Pushing the Limit of Non-Profiling DPA using Multivariate Leakage Model

Suvadeep Hajra
Dept. of Computer Science & Engg.
Indian Institute of Technology, Kharagpur, India
suvadeep.hajra@gmail.com

Debdeep Mukhopadhyay
Dept. of Computer Science & Engg.
Indian Institute of Technology, Kharagpur, India.
debdeep.mukhopadhyay@gmail.com

***Abstract*—Profiling power attacks like Template attack and Stochastic attack optimize their performance by jointly evaluating the leakages of multiple sample points. However, such multivariate approaches are rare among non-profiling Differential Power Analysis (DPA) attacks, since integration of the leakage of a higher SNR sample point with the leakage of lower SNR sample point might result in a decrease in the overall performance. One of the few successful multivariate approaches is the application of Principal Component Analysis (PCA) for non-profiling DPA. However, PCA also performs sub-optimally in the presence of high noise. In this paper, a multivariate model for an FPGA platform is introduced for improving the performances of non-profiling DPA attacks. The introduction of the proposed model greatly increases the success rate of DPA attacks in the presence of high noise. The experimental results on both simulated power traces and real power traces are also provided as an evidence.**

***Index Terms*—Differential Power Attack (DPA), Correlation Power Attack (CPA), leakage model, multivariate leakage model, non-profiling attack, multivariate distinguisher, multivariate DPA.**

I. Introduction

The success rate of the Differential Power Analysis (DPA) [12],[13] attacks is largely influenced by the Signal-to-Noise Ratio (SNR) [13] of the power traces. As a consequence, in many applications, Power Analysis attacks are either preceded by various pre-processing techniques like integration (Chapter 4.5.2 of [13]), PCA [4], filtering [15] for the reduction of noise in the power traces or followed by some post-processing techniques like averaging [7],[2],[4] for the reduction of the effect of noise on the outputs of the distinguisher. These techniques attempt to improve the performance of the DPA attacks directly or indirectly by extracting information from multiple sample points. However, those techniques are mainly based on some heuristic approaches and do not exhibit performance improvement in many scenarios.

Various profiling attacks like Template attack [6] and Stochastic attack [16] provide optimal performance by jointly evaluating the leakages at multiple sample points. However, they use a separate profiling step for approximating the multivariate leakage distribution [18] of the power traces. The profiling step requires a large number of power traces to estimate the multivariate leakage distribution with sufficient accuracy. Moreover, in most of the cases, it needs the knowledge of the secret key. Thus, optimising the performance of non-profiling DPA by considering the joint distribution of the leakages of multiple sample points is an open issue. This work attempts to do so using a model based approach.

In this work, our goal is to gain partial information of the multivariate leakage distribution of the power traces from the overall trace statistics like mean, variance etc. which can be easily computed without knowing the secret key. It should be noted that such attempt already exists in the form of using Principal Component Analysis (PCA) [1],[17],[4] in side-channel analysis. PCA projects the data-dependent variations from all the sample points of the power traces into the first principal component by analysing its covariance matrix. However, it performs sub-optimally on noisy power traces (see Sec. VI-B). In this paper, we extend the conventional leakage model for multiple sample points which, in turn, leads us to a multivariate leakage model. The proposed multivariate leakage model, once verified for a device, can be used to predict the (relative) SNR of each sample point of the power traces. Hence, it can strengthen the existing non-profiling DPA attacks by introducing new multivariate distinguishers which can combine the results from multiple sample points according to their relative SNR. Additionally, it can be applied to improve the sub-optimal behavior of PCA (described in [4]) for low SNR power traces. The model is experimentally verified for iterative hardware architectures on the Xilinx Virtex-5 FPGA embedded in a side-channel evaluation SASEBO-GII board (see Section III). A multivariate distinguisher based on the multivariate leakage model has been introduced. We also experimentally verified the effectiveness of the new distinguisher using both simulated traces with varying SNR and real traces. The results show a significant improvement in the performance of the new distinguisher for low SNR traces as compared to other existing distinguishers.

Rest of the paper is organized as follows. In Section II, preliminaries of Differential Power Analysis are described. Section III describes some profiling results on AES power traces. In Section IV, the multivariate model has been introduced. Section V provides a way to compute the relative SNR's of sample points using the multivariate leakage model. In Section VI, a new multivariate distinguisher has been introduced along with its application to principal component decomposition of the traces. Sections VII and VIII describe the attack results on simulated traces and real traces respectively. Finally conclusions have been drawn in Section IX.

II. Preliminaries

A. Notations

For the rest of the paper, we will use a calligraphic letter like $\mathcal{X}$ to denote a finite set and the corresponding capital letter X to denote a random variable over the set. Corresponding small letter x is used to denote a particular realisation of X. $P(.)$ is used to denote the probability of the event. $E[X]$, σ_X and $Var(X)$ are used to denote mean, standard deviation and variance of the random variable X respectively. We also denote by $Cov(X, Y)$ and $Corr(X, Y)$, the covariance and the Pearson's correlation coefficient between random variables X and Y respectively. We denote a vector $\{x_0, x_2, \cdots, x_k\}$ by $\{x_i\}_{0 \leq i \leq k}$. Gaussian distribution with mean m and standard deviation σ is represented by $\mathrm{N}(m, \sigma)$.

B. Differential Power Analysis

We will mainly follow the formalisation of Differential Power Analysis by Standaert et al. in [18]. It is briefly described below.

Let E be an iterative block cipher with block size b and number of rounds r. Let S be a key dependent intermediate variable of E. S is called *target* and satisfies $S = F_{k^*}(X)$, where X be a random variable representing a part of the known plaintext or ciphertext and $F_{k^*} : \mathcal{X} \to \mathcal{S}$ be a function determined by both the algorithm and the subkey $k^* \in \mathcal{K}$ (note that subkey is a small part of the secret key such that it is efficiently enumerable). We denote by L_t the random variable that represents the side channel leakage of an implementation of E at time instant t, $0 \leq t < rT$ where T is the number of samples collected per round.

