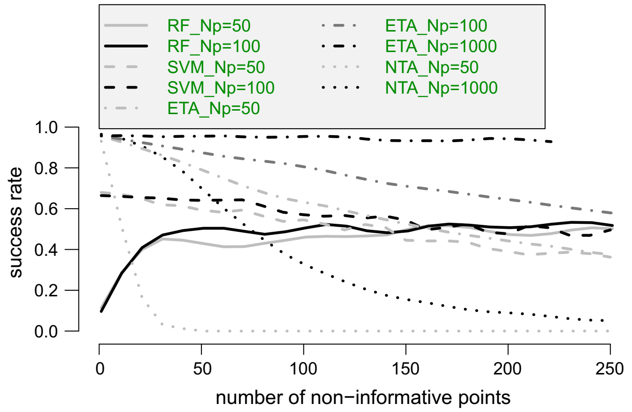

Fig. 4 Success rate for naive template attack (NTA), efficient template attack (ETA), support vector machine (SVM) and random forest (RF) in function of the number of useless samples u, for various sizes of the profiling set N_p, with $d = 2$, SNR = 1, $N_a = 15$

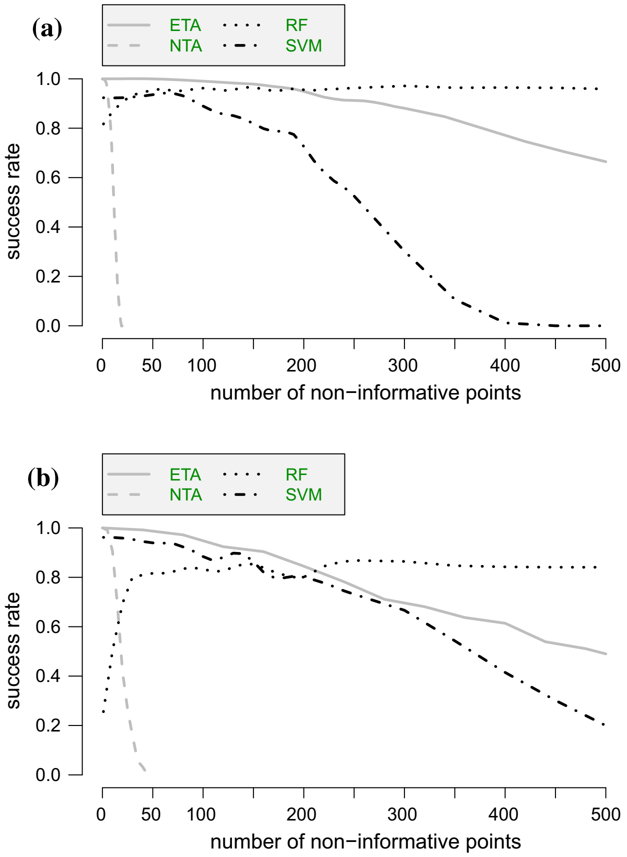

Fig. 5 **a** Success rate for naive template attack (NTA), efficient template attack (ETA), support vector machine (SVM) and random forest (RF) in function of the number of useless samples u, with parameters $N_p = 25$, $d = 5$, SNR = 1 and $N_a = 15$. **b** Similar experiment with parameters $N_p = 50$, $d = 2$, SNR = 1 and $N_a = 30$

