# Side-Channel Leakage and Trace Compression using Normalized Inter-Class Variance

Shivam Bhasin [1], Jean-Luc Danger [1,2], Sylvain Guilley [1,2] and Zakaria Najm [1]

[1] Institut MINES-TELECOM, TELECOM ParisTech,
CNRS LTCI (UMR 5141),
Department COMELEC, 46 rue Barrault,
75 634 PARIS Cedex 13, FRANCE.
[2] Secure-IC S.A.S., 80 avenue des Buttes de Coësmes,
35 700 Rennes, FRANCE.

**Abstract**

Security and safety critical devices must undergo penetration testing including Side-Channel Attacks (SCA) before certification. SCA are powerful and easy to mount but often need huge computation power, especially in the presence of countermeasures. Few efforts have been done to reduce the computation complexity of SCA by selecting a small subset of points where leakage prevails. In this paper, we propose a method to detect relevant leakage points in side-channel traces. The method is based on Normalized Inter-Class Variance (NICV). A key advantage of NICV over state-of-the-art is that NICV does neither need a clone device nor the knowledge of secret parameters of the crypto-system. NICV has a low computation requirement and it detects leakage using public information like input plaintexts or output ciphertexts only. It is shown that NICV can be related to Pearson correlation and signal to noise ratio (SNR) which are standard metrics. NICV can be used to theoretically compute the minimum number of traces required to attack an implementation. A theoretical rationale of NICV with some practical application on real crypto-systems are provided to support our claims.

**Keywords**: Cryptography, side-channel analysis, leakage detection, correlation power analysis (CPA), ANOVA, NICV, TVLA, SNR, AES, linear regression analysis (LRA).

# 1 Introduction

Security-critical devices must undergo a certification process before being launched into the public market. One of the many security vulnerabilities tested in the certification process is Side-Channel Attacks (SCA [4, 5]). SCA pose a serious practical threat to physical implementation of secure devices by exploiting unintentional leakage from a device like the power consumption, electromagnetic emanation or timing. Several certification/evaluation labs are running SCA daily on devices under test to verify their robustness.

The certification process is expensive and time-consuming which also increases the overall time-to-market for the device under test. It worsens when the desired security level increases. For instance, it is usually considered that a Common Criteria (CC [7]) evaluation at highest assurance level for penetration attacks (AVA.VLAN.5) requires the device to resist attacks with 1 million traces. Similarly, the draft ISO standard 17,825 [12] (application note for international standard ISO/IEC 19 790, sibling to NIST / FIPS 140-2) demands resistance against side-channel analysis with 10,000 traces (level 3) and with 100,000 traces (level 4). The traces can have millions of points and thus running SCA on these traces can be really time consuming. Also several attacks must be tested on the same set of traces before certifying a device. To accelerate the evaluation process, a methodology should be deployed which compresses the enormous traces to a small set of relevant points.

## 1.1 Related Works

The compression of SCA traces which results in reduced time complexity of the attacks, can be achieved by selecting a small subset of points where leakage prevails. This issue of selecting relevant time samples have been dealt previously by some researchers. Chari et al. [5] use templates to spot interesting time samples. The method involves building templates $T$ on $n$ different values of the subkey. Interesting time samples can then be found as points which maximizes $\sum_{i,j=1}^{n}(T_i - T_j)$. In this equation, $T_i$ is the average of the traces when the sensitive variable belongs to the class $i$. Two further improvements were then proposed by Gierlichs et al. [14]. The first improvement, also called as Sum Of Squared pairwise Differences (SOSD), simply computes $\sum_{i,j=1}^{n}(T_i - T_j)^2$. SOSD avoids cancellation of positive and negative differences. SOSD can be further improved by normalizing it by some variance. This normalized SOSD is called SOST (Sum Of Squared pairwise T-differences [14], where a *T-difference* means a *Student T-test*) and computed as

$$\sum_{i,j=1}^{n}\left(\frac{T_i - T_j}{\sqrt{\frac{{\sigma_i}^2}{m_i} + \frac{{\sigma_j}^2}{m_j}}}\right)^2, \tag{1}$$

where $\sigma_i$ is the variance of $T$ in class $i$, and $m_i$ is the number of samples in class $i$. If $m$ is the total number of traces, we have $\sum_{i=1}^{n} m_i = m$, and $m_i$ is also $m$

times the estimated probability for the traces to belong to class $i$. When the classes are equally populated (i.e., $\forall i,\ m_i = m/n$), the SOST rewrites as:

$$\frac{m}{n} \sum_{i,j=1}^{n} \frac{(T_i - T_j)^2}{{\sigma_i}^2 + {\sigma_j}^2} \ ,$$

Similar ideas were also sketched earlier in [9].