In DPA, the attacker collects a set of traces $\mathrm{O} = \{o_0, \cdots, o_{q-1}\}$ resulted from the encryption (or decryption) of a sequence of q plaintexts (or ciphertexts) $\{p_0, \cdots, p_{q-1}\}$ (or $\{c_0, \cdots, c_{q-1}\}$) using the fixed but unknown key with subkey $k^* \in \mathcal{K}$ in a physical implementation of E. It should be noted that each o_i is a vector of size rT i.e. $o_i = \{o_{i,j}\}_{j=0}^{rT-1}$ where $o_{i,j}$ be the leakages of the j^{th} time instant during the i^{th} encryption (or decryption). Then, a distinguisher D is used which by taking the leakage vector $\{o_0, \cdots, o_{q-1}\}$ and the corresponding input vector $\{x_0, \cdots, x_{q-1}\}$ as inputs, outputs a distinguishing vector $\mathrm{D} = \{d_k\}_{k \in \mathcal{K}}$. For a successful attack, $k^* = argmax_{k \in \mathcal{K}}\ d_k$ holds with a non-negligible probability.

C. Leakage Model and Univariate Distinguisher

In DPA, it is assumed that the power consumption of a CMOS device at a time instant is dependent on the intermediate value manipulated at that point. Suppose the *target* S is manipulated at time instant t^* (call it *interesting* time instant). According to the conventional leakage model [5]:

$$L_{t^*} = \tilde{\Psi}(S) + N \quad (1)$$

$$= \tilde{\Psi}(F_{k^*}(X)) + N \quad (2)$$

where the function $\tilde{\Psi} : \mathcal{S} \to \mathbb{R}$ maps the target S to the deterministic part of the leakage and $N \sim \mathrm{N}(m, \sigma)$ accounts for the independent noise.

At the time of attack, the attacker chooses a suitable prediction model $\Psi : \mathcal{S} \to \mathbb{R}$ and computes the hypothetical leakage vector denoted by random variable $P_k = \Psi(S_k) = \Psi(F_k(X))$ for each key hypothesis $k \in \mathcal{K}$. In *univariate DPA*, the attacker is provided with the leakage of the interesting time instant t^*, $L_{t^*} = \tilde{\Psi}(F_{k^*}(X)) + N$. On receiving the leakage, she computes the distinguishing vector $\mathrm{D} = \{d_k\}_{k \in \mathcal{K}}$ such that $d_k = D(L_{t^*}, P_k) = D(\tilde{\Psi}(F_{k^*}(X)) + N, \Psi(F_k(X)))$ using a distinguisher D.

When the hardware leakage behavior follows a well known leakage model like Hamming weight model or Hamming distance model, some known prediction model Ψ closely approximates $\tilde{\Psi}$ i.e. $\tilde{\Psi}(s) \approx a \cdot \Psi(s)$ holds for some real constant a and for all $s \in \mathcal{S}$. Then, Eq. 1 can be approximated as

$$L_{t^*} = a \cdot \Psi(S) + N \quad (3)$$

Thus, the actual leakage vector L_{t^*} is linearly related to the hypothetical leakage vector for the correct key $P_{k^*} = \Psi(F_{k^*}(X))$. On the other hand, there is no such relation between L_{t^*} and the hypothetical leakage vector for a wrong key $P_k = \Psi(F_k(X))$ since $F_{k^*}(X)$ and $F_k(X)$ are almost independent for $k \neq k^*$. In Correlation Power Analysis (CPA) [5], Pearson's correlation is used to detect the linearity between L_{t^*} and P_k by computing $d_k = Corr(a \cdot \Psi(F_{k^*}(X)) + N, \Psi(F_k(X)))$ for all $k \in \mathcal{K}$. Since, Pearson's correlation detects the linear relation between two variables, it performs better than other attacks like Mutual Information Analysis (MIA) [8], Difference of Mean (DoM) [12]. When the hardware leakage model is not sufficiently known, 'generic' attacks like MIA perform better than CPA. In the rest of the paper, we will consider only the scenarios where the hardware follows a well known leakage behavior.

D. Multivariate DPA

In most of the practical scenarios, the point of interest t^* is not known before hand. Thus in practice, DPA attacks are multivariate in nature i.e. they take the leakages of multiple sample points as the inputs and generate the output. Most common form of multivariate DPA attacks applies a univariate distinguisher on each of the sample points independently and then, simply chooses the best result among those. However, in a different strategy, the attacker sometimes uses multivariate distinguishers which produce results based on the joint evaluation of the leakages at multiple sample points. Such multivariate distinguishers are common in profiling attacks like Template attack [6] and Stochastic attack [16].

Though multivariate distinguishers on unprotected implementations are rare in non-profiling context (example exists in [19]), there have been several attempts to improve the success rates of non-profiling DPA attacks by *integrating* the outputs of a univariate distinguisher at multiple sample points [2],[15],[4]. However, unlike profiling attacks where the multivariate leakage distribution of the power traces is approximated in an explicit profiling step, non-profiling attacks are vulnerable to decrease in success rate resulting from the integration of the output of a high SNR sample point to that of low SNR sample point. Thus, a successful integration of the leakages of multiple sample points requires the successful determination of the relative SNR of each sample point. We take a step in this direction in the next section.

III. Profiling the Power Traces of AES

In this section, we investigate the behaviour of the leakages of an AES implementation over a wide range of sample points due to the computation of an intermediate variable. We start with an unprotected AES which is implemented using parallel iterative hardware architecture on SASEBO-GII. SASEBO-GII [11] is a standard side-channel evaluation board which consists of FPGA device Virtex-5 xc5vlx50 for implementing cryptographic algorithms. The traces acquired using this setup are not vertically aligned. The vertical alignment of the traces are performed by subtracting the DC bias from each sample point of the trace. The DC bias of each trace is computed by averaging the leakages of a window taken from a region where no computation is going on.