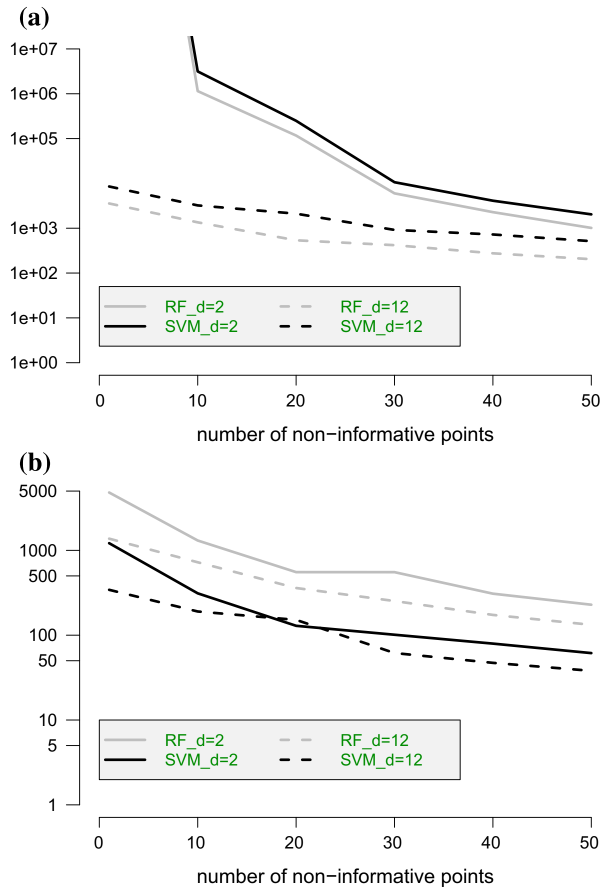

Fig. 6 Time complexity for efficient template attack (ETA), support vector machine (SVM) and random forest (RF) in function of the number of useless samples, for $d \in \{2, 12\}$ and $N_p = 25$. **a** Profiling time relatively to TA as a function of the number of non-informative points. **b** Attacking time relatively to TA as a function of the number of non-informative points

approaches as well. For the rest, the latter figure does not bring significantly new elements. We essentially notice that random forest becomes interesting over efficient template attack for very large number of useless dimensions and that efficient template attack is most efficient otherwise.

Eventually, the interest of the random feature selection in random forest-based models raises the question of the time complexity for these different attacks. That is, such a random feature selection essentially works because there is a large enough number of trees in our random forest models. But increasing this number naturally increases the time complexity of the attacks. For this purpose, we report some results regarding the time complexity of our attacks in Fig. 6. As a preliminary note, we mention that those results are based on prototype implementations in different programming languages (C for template attack, R for support vector machine and random forest). So they should only be taken as a rough indication. Essentially, we observe an overhead for the time complexity of machine learning-based attacks, which vanishes as the size of the leakage traces increases. Yet, and most importantly, this overhead remains comparable for support vector machine and random forest in our experiments (mainly due to the fact that the number of trees was set to a constant 500). So although the computational cost of these attacks is not negligible, it remains tractable for the experimental parameters we considered (and could certainly be optimized in future works).

5 Bias–variance decomposition analysis

The goal of this section is to understand more deeply (i) why template attack can have a higher success rate than machine learning-based attack in a low dimensionality context, and (i) why a random forest outperforms template attack in a high dimensionality context. Our analyzes are based on the bias–variance decomposition of the error rate first proposed by Domingos in the field of machine learning [10,11] and then introduced in the side-channel literature by Lerman et al. [25].

5.1 Background

Domingos showed that the error rate of a model can be decomposed in three weighted components [10,11]: the error rate of the Bayes classifier $\mathrm{ER}_b(\cdot)$ (defined in this section and also known as the noise term in the machine learning field), the bias $\mathrm{B}(\cdot)$ and the variance $\mathrm{V}(\cdot)$, generally leading to the equality:

$$\begin{aligned}\text{Error rate} = {} & E_{\mathcal{L}_{\mathrm{AS}}}\left[c_1 \times \mathrm{ER}_b(\mathcal{L}_{\mathrm{AS}})\right] \\ & + E_{\mathcal{L}_{\mathrm{AS}}}\left[\mathrm{B}(\mathcal{L}_{\mathrm{AS}})\right] \qquad \text{Bias} \\ & + E_{\mathcal{L}_{\mathrm{AS}}}\left[c_2 \times \mathrm{V}(\mathcal{L}_{\mathrm{AS}})\right], \qquad \text{Variance} \qquad (9)\end{aligned}$$

where $\{c_1, c_2, \mathrm{ER}_b(\mathcal{L}_{\mathrm{AS}}), \mathrm{B}(\mathcal{L}_{\mathrm{AS}}), \mathrm{V}(\mathcal{L}_{\mathrm{AS}})\} \in \mathbb{R}^5$, and $\mathcal{L}_{\mathrm{AS}}$ represents a set of attack traces.