A practical problem with template-based detection techniques comes from the computation of templates. First of all, templates require an access to a clone device. Secondly, templates need two sets of traces: one for profiling with random keys and another for attacking with an unknown but fixed key. An alternative to the latter limitation is model-based templates which can exploit the same set of traces as proposed in [1]. Although model-based templates can be really efficient, they are relevant for the chosen power model only.

Another method proposed in this context is the Principal Components Analysis (PCA [20]). PCA is used for dimensionality reduction. It yields a new basis of the time samples in which the inter-class variance is greater. This basis takes into account the covariance of the samples. In side-channel analysis, the goal of PCA is to gather all the information in a single (or few) component(s) [2]. Eventually, other empirical methods use chosen plaintext attacks, such as the differences between plaintexts `0x00...0000` and `0x00...00ff`. This technique requires many requests to check for all the bytes, not only the least significant byte. Furthermore, it is not always possible to choose the plaintext messages (e.g., when modes of operations with initial vectors are used).

In this paper, we propose a new method for leakage detection relying on a metric called "Normalized Inter-Class Variance" (NICV). This NICV method allows to detect interesting time samples, without the need of a profiling stage on a clone device. Hence the SCA traces can be compressed and the analysis could be greatly accelerated. The main characteristics of the proposed method are:

- NICV operates without the need of a clone device, i.e., it requires no profiling stage and use the same set of traces which are to be analyzed,
- it uses only public information like plaintext or ciphertext,
- the method is leakage model agnostic, it is not an analysis tool but a helper to speed up the analysis, but
- it can serve to evaluate the accuracy of various leakage models and choose which is the best applicable.

Compared to PCA, the purpose of NICV is to return the total variation of the traces at each time sample, so as to test which leakage model causes inter-class variation.

**Our Contributions:** Our contributions in this paper is three-fold. Firstly, we develop the theoretical background of NICV for leakage-detection. We relate

NICV with existing side-channel metrics like **Pearson Correlation Coefficient** and **Signal to Noise Ratio (SNR)**. NICV is also compared against other leakage detection techniques like SOST and SOSD. Secondly, we provide the theoretical lower bound of number of traces to recover the key. Finally, we apply NICV to real implementations on FPGA and smartcards to show its ability of leakage detection and other related applications.

The rest of the paper is organized as follows. General background to SCA is recalled in Sec. 2. The rationale of NICV to select SCA relevant time samples is detailed in Sec. 3. We also derive in this section a lower bound on the number of traces to recover the key. This is followed by some practical use cases applied on real devices like FPGA and smartcards in Sec. 4. Finally, Sec. 5 draws general conclusions. In appendix A, we provide with a comparison between NICV and TVLA (Test Vector Leakage Analysis [15]), two leakage detection metrics.

## 2 General Background

Side-channel analysis consists in exploiting dependencies between the manipulated data and the analog quantities (power consumption, electromagnetic radiation, ...) leaked from a CMOS circuit. Suppose that several power consumption traces, denoted $Y$, are recorded while a cryptographic device is performing an encryption or decryption operation. An attacker predicts the intermediate leakage $L(X)$, for a known part of the ciphertext (or plaintext) $X$ and key hypothesis $K$. Next, the attacker uses a distinguisher like Correlation Power Analysis (CPA [4]), to distinguish the correct key $k^\star$ from other false key hypotheses. CPA is a computation of the *Pearson Correlation Coefficient* $\rho$ between the predicted leakage $L(X)$ and the measured leakage $Y$, which is defined as:

$$\rho\left[L(X);Y\right] = \frac{\mathbb{E}\left[\left(L(X) - \mathbb{E}\left[L(X)\right]\right) \cdot \left(Y - \mathbb{E}\left[Y\right]\right)\right]}{\sqrt{\mathsf{Var}\left[L(X)\right] \cdot \mathsf{Var}\left[Y\right]}} \ ,$$

where $\mathbb{E}$ and $\mathsf{Var}$ denote the mean and the variance respectively. We notice that $\rho\left[L(X);Y\right] \in [-1;+1]$.