We choose the target S to be the 128-bit input to the last round which is computed from the ciphertext using the secret key. Consequently, predicted leakage $P = P_{k^*}$ is calculated using Hamming distant model i.e. by computing the Hamming distance of the target S and the ciphertext. To examine how the dependency between the actual leakage L_t and the correct predicted leakage P varies over a range of sample points, we estimate the following metrics over 300 sample points around the register update of the last round of AES using $20,000$ power traces.

1) *Squared Pearson's Correlation between Data Dependent Leakage and Predicted Leakage (SCDP)*: It is defined as follows:

$$SCDP_t = Corr^2(E[L_t|P], P)$$

It reveals the linear dependency between the deterministic leakage $E[L_t|P]$ at sample point t and the predicted leakage P. It should be noted that if the leakage of a sample point t follows Eq. 3, then the empirical estimation of $SCDP_t$ using a finite number of traces will be close to one. On the other hand, if no such relation holds for a sample point t, $SCDP_t$ will be almost zero.

2) *Variation of Data Dependent Leakage (VDL)*:

$$VDL_t = Var(E[L_t|P])$$

It reveals the variations in leakage due to the predicted leakage P at sample point t. Sometimes, it is used to quantify the signal in the leakage. On the other hand, noise is quantified by $Var(L_t - E[L_t|P])$.

3) *Squared Mean Leakage (SML)*:

$$SML_t = E^2[L_t]$$

It has been included to study the behavior of the other metrics in relation with the mean leakage.

Fig. 1a shows that as the cycle begins, with the mean leakage (SML), SCDP also rises rapidly, remains almost constant for about 150 sample points and then it decreases slowly. The slight fluctuations in the curve are due to the presence of small amount of noise after averaging a limited number of power traces. This leads us to the following observation:

Observation 1: The deterministic part of the leakages at a *large* number of sample points show high linear dependencies with the correct predicted leakage P.

Various profiling attacks also take advantage of the data dependency of multiple leakage points. However, they are more generic since they can consider different prediction model for different sample points at the cost of expensive profiling step.

From figure 1b, we see that VDL almost superimposes on SML i.e. VDL is highly correlated to SML. This leads us to the following observation:

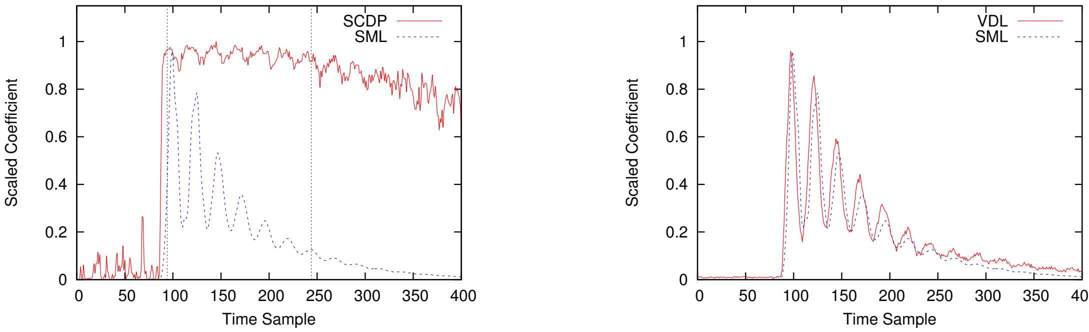
Observation 2: The variation in the deterministic part of the leakages is correlated to the square of the mean leakages. In other words, the second observation states that the magnitude of the variation at a sample point due to target S is proportional to the mean value (strength) of the leakage at that sample point. It should be noted that a similar kind of observation can be found in Chapter 4.3.2 of [13] for the leakages of a micro-controller. The authors have also suggested several trace compression techniques based on the observation and have shown their usefulness to attack software implementation of AES. However, to the best of the authors' knowledge, no attempt has been made to incorporate these observations into the conventional leakage model. In the next section, we extend the conventional leakage model by using these two observations.

IV. Introducing Multivariate Leakage Model

In [12], Kocher et al. mentioned the possibility of using the leakages of multiple sample points by the attacker in higher-order DPA. Later in [14], Messerges formalized the notion of *nth-order DPA* as an attack mechanism which exploits the leakages of n different sample points corresponding to n *different* intermediate values calculated during the execution of the algorithm. In this paper, we are interested in *n-variate* DPA which can exploit the leakages of n different sample points related to a *single* intermediate value calculated during the execution of the algorithm. Motivated by the observations of Sec. III, we define *n-variate* leakage model as follows.

Definition 1: In **n-variate leakage model**, leakages of n distinct sample points are assumed to be dependent upon a single intermediate value calculated during the execution of an algorithm.

Note that since $Corr(E[L_t|P], P) \approx 1$ for $t_0 \leq t < t_0 + \tau$, in a noise-free environment, all the leakage samples in the window contain almost same information about the target S (as far as the linear part of the leakage is considered). Thus, combining those would not provide any advantage. But, in practical scenarios i.e. in the presence of noise, combining the information from multiple leakage samples would actually help to reduce the noise.



(a) Plots of SCDP and SML

(b) Plots of VDL and SML

Fig. 1: Plots of the chosen metrics in the last round of unprotected implementation of AES

A. A Multivariate Leakage Model for Iterative Hardware Architecture on FPGA Platform

Observation 1 and 2 immediately extend the conventional leakage model given by Eq. (3), into the following multivariate leakage model:

$$\begin{aligned} L_t &= a_t \cdot \Psi(S) + N_t \\ &= a_t \cdot P + N_t, \quad t_0 \le t < t_0 + \tau \end{aligned} \tag{4}$$

where $a_t \in \mathbb{R}$ and the random vector $\{N_{t_0}, \cdots, N_{t_0+\tau-1}\}$ follows a multivariate Gaussian distribution with zero mean vector. It should be noted that the linear relation in Eq. (4) is a consequence of Observation 1 while Observation 2 enforces mean vector of the multivariate Gaussian distribution to be a zero vector. In a parallel iterative hardware architecture, a single round consists of several parallel S-boxes and the attacker targets only a part of it (usually a single S-box). Thus, in addition to the predicted leakage P due to the computation of the target $S = F_{k^*}(X)$, leakage due to the computation of the other parallel bits adds to it. This is known as algorithmic noise and we denote it by U. It should be noted that for a fully serialized architecture, U takes the value zero. Leakages due to the key bits and the control bits is denoted by c. Since key scheduling and the controlling operations are fixed for a fixed round in all the encryptions, c is constant for all the inputs.

