Support Vector Machines for Improved IP Detection with Soft Physical Hash Functions

Ludovic Gustin[1,2], François Durvaux[1], Stéphanie Kerckhof[1], François-Xavier Standaert[1] and Michel Verleysen[2]

[1] Crypto Group - ICTEAM, Université catholique de Louvain, Belgium.
[2] Machine Learning Group - ICTEAM, Université catholique de Louvain, Belgium.

Abstract. Side-channel analysis is a powerful tool to extract secret information from microelectronic devices. Its most frequently considered application is destructive, i.e. key recovery attacks against cryptographic implementations. More recently, it has also been considered constructively, in the context of intellectual property protection/detection, e.g. through the use of side-channel based watermarks or soft physical hash functions. The latter solution is interesting from the application point-of-view, because it does not require any modification of the designs to protect (hence it implies no performance losses). Previous works in this direction have exploited simple (correlation-based) statistical tools in different (more or less challenging) scenarios. In this paper, we investigate the use of support vector machines for this purpose. We first argue that their single-class extension is naturally suited to the problem of intellectual property detection. We then show experimentally that they allow dealing with more complex scenarios than previously published, hence extending the relevance and applicability of soft physical hash functions.

1 Introduction

Protecting Intellectual Properties (IP) from illegal use is an important issue for the development of markets based on third party designs (next refered to as IP cores). Different solutions have been proposed to mitigate this problem, among which permission-based and watermarking-based techniques are usual candidates. The first one consists in checking whether the system has the right permission before performing any operation (i.e. works *a priori*). Most common solutions are implemented with an enhanced security chip or a Physically Unclonable Function (PUF) that contain some secret [5, 17, 28, 38]. If the IP gets the right answer to a defined challenge, it means that it is used properly and can start processing. The second family (i.e. watermarking-based protections) consists in hiding a piece a information for authentication or identification in the IP, that is recovered by its owner(s) if needed [3, 22, 23, 27]. The inserted information must be robust to noise and to slight transformations that may occur in the IP manipulation. It must also be invisible to the adversary. Recently, it has been proposed to place the mark in physical features such as the temperature, power consumption, ... of the device on which the IP is executed [7, 42]. In opposition to the permission-based mechanism that prevents illegal use of IP from running, watermarks can only detect this illegal use (i.e. they work *a posteriori*).

As an alternative to these proposals, Soft Physical Hash (SPH) functions are an *a posteriori* solution that also exploits information extracted from the physical features of a target circuit. The difference with the watermarking-based solution lies in the fact that the information extracted comes from the very characteristics of the implementation, and is not inserted by the IP owner. Therefore, and in contrast with both previous families, this solution does not need any piece of hardware to be added, and it cannot be removed or altered as it depends on the IP itself. SPH functions have been formalized together with IP detection infrastructures in [15], where the requirements for this solution to be effective (namely perceptual robustness and content sensitivity) have also been defined. They have first been applied in a simple case study of 8-bit software implementations, and next in the more challenging context of FPGA implementations [24].

While these previous works can be seen as encouraging proofs-of-concept, and validate the idea that SPH can be useful components in the detection of IP theft, the main question naturally remains to know how robust it is against challenging adversarial conditions. For example, can it be effective without knowing the inputs/outputs of the IP to detect?, how does it resist against re-compiled/re-synthetized IPs?, and (in the most interesting case of hardware IP) how does it react to other parasitic IP running in parallel (i.e. when included in a larger system combining several proprietary designs)? In particular, the work in [24] suggested that simple instances of SPH (where the detection procedure is based on Pearson's correlation-based statistics) start to encounter some failures in the context of re-synthetised FPGA designs with parasitics IP running in parallel. In this work, we aim to complement these first results, and consider Support Vector Machines (SVM) to enhance IP detection in such complex scenarios. For comparison purposes, we consider the same case study as [24] (i.e. six FPGA implementations of block ciphers) and show experimentally that SVM lead to significant (perceptual robustness and content sensitivity) improvements.

Why SVM? As IP detection infrastructures based on SPH functions exploit standard techniques from side-channel analysis, one can wonder why SVM are prefered to other more standard tools that usually improve over correlation-based statistics (e.g. templates [12] or stochastic approaches [35]). Before entering the core of the paper, we provide a brief argument in this respect. First, one can notice that SVM have been shown to provide an interesting alternative to such tools (see, e.g. [6, 18, 20, 26]). So while it is unclear that they generally provide significant advantages over other distinguishers (especially in the context of "standard" first-order side-channel attacks where many statistics are equivalent to some extent [30]), they are at least expected to work reasonably in this context too. Second and more importantly, the problem of IP detection differs from the one of key recovery in one important aspect. Namely, the number of classes in key recovery is usually well identified (e.g. 256 when targeting the AES S-box output), while is is not enumerable in IP detection. Indeed, the IP owner can only characterize his own design, and the number and shape of suspicious IP is a priori unknown. As a result, the single-class extension of SVM appears to be naturally suited to this context (since it only requires the knowledge of the

reference IP during its characterization). Eventually, SVM are particularly interesting for dealing with large dimensionalities (e.g. large measurement traces in side-channel attacks), which contrasts with templates and stochastic approaches that work best if a good points-of-interest are identified. But the selection of such points-of-interest usually relies on criteria such as the signal-to-noise ratio, that (ideally) requires the knowledge of the different classes to discriminate (since they define the signal). So the fact that we do not need to find points-of-interest in SVM is appealing in our context. Summarizing, while we certainly do not rule out the possibility that other standard side-channel distinguishers provide further improvements to our results under certain heuristic assumptions, we believe SVM are natural candidates to investigate for enhanced IP detection.