In order to implement the bias–variance decomposition, we first need a Bayes classifier [denoted $\mathrm{A}_b(\cdot)$] which represents the best model that an adversary can build (i.e., a model with no estimation nor assumption errors). More formally, the Bayes classifier minimizes the probability of misclassification:

$$\mathrm{A}_b(\mathcal{L}_{\mathrm{AS}}) = \underset{k^*}{\operatorname{argmax}} \Pr\left[\mathcal{L}_{\mathrm{AS}} \mid k^*\right] \times \Pr\left[k^*\right]. \qquad (10)$$

Next, the loss function $\mathrm{L}(k, k')$ represents the cost of predicting k' when the true target value is k. In this paper, we consider the zero-one loss function: The cost is zero when k equals k' and one in the other cases.

Intuitively, the error rate of the Bayes classifier represents the unavoidable component of the error rate, i.e., the minimum error rate of a model. More formally, the error rate of the Bayes classifier equals to:

$$\mathrm{ER}_b(\mathcal{L}_{\mathrm{AS}}) = \mathrm{L}(k, \mathrm{A}_b(\mathcal{L}_{\mathrm{AS}})). \tag{11}$$

Let now $\mathrm{A}_m(\mathcal{L}_{\mathrm{AS}})$ be the *main prediction* that represents the most frequent prediction on the set of attack traces $\mathcal{L}_{\mathrm{AS}}$ given by the estimated model when varying the profiling set. The bias term represents the difference (according to the loss function) between the main prediction and the prediction provided by the Bayes classifier. Mathematically the bias term equals:

$$\mathrm{B}(\mathcal{L}_{\mathrm{AS}}) = \mathrm{L}(\mathrm{A}_m(\mathcal{L}_{\mathrm{AS}}), \mathrm{A}_b(\mathcal{L}_{\mathrm{AS}})). \tag{12}$$

The variance term then measures the variation of a prediction on a set of attack traces as a function of different profiling sets. Mathematically, the variance term equals:

$$\mathrm{V}(\mathcal{L}_{\mathrm{AS}}) = E_{\mathcal{L}_{\mathrm{PS}}}\left[\mathrm{L}(\mathrm{A}_m(\mathcal{L}_{\mathrm{AS}}), \mathrm{A}(\mathcal{L}_{\mathrm{AS}}, \mathcal{L}_{\mathrm{PS}}))\right], \tag{13}$$

where $\mathcal{L}_{\mathrm{PS}}$ is a set of profiling traces and $\mathrm{A}(\mathcal{L}_{\mathrm{AS}}, \mathcal{L}_{\mathrm{PS}})$ is the prediction of the estimated model based on the profiling set $\mathcal{L}_{\mathrm{PS}}$ and the attacking set $\mathcal{L}_{\mathrm{AS}}$.

Based on these notations, Domingos demonstrated that the multiplicative factors c_1 and c_2 equal:

$$\begin{aligned} c_1 &= \Pr[\mathrm{A} = \mathrm{A}_b] \\ &\quad - \Pr[\mathrm{A} \neq \mathrm{A}_b] \times \Pr[\mathrm{A} = k \mid \mathrm{A}_b \neq k], \end{aligned} \tag{14}$$

$$c_2 = \begin{cases} -\Pr[\mathrm{A} = \mathrm{A}_b \mid \mathrm{A} \neq \mathrm{A}_m] & \mathrm{A}_m \neq \mathrm{A}_b \\ 1 & \mathrm{A}_m = \mathrm{A}_b \end{cases}, \tag{15}$$

where $\mathrm{A} = \mathrm{A}(\mathcal{L}_{\mathrm{AS}})$, $\mathrm{A}_b = \mathrm{A}_b(\mathcal{L}_{\mathrm{AS}})$ and $\mathrm{A}_m = \mathrm{A}_m(\mathcal{L}_{\mathrm{AS}})$.