Thus, we can adopt Eq. 4 to incorporate these new variables as follows:

$$L_t = a_t \cdot (P + U + c) + N_t, \quad t_0 \le t < t_0 + \tau \tag{5}$$

$$= a_t \cdot (I + c) + N_t \tag{6}$$

where $I = P + U$. We are interested in the leakages of the above window namely $\{t_0, t_0 + 1, \cdots, t_0 + \tau - 1\}$ that can be roughly determined by the clock cycle in which the target operation is being performed (see Section VI-C). We denote this time span by $\{0, 1, \cdots, \tau - 1\}$ and in the rest of the paper, power trace is referred by the sample points of this time span only.

Next section demonstrates how this model can be useful for predicting the relative SNR of each sample point of a power trace in low SNR scenarios.

V. Application of the Multivariate Leakage Model to Estimate the SNR of the Sample Points

Mangard et al. quantifies the information leakage for each sample point of a trace using signal-to-noise ratio (SNR) [13]. In our context, it can be defined as

$$SNR_t = \frac{Var(E[L_t|I])}{Var(L_t - E[L_t|I])} \tag{7}$$

Here, $Var(E[L_t|I])$ quantifies the signal part of the leakage and $Var(L_t - E[L_t|I])$ quantifies the electronic noise.

There are several existing techniques to compute the SNR of the sample points. They are mostly used to compress the traces in profiling attacks. But, most of them such as *sosd*, *sost* [9] assume the key to be known. Other techniques like PCA perform sub-optimally in the presence of high noise [4]. However, the multivariate leakage model provides a way to estimate the relative SNR (i.e. SNR of a sample point with respect to the SNR of the other sample points instead of the absolute value of the SNR) of each sample point without the knowledge of the secret key, hence, makes it applicable to non-profiling setup also. Let $\alpha(t)$, $\mu_L^2(t)$ and $\sigma_L^2(t)$ be the functions over time such that $\alpha(t) = SNR_t$, $\mu_L^2(t) = SML_t = E^2[L_t]$ and $\sigma_L^2(t) = Var(L_t)$. Then, the multivariate leakage model given in Eq. (6) leads us to Proposition 1.

Proposition 1: Suppose that the power traces are following the *multivariate leakage model* described in Eq. (6). If the variance of the electronic noise at each sample point is significantly higher than the signal variance i.e. $Var(E[L_t|I]) \ll Var(L_t - E[L_t|I])$ for $0 \le t < \tau$, then the SNR of a sample point t, $\alpha(t)$ is proportional to Squared Mean to Variance Ratio (SMVR) $\frac{\mu_L^2(t)}{\sigma_L^2(t)}$.

Proof: By taking the expectation of both sides of Eq. (6), we get

$$\begin{aligned} E[L_t] &= a_t \cdot (E[I] + c) \\ \text{or,} \quad a_t &= \frac{E[L_t]}{E[I] + c} \end{aligned} \tag{8}$$

From the definition of SNR in Eq. (7), we get

$$\begin{aligned}
\alpha(t) &= \frac{Var(E[L_t|I])}{Var(L_t - E[L_t|I])}, \\
&= \frac{Var(a_t \cdot (I + E[U] + c))}{Var(L_t) - Var(E[L_t|I])}, && \text{from Eq. (5) and independent noise assumption} \\
&\approx \frac{Var(a_t I)}{Var(L_t)}, && \text{since } Var(E[L_t|I]) \ll Var(L_t - E[L_t|I]) < Var(L_t) \\
&= \frac{a_t^2 Var(I)}{Var(L_t)}, \\
&= \frac{E^2[L_t] Var(I)}{(E[I]+c)^2 Var(L_t)}, && \text{from Eq. (8)} \\
&= \frac{\mu_L^2(t)}{\sigma_L^2(t)} \times \frac{Var(I)}{(E[I]+c)^2} && \text{from the definition of } \mu_L^2(t) \text{ and } \sigma_L^2(t)
\end{aligned}$$

■

It should be noted that both $\mu_L(t)$ and $\sigma_L(t)$ can be computed without knowing the correct key. Thus, Proposition 1 can be used to determine the relative SNR of a sample point in the presence of high noise. Next, we will see how it can be useful for designing multivariate distinguishers in non-profiling DPA attacks.

VI. Designing new Multivariate Distinguishers

The performances of many univariate distinguishers including CPA and classical DPA are susceptible to the level of SNR. Their performances get better at a sample point with higher SNR and become worse at a sample point with lower SNR [13]. We can adopt a univariate distinguisher for multivariate DPA by applying the univariate distinguisher on each sample point of the power traces separately and combining the result of each sample point using a second level distinguisher according to their relative SNR.

To elaborate the above approach, let us consider D to be a univariate distinguisher and we apply it to each sample point t, $0 \le t < \tau$, of the power traces independently. At the end, D outputs τ distinguishing vectors $\{\mathrm{D}(t)\}_{t=0}^{\tau-1}$ where each $\mathrm{D}(t)$ is a vector of $|\mathcal{K}|$ elements i.e. $\mathrm{D(t)} = \{\mathrm{d_k(t)}\}_{\mathrm{k} \in \mathcal{K}}$. Thus, the vector $\{d_k(0), \cdots, d_k(\tau-1)\}$ represents the distinguishing values for the key hypothesis k at all the τ sample points. Since the correct key hypothesis k^* can compute the *target* S correctly, the distinguishing values for the correct key at time t, $d_{k^*}(t)$ depends on the SNR at t, and thus on SMVR $\mu_L^2(t)/\sigma_L^2(t)$ (thanks to Proposition 1). In other words, the vector $\{d_{k^*}(0), \cdots, d_{k^*}(\tau-1)\}$ will be strongly 'correlated' to the SMVR vector $\{\frac{\mu_L^2(0)}{\sigma_L^2(0)}, \cdots, \frac{\mu_L^2(\tau-1)}{\sigma_L^2(\tau-1)}\}$. On the other hand, since a wrong key hypothesis $k \neq k^*$ wrongly guesses the value of S i.e. $S \neq F_k(X)$, there is almost no correlation between $\{d_k(0), \cdots, d_k(\tau-1)\}$ and the SMVR vector. Thus, we can deploy a second level distinguisher $\tilde{D}$ to detect the correlation between the vectors $\{\frac{\mu_L^2(t)}{\sigma_L^2(t)}\}_{t=0}^{\tau-1}$ and $\{d_k(t)\}_{t=0}^{\tau-1}$ for all key hypothesis $k \in \mathcal{K}$ and return k as the correct key for which the correlation is maximum.