The rest of the paper is structured as follows. Section 2 introduces the necessary background regarding the IP detection infrastructure, its underlying definitions, and the SVM classification tool. Section 3 instantiates our IP detection infrastructure. Experimental results in different contexts are presented and discussed in Section 4 and a conclusion is eventually given in Section 5.

2 Background

2.1 IP detection infrastructure

We take advantage of the IP detection infrastructure introduced in [15] and represented in Figure 1. It essentially makes use of soft hash functions [25]. By contrast to cryptographic hash functions, for which the output string is highly sensitive to small perturbations of the input, soft hash functions are such that similar objects should return highly correlated digests (i.e. be *perceptually robust*), while different objects should produce uncorrelated ones (i.e. be *content-sensitive*). The SPH used in this paper are a variation of soft hash function, where we additionally extract a physical feature from the objects to protect. The resulting IP detection infrastructure embeds the following elements:

- *Object to protect:* this can be any type of IP (source code, netlist, layout).
- *Physical feature vector evaluation:* this process outputs an intermediate response that is expected to represent the object to characterize in the most accurate manner. In other words, this intermediate string must be very content-sensitive. It can correspond to any physical emanation of the device running the IP (e.g. power consumption, electromagnetic radiation, ...).
- *Extraction:* this (optional) process essentially applies some signal processing and summarizes the feature vector into a (usually smaller) output hash value, that best trades content sensitivity for perceptual robustness (e.g. by selecting the "points of interest" in a side-channel measurement).
- *Detection:* this can be any statistical tool that allows determining the level of similarity between two hash values. Most side-channel distinguishers can be used for this purpose (e.g. Pearson's correlation coefficient in [15, 24]).

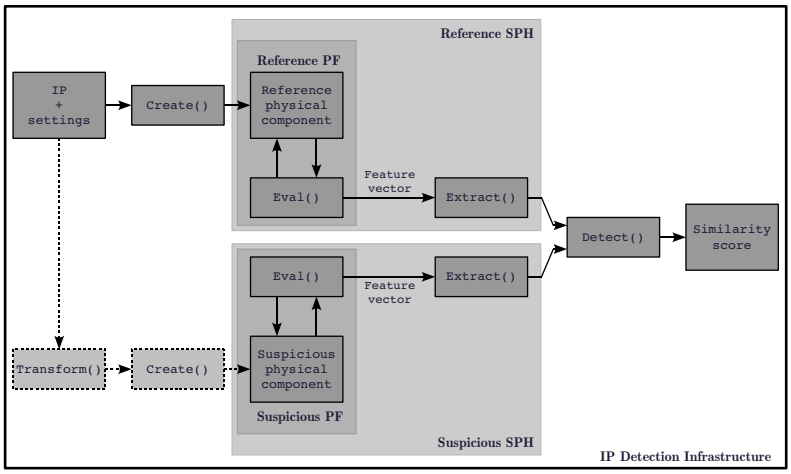

Fig. 1. Generic framework for IP detection.

As the detection of counterfeited IP essentially works by comparing hash values, it is assumed that the IP owner has characterized the SPH function of his design. Note that this characterization process does not have to be done during development time, but may also be done after the product has been released. As indicated by the dotted part on Figure 1, the suspicious IP may directly correspond to counterfeited designs, or slightly transformed ones. In the following, we will consider the re-synthesis of a design under a different set of constraints and the addition of a parasitic IP running in parallel as IP-preserving transformations. By contrast, a change of block cipher is naturally considered as non IP-preserving. In this context, the detection performances are measured with the content sensitivity and perceptual robustness properties defined as:

- *Perceptual robustness*: probability that two implementations that only differ by IP-preserving transformations lead to a similarity score higher than τ_r.
- *Content sensitivity*: probability that two implementations that differ by non-IP-preserving transformations lead to a similarity score lower than τ_s.

For the detection process to be successful, the condition $\tau_r \geq \tau_s$ must always be satisfied. Otherwise, a legitimate IP could be mistaken for a fraudulent one (or conversely). Eventually, such a generic framework can be run in different scenarios that make the detection more or less easy to perform. For example, the inputs/outputs of the design to protect and its source code can be known or unknown, and the framework can be applied to identical or different technologies, using identical or different measurement setups. These scenarios affect the physical features that are measured during the evaluation process.

2.2 Support Vector Machines

SVM are a class of statistical models used for data classification, popular in machine learning and artificial intelligence [9, 41]. Their goal is to learn target properties from a finite set of data points, possibly living in a high-dimensional space, in order to classify unseen samples. For this purpose, they essentially estimate a classification function, taking a vector of attributes as argument and returning a discrete decision. In our context, the data samples are the hashed vectors of the power consumption (next denoted as $\mathbf{x}$ and living in $\mathbb{R}^n$), and the estimated function is the belonging of those vectors to a class of IP to protect.

Binary SVM. The standard SVM model is binary and targets the estimation of Boolean functions. It requires a supervised (i.e. profiled) learning with labels $\{-1, +1\}$, annotating each sample of the training dataset. Its goal it to output a correct label for the samples of an independent test set. Let us assume that the training set is composed of m samples $\mathbf{x_i} \in \mathbb{R}^n$, with $i = 1, .., m$. For each sample $\mathbf{x_i}$, let $y_i \in \{-1, +1\}$ be the associated label. The binary SVM will estimate a decision function g whose sign defines the binary outcome of the classifier. This function g is based on the construction of an hyperplane, separating the two classes with the largest possible margin, in the geometrical space of the vectors. The margin represents the distance between the hyperplane and the closest member(s) of both distributions, identical for both sides. Maximizing this quantity leads to a better discrimination ability on unseen samples. These concepts are illustrated in Figure 2, handling a simple 2D classification task (diamonds vs. circles) where dashed lines are margin borders around the hyperplane.