Various distinguishers have been proposed in literature. It is shown in [24] that customary statistical distinguishers eventually turn out to be equivalent when the signal-to-noise ratio gets high. The differences observed by an attacker are due to statistical artifact which arises from imprecise estimations due to limited numbers of observations. In the rest of the paper without loss of generality, we use CPA as a distinguisher.

Authors of [24] also show that a proper estimation of leakage model $L(X)$ can define the efficiency of the attack. Therefore a detection technique is needed which can pinpoint the relevant leakage points and the most efficient leakage model. In the following, we introduce NICV as a leakage detection technique and its power to evaluate estimated leakage models. As shown later, NICV is not a SCA channel distinguisher itself. NICV works in co-ordination with

any SCA distinguishers like CPA to enhance their performance. Even variance-based distinguishers as introduced in [35, 3, 27] can be made more efficient using NICV. As a preprocessing step, NICV can leverage metrics and distinguishers based on *Linear Discriminant Analysis* (LDA) [21] or on *Principal Component Analysis* (PCA) [16, 21, 34] are also using some kind of $L^2$ distance between classes (as does the NICV, see next section).

# 3 Leakage Detection using NICV

In this section, we first describe our *normalized inter-class variance* (NICV) detection technique. We provide the mathematical background of NICV and then discuss its behavior in a side-channel context. The practical application of NICV is covered in the next section.

## 3.1 Rationale of the NICV Detection Technique

Let us call $X$ one byte of the plaintext or of the ciphertext (that is, the domain of $X$ is $\mathcal{X} = \mathbb{F}_2^8$), and $Y \in \mathbb{R}$ the leakage measured by the attacker[1]. Both random variables are public knowledge. Then, for all leakage prediction function $L$ of the leakage knowing the value of $x$ taken by $X$ (as per Proposition 5 in [28]), we have:

$$\rho^2\left[L(X);Y\right] = \underbrace{\rho^2\left[L(X);\mathbb{E}\left[Y|X\right]\right]}_{0\leq\ \cdot\ \leq 1} \times \rho^2\left[\mathbb{E}\left[Y|X\right];Y\right] \quad . \tag{2}$$

Again in Corollary 8 of [28], the authors derive:

$$\rho^2\left[\mathbb{E}\left[Y|X\right];Y\right] = \frac{\mathsf{Var}\left[\mathbb{E}\left[Y|X\right]\right]}{\mathsf{Var}\left[Y\right]} \quad , \tag{3}$$

which we refer to as the *normalized inter-class variance* (NICV). It is an ANOVA (ANalysis Of VAriance) *F-test*, as a ratio between the explained variance and the total variance (see also the recent articles [6, 10]). The NICV is also used for linear regression analysis, where it is called the *coefficient of determination* and usually denoted by the symbol "$R^2$". Eventually, NICV has also the denomination *correlation ratio* [31]; for instance, it is employed in the context of side-channel analysis in [33], where it is used as a distinguisher and as a linearity metric.

Once combined, equations (2) and (3) yield that for all prediction function $L : \mathbb{F}_2^8 \to \mathbb{R}$ (realistic or not), we have:

$$\boxed{0 \leq \rho^2\left[L(X);Y\right] \leq \frac{\mathsf{Var}\left[\mathbb{E}\left[Y|X\right]\right]}{\mathsf{Var}\left[Y\right]} = \text{NICV} \leq 1} \quad . \tag{4}$$

[1] In general, $Y$ can be continuous, but $X$ must be discrete (i.e., $\mathcal{X}$ is of finite cardinality).