To summarise, a univariate distinguisher D can be extended for multivariate DPA as follows:

1) Apply the distinguisher D for each sample point t, $0 \le t < \tau$, of the power traces independently. At the end, D outputs τ distinguishing vectors $\{\mathrm{D}(t)\}_{t=0}^{\tau-1}$ where each $\mathrm{D}(t)$ is a vector of $|\mathcal{K}|$ elements i.e. $\mathrm{D}(t) = \{d_k(t)\}_{k \in \mathcal{K}}$.
2) Construct $|\mathcal{K}|$ vectors $\{d_k(t)\}_{t=0}^{\tau-1}$ for each key hypothesis $k \in \mathcal{K}$. And also construct the SMVR vector $\{\frac{\mu_L^2(t)}{\sigma_L^2(t)}\}_{t=0}^{\tau-1}$.
3) Employ a second univariate distinguisher $\tilde{D}$ which outputs a distinguishing vector $\tilde{\mathrm{D}} = \{\tilde{d}_k\}_{k \in \mathcal{K}}$ where $\tilde{d}_k = \tilde{D}(\{d_k(t)\}_{t=0}^{\tau-1}, \{\frac{\mu_L^2(t)}{\sigma_L^2(t)}\}_{t=0}^{\tau-1})$.
4) Return k as the correct key for which $\tilde{d}_k$ is maximum.

We will now explore this approach in several contexts in the following sections.

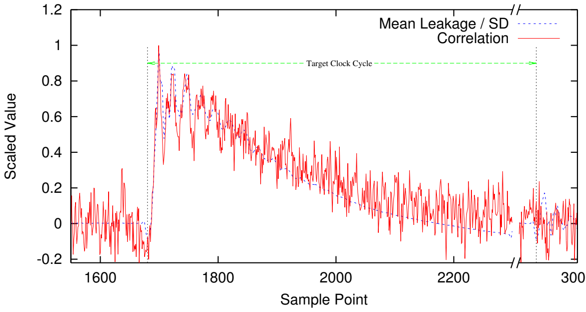


Fig. 2: Plots of the mean leakage normalised by the standard deviation and correlation of the correct key during the last round register update during AES encryption.

A. Extending CPA for Multivariate Leakage Model

In order to construct an effective multivariate distinguisher, we choose CPA as the first level univariate distinguisher since it is well accepted as one of the best performer when the hardware leakage follows a standard leakage model [3],[20]. To choose a proper second level distinguisher, we compute the Pearson correlation $\rho_{k^*}(t)$ between the leakage at sample point t and the predicted leakage for the correct key hypothesis $P = \Psi(S) = \Psi(F_{k^*}(X))$ using Eq. (5).

$$\begin{aligned}
\rho_{k^*}(t) &= \frac{Cov(L_t, P)}{\sqrt{Var(L_t)Var(P)}} \\
&= \frac{Cov(a_t(P + U + c) + N_t, P)}{\sqrt{Var(L_t)Var(P)}} \\
&= \frac{a_t Cov(P, P)}{\sqrt{Var(L_t)Var(P)}} \\
&= \frac{a_t Var(P)}{\sqrt{Var(L_t)Var(P)}} \\
&= \frac{\mu_L(t)}{\sigma_L(t)} \times \frac{\sigma_P}{E[I] + c}, \quad \text{from Eq. 8} \qquad (9)
\end{aligned}$$

According to Eq. (9), not only the magnitude of $\rho_{k^*}(t)$ is proportional to $\frac{\mu_L(t)}{\sigma_L(t)}$ but the sign of $\rho_{k^*}(t)$ is also determined by the sign of $\mu_L(t)$. Moreover, the relation no more depends on the high noise condition as in Proposition 1, thus, is applicable to power traces with all SNR levels.

Fig 2 plots the mean leakage $\frac{\mu_L(t)}{\sigma_L(t)}$ and the correlation $\rho_{k^*}(t)$ between leakage L_t and the correct key guess for the first S-box at 200 sample points during the last round of the encryptions. To generate it, we have used $32,000$ traces collected from parallel iterative implementation of AES on SASEBO-GII (please refer to Section III. The figure clearly indicates that the correlation curve has high positive correlation with the mean leakage curve.

To exploit the above knowledge of the relation between $\rho_{k^*}(t)$ and $\frac{\mu_L(t)}{\sigma_L(t)}$, we propose the following distinguisher:

Scalar Product It takes the scalar product of the vectors $\{\rho_k(t)\}_{t=0}^{\tau-1}$ and $\{m(t)\}_{t=0}^{\tau-1}$ i.e. $\tilde{d}_k = \sum_{t=0}^{\tau-1}\rho_k(t)m(t)$ where $m(t) = sgn(\mu_L(t))\mu_L^2(t)/\sigma_L^2(t)$. Here function $sgn(\mu_L(t))$ takes the value 1 for $\mu_L(t) \geq 0$ and -1 otherwise.

In other words, the distinguisher takes the sum of the outputs of CPA at all the sample points weighted by the 'signed' SMVR of each sample point.