5.2 Template attack

Recently, Lerman et al. [25] showed that template attacks have a high variance while stochastic attack correspond to a trade-off between the bias and the variance terms. In this section, we aim to evaluate when and why template attacks generally worked well in our previous experiments, and machine learning algorithm (and more precisely random forests) can outperform them in extreme profiling conditions.[4]

Our first experiment aims to recall the effect of the leakage function on the error rate of template attack. We use 10×256 traces in the profiling set, 10 informative points per trace, 1 attacking trace and an SNR of 1. We consider two leakage functions (one linear and one random) representing two bijective functions providing the same amount of information per leakage but different dependency between the leakages and the target values. The purpose is to show the error rate, the bias and the variance of template attacks. Figure 7 clarifies that the success rate of template attack is independent of the leakage function (as already put forward by Lerman et al. [25]). More precisely, template attack has a high variance and a low bias, confirming the high(er) complexity of the model leading template attacks to be able to represent any kind of dependency between the target value and the leakage function.

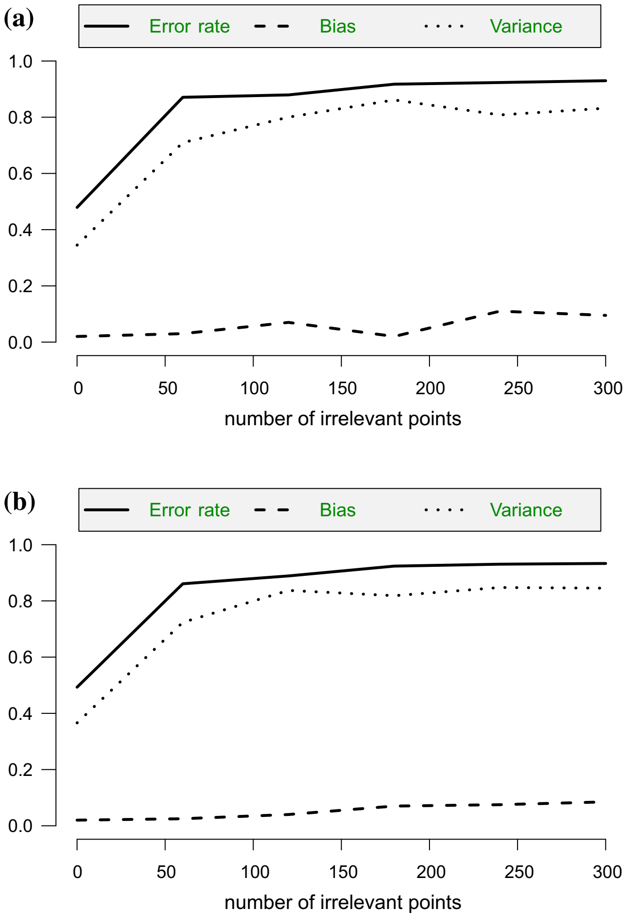

Fig. 7 Error rate, bias and variance of a template attack as a function of the number of irrelevant points per trace where $u \in \{0, 60, 120, 180, 240, 300\}$. There are 10×256 traces in the profiling set, 10 informative points per trace, 1 attacking trace, an SNR of 1, and the leakage function is linear (**a**) and random (**b**)

In order to reduce the variance of template attacks, we need to increase the size of the profiling set or to use a stochastic attack with a low degree. The first strategy keeps the bias low while the second may increase the bias. This phenomenon led us to consider the first approach as an additional illustration of our previous conclusions. Figure 8 shows what happens when

[4] By contrast, we do not discuss the impact on the bias and on the variance term of each meta-parameter of a random forest and a template attack. For the interested readers about this aspect, we refer to the document of Louppe [28] analyzing random forests and to the paper of Lerman et al. [25] analyzing template attack.