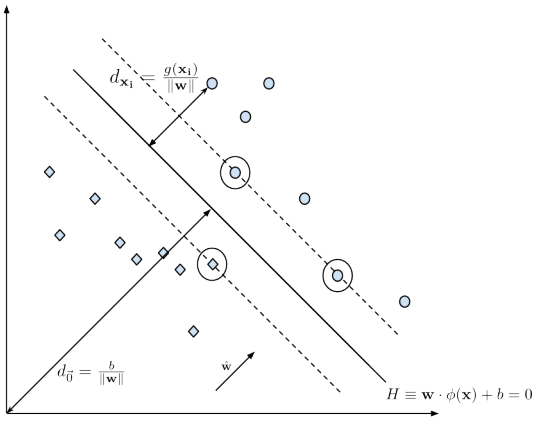

Fig. 2. Binary SVM with a 2D classification problem (the analytical distances from the hyperplane H to the origin and a datapoint $\mathbf{x_i}$ are given by $d_{\mathbf{0}}$ and $d_{\mathbf{x_i}}$, respectively).

Let $b \in \mathbb{R}$ and $\mathbf{w} \in \mathbb{R}^n$ be the parameters of the hyperplane, and $\mathbf{x}$ a data vector such that we have the following decision function:

$$g(\mathbf{x}) = \mathbf{w}^t \phi(\mathbf{x}) + b = \mathbf{w} \cdot \phi(\mathbf{x}) + b. \tag{1}$$

Here ϕ denotes a projection function that maps the data vectors into a higher (sometimes infinite) dimensional space. If ϕ is not defined as the trivial identity function (as in the figure), the hyperplane will be built in the projected space, usually called feature space (as opposed to the original data space). This allows to find non-linear boundaries in the data space, fitting to more complex observations. Equation (1) returns a quantity proportional with the analytical distance of a sample from the frontier ($d_{\mathbf{x_i}}$ in Figure 2): the further the point, the more robust the classification. When further referring to the distance from the hyperplane, we will always mean an evaluation of the g function. Parameters b and $\mathbf{w}$ are obtained by solving the following constrained quadratic program:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^t\mathbf{w} \\ \text{subject to} \quad & y_i * g(\mathbf{x_i}) \geq 1 \quad \forall i = 1, ..., m. \end{aligned} \tag{2}$$

When the problem is feasible with respect to the constraints, the data is said to be linearly separable in the feature space. As the problem is convex, there is a guarantee to find a unique global minimum. Usually, solvers compute the solution of the associated dual problem, with dual variables α_i, $i = 1, .., m$. This leads to an alternative formulation of the decision function g, whose sign provides the outcome of the classification of a new data vector $\mathbf{x}$:

$$f_{classify}(x) = sign\left(\sum_{\mathbf{x_i} \in SV} \alpha_i y_i \ \boldsymbol{K}(\mathbf{x_i}, \mathbf{x}) + b\right). \tag{3}$$

Here, SV denotes a subset of the original training set called the support vectors. They are solely needed to define the hyperplane associated with non-zero dual variables ($\alpha_i \neq 0$), and represent the closest points to the hyperplane living on the margin (circled in Figure 2). The symmetric kernel function $\boldsymbol{K}$ implicitely takes into account the projection ϕ in the dual problem: $\boldsymbol{K}(\mathbf{x_i}, \mathbf{x_j}) \equiv \phi(\mathbf{x_i})^T \phi(\mathbf{x_j})$. It substitutes the dot product appearing in Equation (1). This *kernel trick* allows the usage of complex projections without explicitly computing ϕ, which can be computational intensive (or even impossible) in high dimensions (e.g. using a Gaussian Kernel as we did in our experiments - see next - is equivalent to using an infinite-dimensional feature space).

The single class extension. Binary SVM require both positive and negative samples in the training set. In the context of IP detection, this is not directly applicable since the negative distribution is unknown (i.e. we cannot expect to obtain samples for all the IP that have been developed by third parties). For this reason, we will use a slightly more complex version of SVM called "unsupervised

One-class SVM" (OSVM). It allows getting rid of any hypothesis made about the distribution of the negative class, by working with unlabeled data. This extension was derived from the binary case thanks to the work of B. Schölkopf et al. in the early 2000's [36, 37]. The main underlying idea is to estimate the geometrical region concentrating most points of the distribution, by building a separating hyperplane having a maximum margin from the origin. For this purpose, OSVM rely on the assumption that the dataset contains a small fraction of outliers that will be considered as rejected samples. They associate these rejected samples with a penality cost, that depends on their distance from the hyperplane (cfr. Figure 3), and is added to the original SVM objective function (adjusted with a new parameter $\nu \in]0, 1]$). Let $\rho \in \mathbb{R}$ and $\mathbf{w} \in \mathbb{R}^n$ be the parameters of the hyperplane and $\xi_i \in \mathbb{R}^+$ a penality cost associated with $\mathbf{x_i}$ $(i = 1, .., m)$, it leads to the following quadratic problem:

$$\begin{aligned} \min_{\mathbf{w},\rho,\xi_i} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{\nu m}\sum_{i=1}^{m}\xi_i - \rho \\ \text{subject to} \quad & \mathbf{w}\cdot\phi(\mathbf{x_i}) \geq \rho - \xi_i \qquad \xi_i \geq 0, \forall i = 1, ..., m, \end{aligned} \tag{4}$$

with the classification function according to dual variables α_i given by:

$$f_{classify}(\mathbf{x}) = sign\left(\sum_{\mathbf{x_i}\in SV} \alpha_i \; K(\mathbf{x_i}, \mathbf{x}) - \rho\right). \tag{5}$$