B. *Improving the Performance of PCA for low SNR Traces*

PCA is a well known statistical technique for dimensionality reduction based on variations of data. It converts a set of interrelated observations (variables) into a set of new variables called principle components (PCs) such that the PCs are uncorrelated to each other and they are ordered decreasingly by their variance. Thus, first few PCs contain most of the variations in data while the later components capture a small amount of variations which are assumed to be caused by noise. Thus, the removal of the later components (which have lower variance) while preserving the first few components is a common noise reduction technique.

PCA was first introduced in the context of SCA by Archambeau et al. [1] where they used it to reduce the dimensions of the traces for Template attack. Later, in [17], Sylvain et al. introduced it as a non-profiling distinguisher and in [4], Batina et al. introduced it as a pre-processing technique. For low noise traces, the PCA on the power traces (represented as matrix with rows containing different traces and columns containing different sample points) projects the variations caused by the target S into the first PC (since it is the largest component). Thus, univariate DPA on the first PC yields better result.

However in [4], Batina et al. also mentioned the limitation of PCA in high noise scenarios. Since, in high noise scenarios, the larger part of the variations is caused by the noise rather than the signal, the SNR's of the first few PCs are in fact quite low. Thus, univariate distinguishers on the first PC perform badly. Moreover, it is difficult to identify the sample points with higher SNR. However, based on some empirical observations, [4] has suggested a new distinguisher, namely *CPA Abs-Avg* distinguisher, which takes the average of the absolute value of the correlations of each sample points to compute the final output.

We suggest to use the multivariate model to find the principal components (PCs) having more information. Since PCA is a linear transformation, the principal component decomposition of the power trace matrix $\mathrm{O} = \{o_{i,j}\}_{(i,j)=(0,0)}^{(q-1,\tau-1)}$ (recall that, $o_{i,j}$ stands for the leakage of j^{th} sample point of the i^{th} trace) is given by the $q \times \tau$ matrix $\tilde{\mathrm{O}} = \mathrm{OW}$ where W is a $\tau \times \tau$ matrix. The j^{th} column of W represents the eigenvector corresponding to the j^{th} largest eigenvalue of the covariance matrix of O. Each column of $\tilde{\mathrm{O}}$ represents a single PC and each row represents an observation or a trace. Due to the linearity of the transformation, the principal component decomposition traces $\tilde{\mathrm{O}}$ also follows the multivariate model given by Eq. 5 and 6 . Thus, we can apply Proposition 1 on $\tilde{\mathrm{O}}$. Fig 3 validates Eq. 9 (a consequence of the multivariate leakage model) by plotting the correlation of the correct key and the mean leakage normalised by the standard deviation at each sample points of the principal component decomposition of the set T_{sim}^8. A consequence of this observation is that *Scalar Product* can be directly applied to the principal component decomposition of the power traces.

It should be noted that most of the tools like MATLAB® removes the mean of each sample point of the original traces as the first step of the transformation. Thus, we computed the mean vector $\mu_L = \{\mu_L(t)\}_{t=0}^{\tau-1}$ of the observation matrix O before applying the transformation. And after the transformation, we multiplied μ_L by the eigenvector matrix W obtained from the MATLAB® function 'princomp' to get $\mu_{\tilde{L}} = \mu_L\mathrm{W}$, the mean vector of the principal component decomposition traces.

C. *Determination of Window*

For an iterative hardware architecture, the window can be set to the whole period of the clock cycle in which the target operation is being performed. However, to reduce the computational complexity resulting from performing computations on all points in the clock period, other measures can be taken based on SMVR. For our experiments, we have roughly chosen the window from the beginning of the target clock cycle up to a sample point for which the SMVR is slightly greater than zero.

VII. Attacks on Simulated Traces with different SNR levels

To test the effectiveness of the new approaches, we collected a set of $20,000$ power traces: T_{org} of the encryptions of AES implemented on the setup described in Section III using parallel iterative hardware architecture. We then removed the noises of all the traces (using the correct key) and created a set of noise-less traces: T_{nl}. Next, we created 4 sets of simulated traces each having $20,000$ traces: T_{sim}^1, T_{sim}^2, T_{sim}^4 and T_{sim}^8 by adding a Gaussian noise to each sample point of T_{nl} having standard deviation 1, 2, 4 and 8 times the standard deviation of the noise at the same sample point of T_{org} respectively. It should be noted that the average noise variance of T_{sim}^2, T_{sim}^4 and T_{sim}^8 are respectively 2^2, 4^2 and 8^2 times the average noise variance of T_{sim}^1 while all the four sets are having same

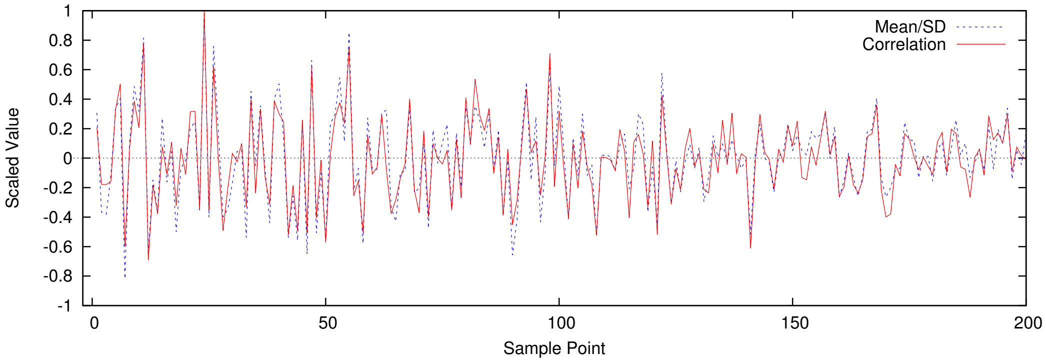


Fig. 3: Plots of the mean leakage normalised by the standard deviation and the correlation of the correct key at the first 200 PCs of the principal component decomposition of the set T^8_{sim}.

signal variances. Thus, average SNR of T^2_{sim}, T^4_{sim} and T^8_{sim} are $1/2^2$, $1/4^2$ and $1/8^2$ times the SNR of T^1_{sim} respectively.