Therefore, the NICV is the *envelop* or maximum of all possible correlations computable from $X$ with $Y$. As a consequence of the Cauchy-Schwarz theorem, there is an equality in (4) if and only if $L(x) = \mathbb{E}[Y|X = x]$, which is the optimal prediction function[2].

In practice, the (square) CPA value does not attain the NICV value, owing to noise and other imperfections. This is illustrated in Fig. 1. The difference can come from various reasons like:

- The attacker knows the exact prediction function, but as usual not the actual key. For instance, let us assume the traces can be written as $Y = w_H(S(X \oplus k^\star)) + N$, where $k^\star \in \mathbb{F}_2^8$ is the correct key, $S : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ is a substitution box, $w_H$ is the Hamming weight function, and $N$ is some measurement noise, that typically follows a centered normal distribution $N \sim \mathcal{N}(0, \sigma^2)$. In this case, the optimal prediction function $L(x) = \mathbb{E}[Y|X = x]$ is equal to: $L(x) = w_H(S(X \oplus k^\star))$ (the only hypothesis on the noise is that it is *centered* and *mixed additively* with the sensitive variable). This argument is at the base of the soundness of CPA: $\forall k \neq k^\star, \rho[w_H(S(X \oplus k)); Y] \leq \rho[w_H(S(X \oplus k^\star)); Y]$ $\leq \sqrt{\mathsf{Var}[\mathbb{E}[Y|X]] / \mathsf{Var}[Y]}$.
- The CPA is smaller than NICV when the attacker assumes a wrong model, for instance $L(x) = w_H(x \oplus k^\star)$, when $Y = w_H(S(x \oplus k^\star)) + N$.
- Eventually, the attacker can have an approximation of the leakage model, for instance $L(x) = w_H(S(x \oplus k^\star))$, whereas actually $Y = \sum_{i=1}^{8} \beta_i \cdot S_i(x \oplus k^\star) + N$, where $\beta_i \approx 1$, but slightly deviate from one (a.k.a. stochastic model [32]).

The distance between CPA and NICV is, in *non*-information theoretic attacks (i.e., attacks in the proportional / ordinal scale, as opposed to the nominal scale [37]) similar to the distance between perceived information (PI) and mutual information (MI) [29]. Like CPA, NICV also achieves an asymptotically constant value, once the measurement set has reached a representative sample size.

It can also be noted that NICV is considered as the relevant distinguisher for the Linear Regression Analysis (LRA) [19, 22], which acts as the CPA but with a parametric leakage model.

Now, if $X$ is uniformly distributed, the NICV in itself *is not* a distinguisher.

[2] Rigorously: if and only if $L(x)$ is an affine function of $\mathbb{E}[Y|X = x]$.

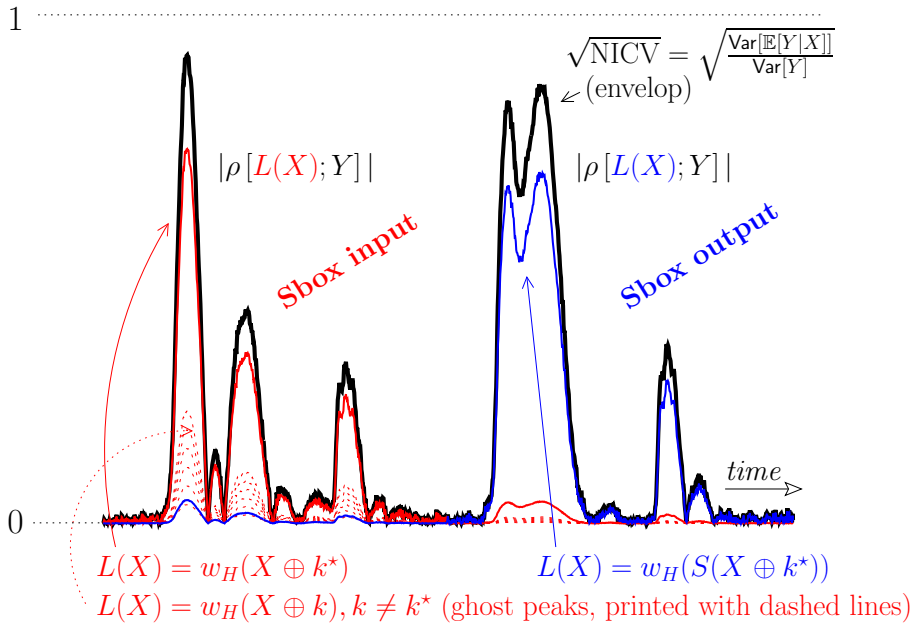



Figure 1: Illustration on NICV metric when $Y = \sum_{i=1}^{8} \beta_i \cdot S_i(x \oplus k^\star) + N$, and correlation coefficient for some prediction functions plotted against time