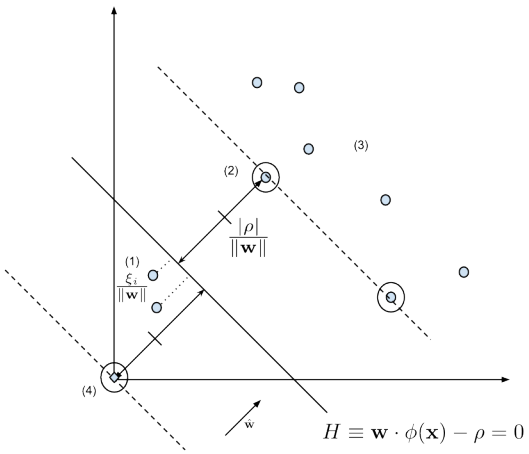

Fig. 3. `OSVM` with a 2-dimensional classification problem in which we distinguish four classes of points: (1) outliers with their penalty cost ξ_i, (2) support vectors, (3) other points, (4) the origin (the analytical half margin size is given by $\frac{|\rho|}{\|\mathbf{w}\|}$).

Instantiation. In this work, we use a Gaussian Radial Basis Function (`RBF`) kernel within the `OSVM` model:

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(\frac{-||\mathbf{x_i} - \mathbf{x_j}||}{2\sigma^2}\right) = \exp(-\gamma||\mathbf{x_i} - \mathbf{x_j}||), \quad \gamma, \sigma > 0. \tag{6}$$

Such a kernel has been proven to work well for a wide range of application data. Moreover, it can be shown that this `RBF-OSVM` model guarantees to find a solution (i.e. any dataset $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_m}$ is separable under a Gaussian kernel [36]). We also observed that the computational overhead of using that kernel instead of the linear one was negligible in our case. Eventually, building a model requires the setting of the parameters γ and ν. We selected them thanks to a grid-search, by optimizing the true positive rate TPR, defined as:

$$TPR = \frac{\#TP}{\#TP + \#FN}, \tag{7}$$

where $\#TP$ represents the frequency of vectors from the validation set correctly detected, while $\#FN$ is the frequency of vectors wrongly rejected by the model. This function has been evaluated on a dataset independent of the training set (the validation set) to avoid overfiting the parameters with the data used to train the model itself. Our choice of objective function is admittedly heuristic. As will be clear next, it was sufficient to obtain good experimental results (and in particular, we verified that it improved over randomly selected parameters).

3 Specification of the IP detection infrastructure

Given the tools presented in Section 2, we now need to incorporate SVM in the generic IP detection infrastructure of Figure 1. In this work we use the same datasets as in the earlier studies exposed in [24], leading to an identical evaluation phase. This section will briefly recall our measurement setup, valid for both reference and suspicious traces, then describes what the construction of the `OSVM` model implies in the extraction and detection phases. The `LibSVM` library suite was used to process the data, train and evaluate models, as it supports `OSVM` and many more features related to SVM classification tasks (see [11] for the details).

Object to protect. We investigated an FPGA case-study and took the netlists of five lightweight ciphers (`HIGHT` [19], `ICEBERG` [39], `KATAN` [10], `NOEKEON` [14] and `PRESENT` [8]), together with the one of the `AES` Rijndael, as objects to protect. These netlists were synthetized for a Xilinx Virtex-II Pro FPGA. We built only one reference model per protected IP, from its measurement obtained in a standalone context (i.e. with no parasitic IP in parallel), loaded and synthesized under standard options. Conversely, suspicious traces were measured in three different contexts corresponding to increasingly difficult detection challenges, namely identical standalone IP, re-synthetized (still standalone) designs, and re-synthetized designs with parasitic IP running in parallel.

Evaluation phase. We used the FPGA power consumption as physical feature vector. Measurement traces were obtained by measuring the voltage variations around a shunt resistor on the Sasebo-G board [1]. The device was running at 24 MHz, and the oscilloscope sampling frequency was set to 2,5 GHz.

Extraction phase. This step slightly differs from the one in previous papers, since we consider profiled side-channel distinguishers. Hence, while the extraction procedure applied to the reference IP (at the top of Figure 1) and the suspicious IP (at the bottom of the figure) was the same when using Pearson's correlation as detection tool, it has to differ in the case of SVM. Namely, the reference IP extraction outputs the parameters of the hyperplane that define the IP to protect, and the physical feature vectors of the suspicious IP will be compared to this model (rather than to other feature vectors). Besides and as usual, the extraction could include additional signal processing and selection of points-of-interest (e.g. some dimensionality reduction can be used to speed up computations) . In our experiments, these optional steps were usually ignored and we manipulated the full measurement traces directly. As mentioned in introduction, it is an interesting feature of SVM to allow dealing with such large dimensionalities efficiently. The only exception is our last case-study, where averaging was performed on the traces, to reduce the noise and improve detection capabilities.

As a technical remark, note that the `OSVM` require to work with vectors having identical dimension, both for the training and the evaluation of the model. Since different IP were processed, a common length had to be fixed. We choose to work with the shortest iteration length among our 6 IP (i.e. $n = 1251$ dimensions for the `AES`). This implicitly assumes that the traces can be synchronized, i.e. starting all their encryption cycle at t_0. This can be achieved by different means, e.g. computing the correlation over a sliding window. In our experiments, we observed that these cropped physical feature vectors were sufficiently specific to their generating IP for making effective detections. Both reference and suspicious extraction processes include this cropping operation as a preliminary step.