We applied *Scalar Product*, classical CPA [5] and *CPA Abs-Avg* [4] to attack the above 4 sets of simulated traces. We also applied the above three distinguishers on the principal component decomposition of the four sets by transforming them using MATLAB® function 'princomp' (refer to Sec. VI-B). For CPA on PCs, we tested both *CPA on first PC* and standard multivariate CPA on all the PCs. However, *CPA on first PC* yields better results. Profiling phase of Stochastic attack also determines the correct key as a byproduct of estimating the deterministic leakages. We also implemented that as a distinguisher. In the rest of the paper, we refer to this distinguisher as *Stochastic* distinguisher.

To compare the performances of the distinguishers, we have used *average guessing entropy* as a metric. The guessing entropy [18] of a distinguisher is given by the average rank of the correct key. Thus, it decreases as the attack becomes better and reaches one if it can find the correct key in all the trials. Average guessing entropy is computed by taking the average of the guessing entropy's of all the 16 S-boxes. To compute the guessing entropy of the above distinguishers, we divided each set of $20,000$ simulated traces among four groups of $5,000$ traces and applied the distinguishers on each group separately and took the average of their results.

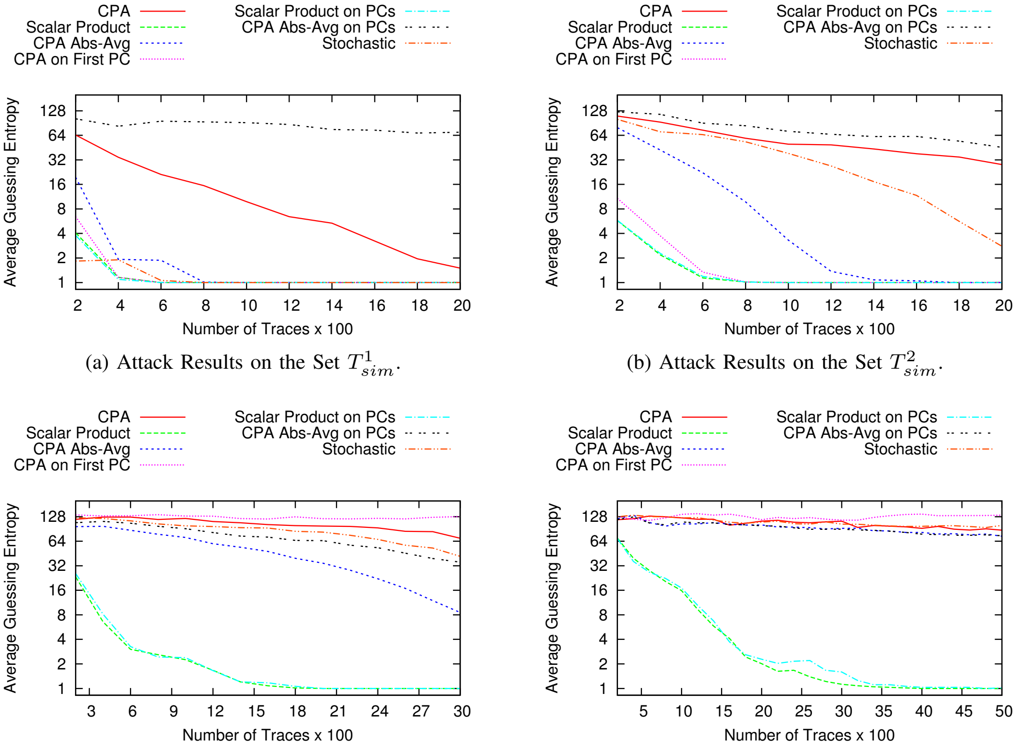
Average guessing entropy of the attacks on the four sets of simulated traces are shown in Fig. 4. From this figure, we can summarise the following observations:

1) *Scalar Product* performs far better than the other distinguishers on both the original traces and the principal component decomposition of the traces. Moreover, the differences of the performances are more if the average noise level of the trace-set is more.
2) When the average noise level is comparatively low i.e. for the trace-sets T^1_{sim} and T^2_{sim}, *CPA on first PC* performs almost equally well to *Scalar Product*. This is due to the fact that most of the data dependent variations (signal part of the leakage) have been projected to the first PC by PCA. Thus *Scalar Product* does not get any extra advantage over *CPA on first PC* by extracting information from multiple sample points.
3) The average noise levels of the trace-sets T^4_{sim} and T^8_{sim} are high enough to make PCA unable to project all the data dependent variations into the first PC. Rather, in Fig. 3, we can see that data gets correlated to multiple sample points of the principal component decomposition traces of T^8_{sim}. As a result, *Scalar Product on PCs* performs far better than *CPA on first PC*.
4) *Scalar Product* on the original traces and *Scalar Product on PCs* perform similarly though the later requires PCA as a pre-processing step which is computationally intensive.
5) The performance of *CPA Abs-Avg* on the principal component decomposition degrades for high SNR traces also. This is due to the fact that for high SNR traces most of the data variations are captured by the first few PCs only. Thus, *CPA Abs-Avg* reduces the effective SNR of the first few PCs by averaging them with rest of the low SNR sample points.
6) Though the non-profiling *Stochastic* attack performs quite well for T^1_{sim}, it performs badly for other sets of traces.

VIII. Attacks on Real Traces

To verify the effectiveness of the proposed distinguisher on real traces, we collected 20 sets of $2,000$ traces of an AES implementation on SASEBO-GII (please refer to Section III). The implementation is based on parallel iterative architecture. The S-boxes are implemented using Xilinx device primitive: distributed ROM. Using our setup, the maximum SNR of the obtained power traces is close to 0.42 which is quite high.

Average Guessing entropy's of *Scalar Product* along with classical CPA, *CPA Abs-Avg* and non-profiling *Stochastic* attacks are shown in Fig. 5. It should be noted that the obtained power traces contain some correlated noise (noises in multiple sample points are correlated among themselves). As a result, the third PC instead of the first PC shows the maximum SNR in the principal component decomposition of the traces. Thus, *CPA on PCs* performs better than *CPA on first PC* and is included in the figure. Due to the computational limitation,



(a) Attack Results on the Set T^1_{sim}.