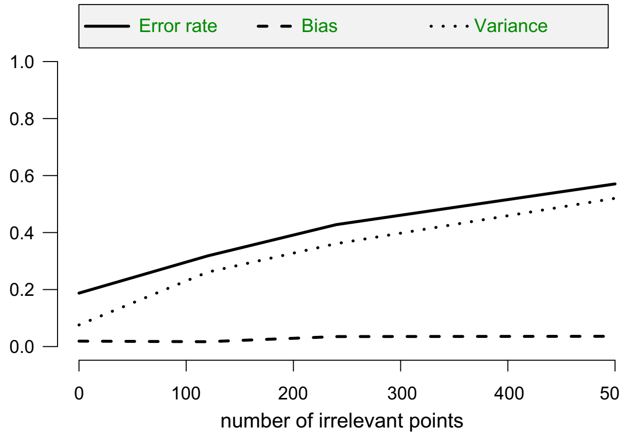

Fig. 8 Error rate, bias and variance of a template attack as a function of the number of irrelevant points per trace where $u \in \{0, 120, 240, 500\}$. There are 30 × 256 traces in the profiling set, 20 informative points per trace, a random leakage, 1 attacking trace, and an SNR of 1

we increase the number of traces in the profiling set and the number of informative points. As expected, template attacks have reduced variance as well as error rate when increasing the number of traces in the profiling set and when increasing the number of informative points. The two previous results suggest that machine learning algorithms could gain interest if they have (1) a lower variance term compared to template attacks and (2) still maintaining a sufficiently low bias term (as template attack) allowing to obtain successful key recoveries.

5.3 Random forests

In general, the main advantage of template attacks as a profiling method is the possibility to target complex leakage functions. Our first experiment on random forest aims to verify whether random forest enjoys the same ability. Figure 9 plots the error rate, the bias and the variance of a random forest with 10 × 256 traces in the profiling set, 10 informative points per trace, 1 attacking trace and an SNR of 1. The figure shows that random forests are indifferent to changes in the leakage function (similarly to template attacks). Moreover, and as previously, we observe that random forests outperform template attacks in very high dimensionality contexts (see Table 1 that summarizes the results of template attack and random forest). More precisely, the higher the number of irrelevant points, the higher the error rate for both models. Interestingly, the error rate of template attacks is mainly due to a high variance while random forests seek to minimize this variance term thanks to its bagging approach. So the bias–variance decomposition here allows understanding the complementary nature of these techniques.

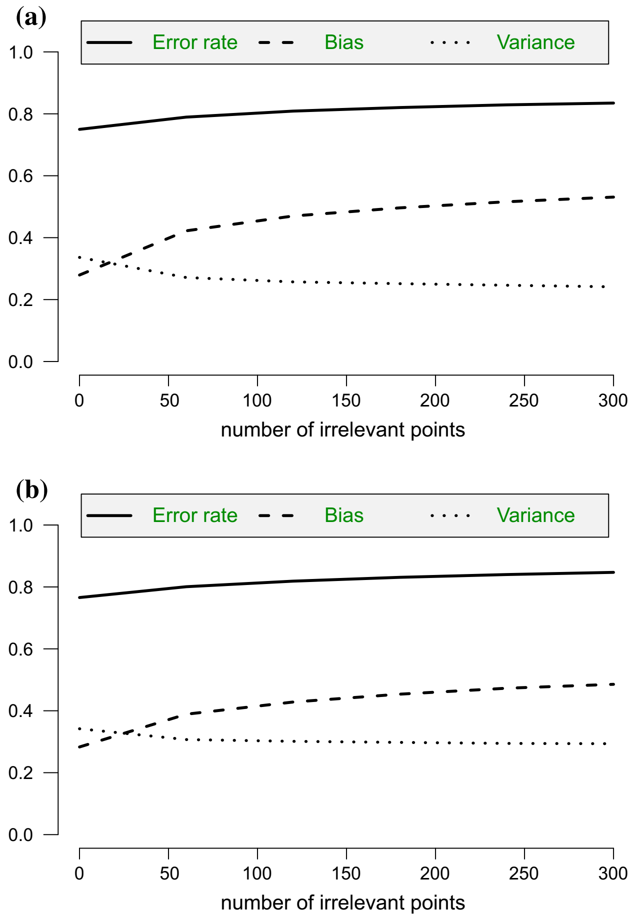

Fig. 9 Error rate, bias and variance of a random forest as a function of the number of irrelevant points per trace where $u \in \{0, 60, 120, 180, 240, 300\}$. There are 10 × 256 traces in the profiling set, 10 informative points per trace, an SNR of 1, and the leakage function is linear (in the *left*) and random (in the *right*). **a** Linear leakage function, **b** random leakage function