Detection phase. Each suspicious hash value is evaluated in the reference model. For this purpose, the `OSVM` simply outputs the a value that is proportional to the distance of this hash from the decision boundary (i.e. the SVM hyperplane), whose sign indicates the classification outcome. As previously mentioned, we will call it a "distance" for simplicity, and it can be interpreted as a similarity score lying on an open scale, whose expression is given by Equation (5).

4 Case studies

In this section, we analyze our `RBF-OSVM` model applied in four detection scenarios of increasing complexity. First we tackled the basic case of suspicious traces emitted by a standalone design for each of the 6 IPs. Secondly, we evaluated the case where each design was re-synthetized under a different set of constraints. We

then moved to a more challenging context, by including a parasitic IP running in parallel of the tested design (which is aimed to emulate a complex system). Eventually, we considered a combination of all these cases (which turned out to be more challenging, as we will explain). In practice, we made use of 2000 measurements per IP in each context. Two thirds were used for building the reference model (i.e. 1333 traces) and the remaining third was used as suspicious IP traces for validating the detection (i.e. 667 traces). We only present results for the case where `PRESENT` is the IP to protect since it was shown previously (and confirmed in our experiments) that it is the one leading to the most ambiguities (hence the most challenging to detect). Unless specified otherwise, the context in which the IP-detection is performed in the next subsections is the following: the inputs provided to the IP are unknown, we do not have access to the source code, and the same device and measurement setup was used for all the tests.

Before describing our experimental results in details, it is important to note a difference between (*i*) the classification outcome provided by the OSVM and (*ii*) the detection outcomes resulting from the use of the OSVM distance as a similarity score. In the first case, classification returns whether a sample has been properly labeled (belonging or not to the reference set) according to its relative position to the hyperplane. In the second case, detection is successful if the perceptual robustness threshold is higher than the content sensitivity one (we further call the difference the disambiguation gap). In theory (and, as it turns out, in practice too), it may of course happen that the classification fails while still giving rise to a sufficient disambiguation gap. This possibility essentially follows from the fact that the IP detection infrastructure can rely on carefully chosen thresholds for the content sensitivity and perceptual robustness.

4.1 Standalone FPGA designs

The results of the IP detection infrastructure applied to the standalone case are presented in Figure 4. Each column contains the similarity scores corresponding to one particular suspicious IP. The black dashed line corresponds to the decision threshold used for classification. The solid green line and red dashed line respectively correspond to the perceptual robustness threshold (i.e. the `PRESENT` trace having the lowest similarity score), and the content sensitivity threshold (i.e. the highest similarity score among all the non-`PRESENT` traces). Having positive similarity scores for the `PRESENT` IP implies that the classification outcome is correct. Having the perceptual robustness threshold higher than the content sensitivity one implies that the detection is successful. Note that the figure is in fact a zoom of the region of interest of Figure 8 available in appendix. In this zoomed version, we omitted the `AES` IP which is (as expected) quite different from other IP, and is therefore strongly rejected by the `OSVM` model. More precisely, we measured a voltage swing about 5 times larger in intensity, explaining this distance. This observation remains valid for the other detection scenarios and the `AES` was therefore left out of all the figures, for readability reasons. We observe from Figure 4 that all non-`PRESENT` IPs are correctly rejected by the `OSVM`

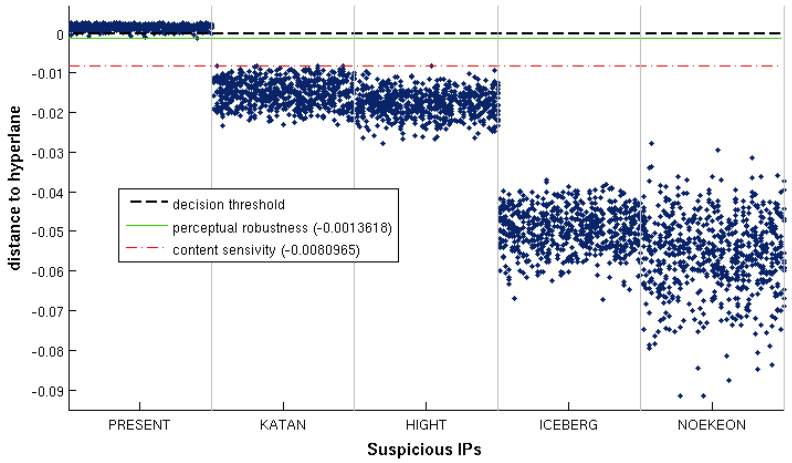

Fig. 4. Similarity scores for single suspicious standalone traces with unknown inputs.

classification since their scores lie below the decision threshold. `PRESENT` traces expose mostly positive scores, which was expected. A few outliers have however been rejected by the classification ($< 2.1\%$). This is mainly a consequence of the construction properties of the `OSVM` model. Still, the detection is succesfull in all our standalone experiments, as a disambiguation gap separates `PRESENT` traces from non-`PRESENT` ones. In [24], this case required to work with 10 times averaged traces (both for suspicious and reference traces) to get a similar result.

4.2 Re-synthetized FPGA designs

In this second (more challenging) scenario, we consider the application of a first IP-preserving transformation by a potential counterfeiter. Namely, we evaluate the impact of a placement and routing of our different block cipher implementations under a different set of constraints (i.e. with parameters to optimize the area of the layout instead of its timing). This reconfiguration does not modify the IP, which lies one abstraction layer above (source code or netlist).