(b) Attack Results on the Set T^2_{sim}.

(c) Attack Results on the Set T^4_{sim}.

(d) Attack Results on the Set T^8_{sim}.

Fig. 4: Plots of the average guessing entropy of various distinguishers with the increase in the number of power traces on the four trace-sets having different average SNR.

Stochastic attack is performed on 160 sample points while other attacks are performed on 300 sample points.

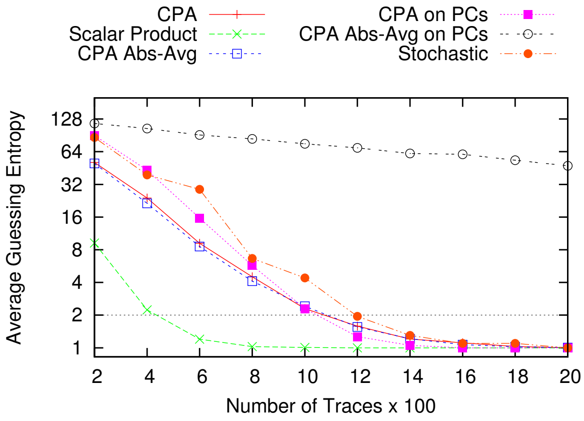


Fig. 5: Average Guessing Entropy of various attacks on the real traces of a parallel iterative implementation of AES on Xilinx FPGA device Virtex-5.

It is clear from Fig. 5 that *Scalar Product* is performing better than all the other attacks. It takes about 400 traces to bring down the average guessing entropy below two, while all other attacks take more than 1,000 traces for the same.

IX. Conclusion

In this paper, we have introduced a *multivariate leakage model* for iterative hardware architecture on FPGA device Virtex-5. The introduced model allows an attacker to predict the relative SNR of each sample point of the power traces without even knowing the correct key. We have further discussed how existing univariate distinguishers can be strengthened by extending it to *multivariate* distinguishers with the help of the relative SNR of the sample points. We have also introduced and empirically verified one multivariate distinguisher namely *Scalar Product* using both simulated power traces and real power traces. The results show that *Scalar Product* performs far better than the classical CPA as well as the recently introduced *CPA Abs-Avg Distinguisher* on low SNR scenarios which are more likely in future devices.

Several advanced DSP techniques like Wavelet transforms have been recently introduced in side-channel literature. However, optimal application of such techniques either requires the knowledge of the correct key or depends on some heuristically chosen parameters such as ‘scale level’. It can be an interesting study to see the applicability of the proposed multivariate leakage model in those situations.

The multivariate leakage model is validated on FPGA device Virtex-V. However, similar kinds of observations have been noticed in the literature on other platforms like micro-controllers. Hence, in future, exploring approaches based on multivariate leakage model on such other platforms could be worthy.

References

[1] C. Archambeau, E. Peeters, F.-X. Standaert, and J.-J. Quisquater. Template Attacks in Principal Subspaces. In Goubin and Matsui [10], pages 1–14.

[2] L. Batina, B. Gierlichs, and K. Lemke-Rust. Differential Cluster Analysis. In C. Clavier and K. Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 112–127. Springer, 2009.

[3] L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F.-X. Standaert, and N. Veyrat-Charvillon. Mutual Information Analysis: a Comprehensive Study. *J. Cryptology*, 24(2):269–291, 2011.

[4] L. Batina, J. Hogenboom, and J. G. J. van Woudenberg. Getting More from PCA: First Results of Using Principal Component Analysis for Extensive Power Analysis. In O. Dunkelman, editor, *CT-RSA*, volume 7178 of *Lecture Notes in Computer Science*, pages 383–397. Springer, 2012.

[5] E. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In M. Joye and J.-J. Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.

[6] S. Chari, J. R. Rao, and P. Rohatgi. Template Attacks. In B. S. K. Jr., Çetin Kaya Koç, and C. Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.

[7] C. Clavier, J.-S. Coron, and N. Dabbous. Differential Power Analysis in the Presence of Hardware Countermeasures. In Çetin Kaya Koç and C. Paar, editors, *CHES*, volume 1965 of *Lecture Notes in Computer Science*, pages 252–263. Springer, 2000.

[8] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel. Mutual Information Analysis. In E. Oswald and P. Rohatgi, editors, *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2008.

[9] B. Gierlichs, K. Lemke-Rust, and C. Paar. Templates vs. Stochastic Methods. In Goubin and Matsui [10], pages 15–29.

[10] L. Goubin and M. 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.

[11] T. Katashita, A. Satoh, T. Sugawara, N. Homma, and T. Aoki. Development of side-channel attack standard evaluation environment. In *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*, pages 403–408, 2009.

[12] P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[13] S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.

[14] T. S. Messerges. Using Second-Order Power Analysis to Attack DPA Resistant Software. In . K. Ko and C. Paar, editors, *CHES*, volume 1965 of *Lecture Notes in Computer Science*, pages 238–251. Springer, 2000.

[15] T. S. Messerges, E. A. Dabbish, and R. H. Sloan. Investigations of power analysis attacks on smartcards. In *Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology*, WOST'99, pages 17–17, Berkeley, CA, USA, 1999. USENIX Association.

[16] W. Schindler, K. Lemke, and C. Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In J. R. Rao and B. Sunar, editors, *CHES*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.

[17] Y. Souissi, M. Nassar, S. Guilley, J.-L. Danger, and F. Flament. First Principal Components Analysis: A New Side Channel Distinguisher. In K. H. Rhee and D. Nyang, editors, *ICISC*, volume 6829 of *Lecture Notes in Computer Science*, pages 407–419. Springer, 2010.

[18] F.-X. Standaert, T. Malkin, and M. Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.

[19] C. Whitnall and E. Oswald. A Comprehensive Evaluation of Mutual Information Analysis Using a Fair Evaluation Framework. In P. Rogaway, editor, *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2011.

[20] C. Whitnall, E. Oswald, and F.-X. Standaert. The myth of generic DPA...and the magic of learning. *IACR Cryptology ePrint Archive*, 2012:256, 2012.