Indeed, if we assume that $Y = w_H(S(X \oplus k^\star)) + N$, then:

$$\begin{aligned}
\mathsf{Var}\left[\mathbb{E}\left[Y|X\right]\right] &= \textstyle\sum_{x\in\mathcal{X}} \mathsf{P}[X=x]\mathbb{E}\left[Y|X=x\right]^2 - \mathbb{E}\left[Y\right]^2 \\
&= \tfrac{1}{2^8}\textstyle\sum_{x\in\mathcal{X}} \mathbb{E}\left[w_H(S(x\oplus k^\star)) + N\right]^2 \\
&\quad - \left(\textstyle\sum_{x\in\mathcal{X}} \mathbb{E}\left[w_H(S(x\oplus k^\star)) + N\right]\right)^2 \\
&= \tfrac{1}{2^8}\textstyle\sum_{x'=x\oplus k^\star\in\mathcal{X}} \mathbb{E}\left[w_H(S(x'))\right]^2 \\
&\quad - \left(\textstyle\sum_{x'=x\oplus k^\star\in\mathcal{X}} \mathbb{E}\left[w_H(S(x'))\right]\right)^2 \\
&= \mathsf{Var}\left[w_H(S(X))\right] \quad .
\end{aligned}$$

On the other hand, $\mathsf{Var}\left[Y\right] = \mathsf{Var}\left[w_H(S(X))\right] + \mathsf{Var}\left[N\right]$. All in one:

$$\boxed{\mathrm{NICV} = \frac{\mathsf{Var}\left[\mathbb{E}\left[Y|X\right]\right]}{\mathsf{Var}\left[Y\right]} = \frac{1}{\frac{1}{\mathrm{SNR}} + 1}} \quad , \tag{5}$$

where the signal-to-noise ratio SNR is the ratio between:

- the signal, i.e., the variance of the informative part, namely $\mathsf{Var}\left[w_H(S(X \oplus k^\star))\right]$, and
- the noise, considered as the variance $\mathsf{Var}\left[N\right]$.

Clearly, Eqn. (5) does not depend on the secret key $k^\star$ as both $Y$ and $X$ are public parameters known to the attacker. Moreover, this expression is free from the leakage model $L(X)$, which means that NICV does not depend on the implementation. Thus, NICV searches for all linear dependencies of public parameter $X$ with available leakage traces $Y$ independant of the implementation. In case of protected implementation, if the countermeasure removes all linear leakages related to $X$, then the NICV should also eventually suppress. On the other hand, NICV can detect accidental linear leakage from a protected implementation.

**Remark 1** *In the binary case ($n = 2$) when both classes are equally probable, the expression of NICV simplifies to:* $NICV = \frac{\mathsf{Var}[\mathbb{E}[Y|X]]}{\mathsf{Var}[Y]} = \frac{\left(\frac{\mathbb{E}[Y|X=0]-\mathbb{E}[Y|X=1]}{2}\right)^2}{\mathsf{Var}[Y]}$.
*It is similar to Cohen's d metric.*

Now comparing NICV with other leakage detection techniques like SOST and SOSD we can give the following remarks.

**Remark 2** *SOSD is actually proportional to the inter-class variance (this point* was not *made by the authors of [14]). Indeed, with our notations,* $T_i \doteq \mathbb{E}\left[Y|X=i\right]$.