Our experiments are summarized in Figure 5. We notice a slight increase of suspicious `PRESENT` wrongly rejected by the classification, as the model was originally trained to recognize traces corresponding to another set of synthesis parameters. Yet, the re-synthesis does not significantly affect the mean similarity scores of the other IP, which still guarantees a safe disambiguation gap. This result is interesting since in the previous work [24], such a detection was not possible in an unknown-plaintext scenario and the authors further had to average their traces to reach good IP detection probabilities. So it already suggests a useful improvement of our `OSVM`-based approach. Note that the set of synthesis options that we used in this section could equally stand as reference. We observed that the results are roughly identical independent of this a priori choice.

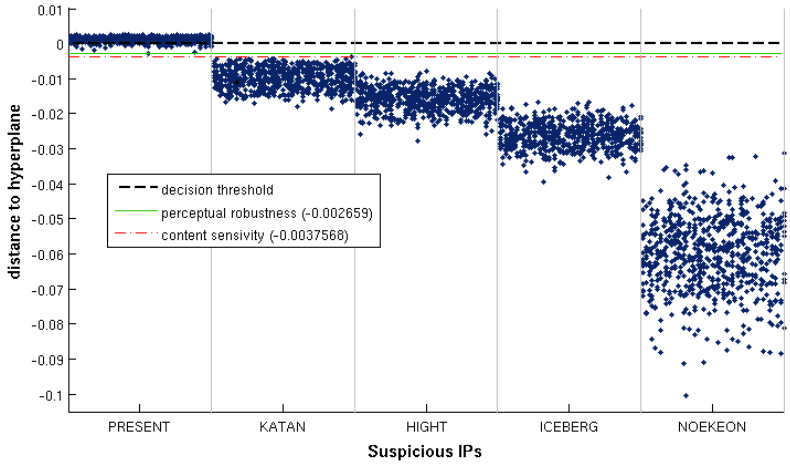

Fig. 5. Similarity scores for single suspicious re-synthesised traces, unknown inputs.

4.3 Parasitic IP running in parallel

We now study a practically-important case-study, where not only the suspicious IP would run on the target platform but also a parasitic one. As previously mentioned, the goal is to emulate a more realistic system where the IP is inserted in a neighborhood made of other running IP, hence altering the measured signal. As a first step in this direction, we investigated the case of a Linear Feedback Shift Register (`LFSR`) generating an "algorithmic noise" essentially proportional to its size (and studied sizes of up to 2018 bits). Figure 6 illustrates the results of our IP detection infrastructure in the most challenging (2048-bit) context. This time, we observe that the `OSVM` classification clearly fails at detecting the IP-preserving transformation applied on `PRESENT`, as their traces lie below the decision threshold. However, there still exists a disambiguation gap that ensures a perfect detection. So at this stage, the interest of the `OSVM`-based detection is clearly exhibited. Indeed, [24] provided an efficient detection until a 1024-bit `LFSR`, while our method allows us to detect IP with a 2048-bit `LFSR`. Moreover, this previous work had to harness data dependencies (i.e. known inputs) and averaging on selected points-of-interest for lowering the noise in the extraction phase, conversely to our work that considers unknown inputs and raw traces.

4.4 Advanced detection scenario

We finally investigated an advanced scenario where we combined the suspicious traces from all the previous contexts in a single experiment. This choice was mainly motivated by the observation of Figures 5 and 6, where we can see that the content sensitivity threshold in the parasitic IP scenario is higher than the perceptual robustness threshold in the re-synthesized one. It means that when

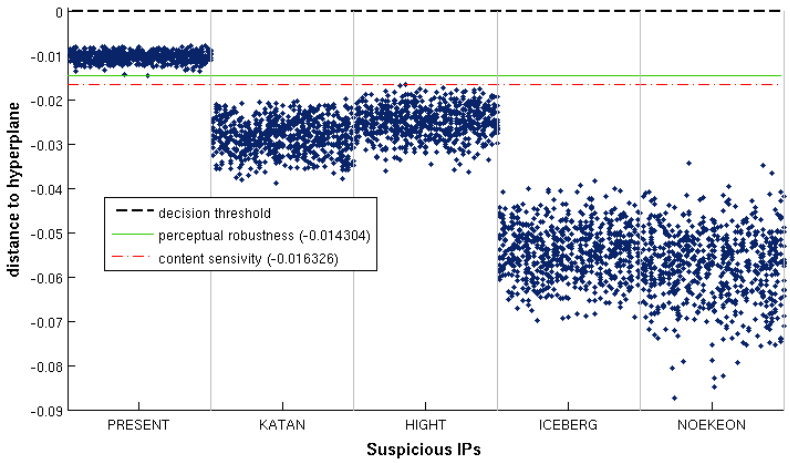

Fig. 6. Similarity scores for single suspicious traces with parasitic IP, unknown inputs.

these scenarios are mixed together, false detection or false non-detection may occur whatever detection threshold is chosen. Such pathological cases typically happen with the most challenging detection of `PRESENT` with parasitic IP from a re-synthesized `KATAN`, and required two modifications/improvements.

First, we had to move to a known input context (i.e. we kept the key and plaintext constant during the experiments), allowing us to take advantage of data dependencies in the traces. Intuitively, this is because the information exploited in the feature vectors captured in an unknown input scenario essentially corresponds to a correlation between the operations performed by the IP and its measured power consumption. The known input context adds a correlation between the data being manipulated and the measurements. Hence, a new (data-dependent) reference model was built for `PRESENT`. Secondly, and in order to get rid of a part of the algorithmic noise, we worked with averaged suspicious traces rather than single ones. This implies a slight change in the definition of the extraction phase, which now includes this 10 times averaging step.