Figure 10 shows additional results when we increase the size of the profiling set as well as the number of informative points. This new setting allows to reduce the variance and the bias of a random forest. Table 2 summarizes the results of template attacks and random forests in this new context. Once again, this experiment highlights that the latter ones gain interest in high dimensionality contexts. Moreover, the increase in the number of irrelevant points has a lower impact on the error rate of random forest compared to the error rate of template attack. More precisely, the increase in the number of irrelevant points impacts less the variance term of random forests compared to the variance term of template attacks. Interestingly, this discussion also allows to understand other previous results obtained in the profiled side-channel attacks literature [1,3,15–19,22–24,26,27,30,31].

Table 1 Error rate of several profiled attacks as a function of the number of irrelevant points per trace

	$u = 0$	$u = 60$	$u = 120$	$u = 180$	$u = 240$	$u = 300$
Template attack	0.49	0.86	0.89	0.92	0.93	0.93
Random forest	0.77	0.80	0.82	0.83	0.84	0.85

There are 10 × 256 traces in the learning set, 10 informative points per trace, a random leakage, 1 attacking trace and an SNR of 1

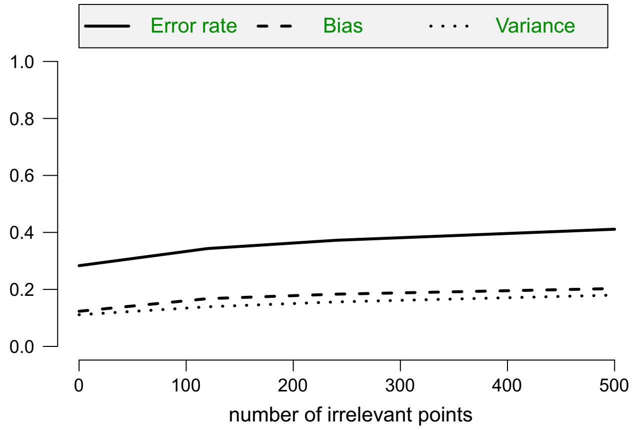

Fig. 10 Error rate, bias and variance of a random forest as a function of the number of irrelevant points per trace where $u \in \{0, 120, 240, 500\}$. There are 30 × 256 traces in the profiling set, 20 informative points per trace, a random leakage, 1 attacking trace, and an SNR of 1

Table 2 Error rate of several profiled attacks as a function of the number of irrelevant points per trace

	$u = 0$	$u = 120$	$u = 240$	$u = 500$
Template attack	0.18	0.31	0.42	0.57
Random forest	0.28	0.34	0.37	0.41

There are 30 × 256 traces in the learning set, 20 informative points per trace, a random leakage, 1 attacking trace and an SNR of 1

6 Conclusion

Our results provide interesting insights on the curse of dimensionality for side-channel attacks. From a theoretic point of view, we first showed that as long as a limited number of points of interests can be identified in leakage traces and contain most of the information, template attacks are the method of choice. Such a conclusion extends to any scenario where the profiling can be considered as "nearly perfect." By contrast, we also observed that as the number of useless samples in leakage traces increases and/or the size of the profiling set becomes too limited, machine learning-based attacks gain interest. In our simulated setting, the most interesting gain is exhibited for random forest-based models, thanks to their random feature selection. These observations nicely fit to the observations made by Banciu et al. in a different context, namely simple power analysis and algebraic side-channel analysis [2]. Our additional analyzes based on the bias–variance decomposition also allow re-stating these observations in more formal terms. That is, template attacks are the method of choice as long as the variance term is low, while machine learning algorithms or linear regression (that can have a lower variance term than template attack) should be used in high dimensionality contexts.

Admittedly, the simulated setting we investigated is probably most favorable to template attacks, since only estimation errors can decrease the accuracy of the adversary/evaluator models in this case. One can reasonably expect that real devices with harder to model noise distributions would improve the interest of machine learning techniques compared to efficient template attacks—as has been suggested in previously published works. As a result, the extension of our experiments toward other distributions is an interesting avenue for further research. In particular, the study of leakage traces with correlated noise could be worth additional investigations in this respect. Eventually, the bias–variance decomposition of other profiled attacks (e.g., based on support vector machine and neural networks) represents future work.

In summary, template attacks are the most efficient strategies for well-understood devices, with sufficient profiling, as they can approach the worst-case security level of an implementation in such context. By contrast, machine learning-based attacks (especially random forest) are promising alternative(s) in black box settings, with only limited understanding of the target implementation and in high dimensionality contexts.

Acknowledgements F.-X. Standaert is a research associate of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in parts by the European Commission through the ERC project 280141 (CRASH). L. Lerman is a postdoctoral researcher working on the SCAUT project and funded by the Brussels Institute for Research and Innovation (Innoviris).

References

1. Banciu, V., Oswald, E., Whitnall, C.: Reliable information extraction for single trace attacks. In: Nebel, W., Atienza, D. (eds.) Proceedings of the 2015 Design, Automation and Test in Europe Conference and Exhibition, DATE 2015, Grenoble, France, March 9–13, 2015, pp. 133–138. ACM (2015)

2. Banciu, V., Oswald, E., Carolyn, W.: Reliable information extraction for single trace attacks. In: IACR Cryptology ePrint Archive, vol. 2015, p. 45 (2015)
3. Bartkewitz, T., Lemke-Rust, K.: Efficient template attacks based on probabilistic multi-class support vector machines. In: Mangard, S. (ed.) CARDIS, volume 7771 of Lecture Notes in Computer Science, pp. 263–276. Springer, Berlin (2012)
4. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
5. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES, volume 2523 of Lecture Notes in Computer Science, pp. 13–28. Springer, Berlin (2002)
6. Choudary, M.O., Poussier, R., Standaert, F.-X.: Score-based vs. probability-based enumeration—a cautionary note. In: Progress in Cryptology—INDOCRYPT 2016—17th International Conference on Cryptology in India, Kolkata, India, December 11–14, 2016, Proceedings (2016) (to appear)
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. Lecture Notes in Computer Science, vol. 8419, pp. 253–270. Springer (2013)
8. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
9. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge (2010)
10. Domingos, P.: A unifeid bias-variance decomposition and its applications. In: Langley, P. (ed.) Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29–July 2, 2000, pp. 231–238. Morgan Kaufmann (2000)
11. Domingos, P.: A unified bias-variance decomposition for zero-one and squared loss. In Kautz, H.A., Porter, B.W. (eds.) Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, July 30–August 3, 2000, Austin, Texas, USA, pp. 564–569. AAAI Press/The MIT Press (2000)
12. Durvaux, F., Standaert, F.-X., Veyrat-Charvillon, N.: How to certify the leakage of a chip? In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT, volume 8441 of Lecture Notes in Computer Science, pp. 459–476. Springer, Berlin (2014)
13. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: concrete results. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES, volume 2162 of Lecture Notes in Computer Science, pp. 251–261. Springer, Berlin (2001)
14. Gierlichs, B., Lemke-Rust, K., Paar, C.: Templates vs. stochastic methods. In: Goubin, L., Matsui, M. (eds.) CHES, volume 4249 of Lecture Notes in Computer Science, pp. 15–29. Springer, Berlin (2006)
15. Gilmore, R., Hanley, N., O'Neill, M.: Neural network based attack on a masked implementation of AES. In: IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2015, Washington, DC, USA, 5–7 May, 2015, pp. 106–111. IEEE (2015)
16. He, H., Jaffe, J., Zou, L.: CS 229 Machine Learning—Side Channel Cryptanalysis Using Machine Learning. Technical Report, Stanford University (2012)
17. 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 Lecture Notes in Computer Science, pp. 249–264. Springer, Berlin (2012)
18. Hospodar, G., Gierlichs, B., De Mulder, E., Verbauwhede, I., Vandewalle, J.: Machine learning in side-channelanalysis: a first study. J. Cryptogr. Eng. **1**(4), 293–302 (2011)