The corresponding results are reported in Figure 7. This time, only `PRESENT` and `KATAN` are considered as suspicious IP, since they are the most challenging ones. Combining averaging with a characterization of the data dependencies naturally gave rise to more detailed profiles for the reference IP, as can be observed in this zoomed figure. First, we can now distinguish the different `PRESENT` IP, even after IP-preserving transformations. For example the similarity scores of standalone (1) and re-synthetized (2) designs are nicely separated (even though the latter ones have the right classification label, as expected). Next, we observe that the classification outcome for `PRESENT` with parasitic IP is even worse than before (moving from -0.01 to -0.04, roughly). However, this new model strongly rejects the different variants of `KATAN` (4,5,6). So the tweaked IP detection returns a positive disambiguation gap that makes these two implementa-

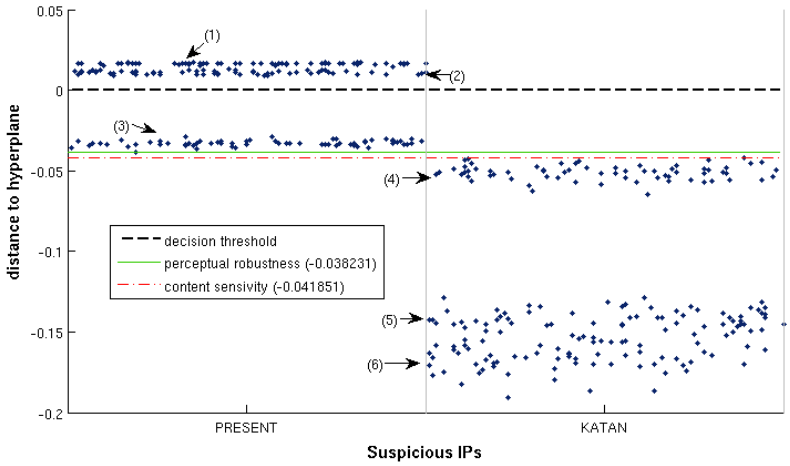

Fig. 7. Similarity scores for 10 times averaged suspicious traces in a combined setting with known inputs: (1) standalone `PRESENT`, (2) re-synthesized `PRESENT`, (3) `PRESENT` with paras. IP, (4) standalone `KATAN`, (5) re-synthesized `KATAN`, (6) `KATAN` with paras. IP.

tions distinguishable. It is interesting to note that in this last combined context, we required both an improvement of the signal (i.e. data dependencies) and a reduction of the noise (i.e. averaging) to obtain successful detections.

5 Conclusion

Our results further validate SPH functions as a useful ingredient in the detection of IP theft. They also suggest a context in which SVM (and their single-class extension) seem an appealing side-channel distinguisher. Of course, IP protection is an extremely challenging (and sometimes hard to define) problem. So the tools in this work should only be seen as one part of the solution. In particular, determined adversaries could envision more complex IP-preserving transformations than the ones we analyzed, and evaluating a change of technology between the reference and suspicious IP is an interesting scope for further research. Hopefully, there also remains tracks from improving the detection results, either by advanced statistical tools, or by improved (e.g. localised electromagnetic) measurements. In other words, there is a wide range of tradeoffs, between the complete reverse engineering of a chip and the characterization of its power consumption, that can be used by designers to protect their IP. As the cheapest and most flexible solution for this purpose, we believe SPH functions can at least be used as a first step in this direction, prior to more expensive approaches.

Acknowledgements. This work has been funded in parts by the Walloon region WIST program project MIPSs and by the European Commission through the ERC project 280141 (acronym CRASH). François-Xavier Standaert is an As-

sociate Researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). Stéphanie Kerckhof is a PhD student funded by a FRIA grant, Belgium.

References

1. Sasebo-G measurement board. `http://www.rcis.aist.go.jp/special/SASEBO/SASEBO-G-en.html`.
2. *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA*. IEEE Computer Society, 2011.
3. Amr T. Abdel-Hamid, Sofiène Tahar, and El Mostapha Aboulhamid. A survey on IP watermarking techniques. *Design Autom. for Emb. Sys.*, 9(3):211–227, 2004.
4. Frederik Armknecht, Roel Maes, Ahmad-Reza Sadeghi, François-Xavier Standaert, and Christian Wachsmann. A formalization of the security features of physical functions. In *IEEE Symposium on Security and Privacy* [2], pages 397–412.
5. Catalin Baetoniu. FPGA IFF copy protection using Dallas semiconductor/Maxim DS2432 secure EEPROMs. XAPP780, May 28, 2010.
6. Timo Bartkewitz and Kerstin Lemke-Rust. Efficient template attacks based on probabilistic multi-class support vector machines. In Mangard [29], pages 263–276.
7. Georg T. Becker, Markus Kasper, Amir Moradi, and Christof Paar. Side-channel based watermarks for integrated circuits. In *HOST*, pages 30–35, 2010.
8. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. Present: An ultra-lightweight block cipher. In Paillier and Verbauwhede [31], pages 450–466.
9. Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.
10. Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. Katan and ktantan - a family of small and efficient hardware-oriented block ciphers. In Clavier and Gaj [13], pages 272–288.
11. Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
12. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Jr. et al. [21], pages 13–28.
13. Christophe Clavier and Kris Gaj, editors. *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, volume 5747 of *Lecture Notes in Computer Science*. Springer, 2009.
14. Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. Nessie proposal: NOEKEON. Available from `http://gro.noekeon.org/`.
15. François Durvaux, Benoît Gérard, Stéphanie Kerckhof, François Koeune, and François-Xavier Standaert. Intellectual property protection for integrated systems using soft physical hash functions. In *WISA*, pages 208–225, 2012.
16. Louis Goubin and Mitsuru Matsui, editors. *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*, volume 4249 of *Lecture Notes in Computer Science*. Springer, 2006.

17. Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls. FPGA intrinsic PUFs and their use for IP protection. In *CHES*, pages 63–80, 2007.
18. Annelie Heuser and Michael Zohner. Intelligent machine homicide - breaking cryptographic devices using support vector machines. In Schindler and Huss [34], pages 249–264.
19. Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bonseok Koo, Changhoon Lee, Donghoon Chang, Jaesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. Hight: A new block cipher suitable for low-resource device. In Goubin and Matsui [16], pages 46–59.
20. Gabriel Hospodar, Benedikt Gierlichs, Elke De Mulder, Ingrid Verbauwhede, and Joos Vandewalle. Machine learning in side-channel analysis: a first study. *J. Cryptographic Engineering*, 1(4):293–302, 2011.
21. Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors. *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*. Springer, 2003.
22. Andrew B. Kahng, John Lach, William H. Mangione-Smith, Stefanus Mantik, Igor L. Markov, Miodrag Potkonjak, Paul Tucker, Huijuan Wang, and Gregory Wolfe. Watermarking techniques for intellectual property protection. In *DAC*, pages 776–781, 1998.
23. Andrew B. Kahng, Stefanus Mantik, Igor L. Markov, Miodrag Potkonjak, Paul Tucker, Huijuan Wang, and Gregory Wolfe. Robust IP watermarking methodologies for physical design. In *DAC*, pages 782–787, 1998.
24. Stéphanie Kerckhof, François Durvaux, François-Xavier Standaert, and Benoît Gérard. Intellectual property protection for fpga designs with soft physical hash functions: First experimental results. In *HOST*, pages 7–12, 2013.
25. Frédéric Lefèbvre, Jacek Czyz, and Benoit M. Macq. A robust soft hash algorithm for digital image signature. In *ICIP (2)*, pages 495–498, 2003.
26. Liran Lerman, Gianluca Bontempi, and Olivier Markowitch. Side channel attack: an approach based on machine learning. *Constructive Side-Channel Analysis and Secure Design, COSADE*, 2011.
27. Matthew Lewandowski, Richard Meana, Matthew Morrison, and Srinivas Katkoori. A novel method for watermarking sequential circuits. In *HOST*, pages 21–24, 2012.
28. Bernhard Linke. Xilinx FPGA IFF copy protection with 1-wire SHA-1 secure memories. XAPP3826, July 21, 2006.
29. Stefan Mangard, editor. *Smart Card Research and Advanced Applications - 11th International Conference, CARDIS 2012, Graz, Austria, November 28-30, 2012, Revised Selected Papers*, volume 7771 of *Lecture Notes in Computer Science*. Springer, 2013.
30. Stefan Mangard, Elisabeth Oswald, and François-Xavier Standaert. One for all - all for one: unifying standard differential power analysis attacks. *IET Information Security*, 5(2):100–110, 2011.
31. Pascal Paillier and Ingrid Verbauwhede, editors. *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*. Springer, 2007.
32. Josyula R. Rao and Berk Sunar, editors. *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*. Springer, 2005.

33. Bimal K. Roy and Willi Meier, editors. *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*, volume 3017 of *Lecture Notes in Computer Science*. Springer, 2004.
34. Werner Schindler and Sorin A. Huss, editors. *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings*, volume 7275 of *Lecture Notes in Computer Science*. Springer, 2012.
35. Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Rao and Sunar [32], pages 30–46.
36. Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001.
37. Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Comput.*, 12(5):1207–1245, May 2000.
38. Eric Simpson and Patrick Schaumont. Offline hardware/software authentication for reconfigurable platforms. In *CHES*, pages 311–323, 2006.
39. François-Xavier Standaert, Gilles Piret, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. Iceberg : An involutional cipher efficient for block encryption in reconfigurable hardware. In Roy and Meier [33], pages 279–299.
40. David Martinus Johannes Tax. *One-class Classification: Concept-learning in the Absence of Counter-examples*. PhD thesis, Delft University of Technology, 2001.
41. Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.
42. Daniel Ziener and Jürgen Teich. Power signature watermarking of IP cores for FPGAs. *Signal Processing Systems*, 51(1):123–136, 2008.

A Stand-alone FPGA Designs: Complete Results

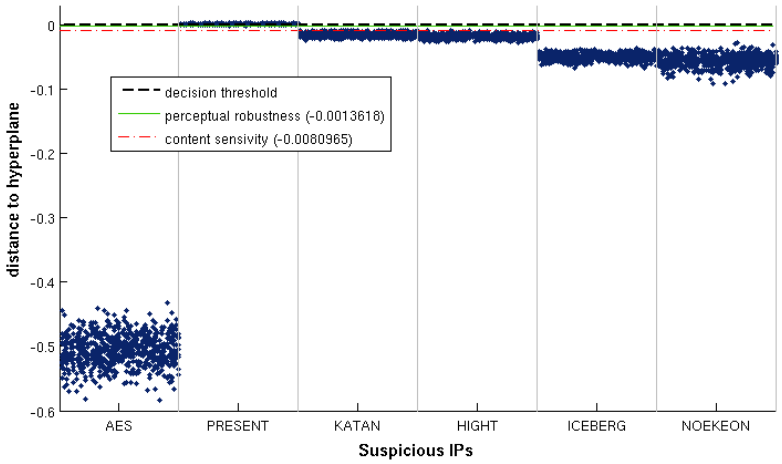

Fig. 8. Similarity scores for single suspicious standalone traces with unknown inputs.