19. Hospodar, G., De Mulder, E., Gierlichs, B., Vandewalle, J., Verbauwhede, I.: Least squares support vector machines for side-channel analysis. In: Second International Workshop on Constructive Side-Channel Analysis and Secure Design, pp. 99–104. Center for Advanced Security Research Darmstadt (2011)
20. Kocher, P.C.: Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In: Koblitz, N. (ed.) CRYPTO, volume 1109 of Lecture Notes in Computer Science, pp. 104–113. Springer, Berlin (1996)
21. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M.J. (ed.) CRYPTO, volume 1666 of Lecture Notes in Computer Science, pp. 388–397. Springer, Berlin (1999)
22. Lerman, L., Bontempi, G., Markowitch, O.: Side-channel attacks: an approach based on machine learning. In: Second International Workshop on Constructive Side-Channel Analysis and Secure Design, pp. 29–41. Center for Advanced Security Research Darmstadt (2011)
23. Lerman, L., Bontempi, G., Markowitch, O.: Power analysis attack: an approach based on machine learning. IJACT **3**(2), 97–115 (2014)
24. Lerman, L., Bontempi, G., Markowitch, O.: A machine learning approach against a masked AES. J. Cryptogr. Eng. **5**(2), 123–139 (2015)
25. Lerman, L., Bontempi, G., Markowitch, O.: The bias–variance decomposition in profiled attacks. J. Cryptogr. Eng. **5**, 1–13 (2015)
26. Lerman, L., Medeiros, S.F., Bontempi, G., Markowitch, O.: A machine learning approach against a masked AES. 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. Lecture Notes in Computer Science, vol. 8419, pp. 61–75. Springer (2013)
27. Lerman, L., Poussier, R., Bontempi, G., Markowitch, O., Standaert, F.-X.: Template attacks vs. machine learning revisited (and the curse of dimensionality in side-channel analysis). In: Mangard, S., Poschmann, A.Y. (eds.) Constructive Side-Channel Analysis and Secure Design—6th International Workshop, COSADE 2015, Berlin, Germany, April 13–14, 2015. Revised Selected Papers, volume 9064 of Lecture Notes in Computer Science, pp. 20–33. Springer (2015)
28. Louppe, G.: Understanding Random Forests: From Theory to Practice. ArXiv e-prints (2014)
29. Mangard, S., Oswald, E., Standaert, F.-X.: One for all–all for one: unifying standard differential power analysis attacks. IET Inf. Secur. **5**(2), 100–110 (2011)
30. Martinasek, Z., Hajny, J., Malina, L.: Optimization of power analysis using neural network. 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 Lecture Notes in Computer Science, pp. 94–107. Springer (2013)
31. Patel, H., Baldwin, R.O.: Random forest profiling attack on advanced encryption standard. IJACT **3**(2), 181–194 (2014)
32. Renauld, M., Standaert, F.-X., Veyrat-Charvillon, N., Kamel, D., Flandre, D.: A formal study of power variability issues and side-channel attacks for nanoscale devices. In: Paterson, K.G. (ed.) EUROCRYPT, volume 6632 of Lecture Notes in Computer Science, pp. 109–128. Springer, Berlin (2011)
33. Rokach, L., Maimon, O.: Data Mining with Decision Trees: Theory and Applications. Series in Machine Perception and Artificial Intelligence. World Scientific Publishing Company, Incorporated, Singapore (2008)
34. Schindler, W., Lemke, K., Paar, C.: A stochastic model for differential side channel cryptanalysis. In: Rao, J.R., Sunar, B. (eds.) CHES, volume 3659 of Lecture Notes in Computer Science, pp. 30–46. Springer, Berlin (2005)
35. Standaert, F.-X., Koeune, F., Schindler, W.: How to compare profiled side-channel attacks? In: Abdalla, M., Pointcheval, D.,

Fouque, P.-A., Vergnaud, D. (eds.) ACNS. Lecture Notes in Computer Science, vol. 5536, pp. 485–498. Springer, Berlin (2009)

36. Standaert, F.-X., Malkin, T., Yung, M.: A unified framework for the analysis of side-channel key recovery attacks. In: Joux, A. (ed.) EUROCRYPT, volume 5479 of Lecture Notes in Computer Science, pp. 443–461. Springer, Berlin (2009)

37. 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, pp. 390–406. Springer, Berlin (2012)