*And thus:*

$$\sum_{i,j}(T_i - T_j)^2 = 2 \times 2^n \sum_i T_i^2 - 2\left(\sum_i T_i\right)^2 =$$
$$2 \times 2^{2n} \sum_x \mathbb{E}\left[Y^2 | X = x\right] \mathsf{P}[x] - 2\left(2^8 \mathbb{E}\left[Y\right]\right)^2$$
$$= 2^{2n+1} \mathsf{Var}\left[\mathbb{E}\left[Y|X\right]\right] \ .$$

*But this inter-class variance* is not *normalized. Therefore, the SOSD can be large at samples where* $\mathsf{Var}\left[N\right]$ *is large, although not containing (much) information.*

**Remark 3** *SOST which was proposed as an improvement over SOSD is normalized. Using our notations, SOST is equal to*

$$m \sum_{(x,x') \in \mathcal{X}^2} \frac{\left(\mathbb{E}\left[Y|X=x\right] - \mathbb{E}\left[Y|X=x'\right]\right)^2}{\sqrt{\frac{\mathsf{Var}[\mathbb{E}[Y|X=x]]}{\mathsf{P}[X=x]} + \frac{\mathsf{Var}[\mathbb{E}[Y|X=x']]}{\mathsf{P}[X=x']}}} \ ,$$

*where* $m$ *is the number of traces. This certainly is an expression that is not usual in statistics, and* a priori *cannot be simplified. At the opposite, NICV can be meaningfully expressed as a function of the CPA attack (recall Eqn. (4)).*

## 3.2 Estimation of NICV

The number of traces is $m$, and each trace is indexed by $i$. In the sequel, when we sum over $i$, we mean $i = 1, \ldots, m$. Let $(y_i)_{i=1,\ldots,m}$ be side-channel measurements, and $(x_i)_{i=1,\ldots,m}$ be plaintext bytes. We define the (empirical) probability of a given plaintext byte:

$$\hat{\mathbb{P}}(X = x) = \frac{1}{m} \sum_{i \text{ s.t. } x_i = x} 1 \ .$$

and the (empirical) per-class average, for a given class $x \in \{0, \ldots, 2^n - 1\}$:

$$\hat{\mathbb{E}}(Y|X = x) = \left(\sum_{i \text{ s.t. } x_i = x} y_i\right) / \left(\sum_{i \text{ s.t. } x_i = x} 1\right) = \frac{1}{m\hat{\mathbb{P}}(X = x)} \sum_{i \text{ s.t. } x_i = x} y_i \ ,$$

The numerator of the NICV is:

$$\widehat{\mathrm{Var}}(\hat{\mathbb{E}}(Y|X)) =$$
$$\frac{1}{2^n} \sum_{x=0}^{2^n-1} \hat{\mathbb{P}}(X = x) \left(\hat{\mathbb{E}}(Y|X = x)\right)^2 - \left(\frac{1}{2^n} \sum_{x=0}^{2^n-1} \hat{\mathbb{P}}(X = x) \hat{\mathbb{E}}(Y|X = x)\right)^2 =$$
$$\frac{1}{2^n m} \sum_{x=0}^{2^n-1} \frac{\left(\sum_{i \text{ s.t. } x_i = x} y_i\right)^2}{\sum_{i \text{ s.t. } x_i = x} 1} - \left(\frac{1}{m} \sum_i y_i\right)^2 \ .$$

Eventually, the (empirical) NICV is:

$$\widehat{\mathrm{NICV}} = \frac{\widehat{\mathrm{Var}}(\hat{\mathbb{E}}(Y|X))}{\widehat{\mathrm{Var}}(Y)} = \frac{\frac{1}{2^n m}\sum_{x=0}^{2^n-1}\frac{\left(\sum_{i \text{ s.t. } x_i=x} y_i\right)^2}{\sum_{i \text{ s.t. } x_i=x} 1} - \left(\frac{1}{m}\sum_i y_i\right)^2}{\frac{1}{m}\sum_i y_i^2 - \left(\frac{1}{m}\sum_i y_i\right)^2} \quad . \tag{6}$$

We notice that Eqn. (6) can be estimated online using accumulators.

### 3.3 Lower Bound on the Number of Traces to Break the Key (with CPA)

The success rate ($0 \leq \mathsf{SR} \leq 1$) of an attack is the probability that is recovers the correct key. Regarding CPA, the success rate has been computed theoretically by Thillard, Prouff and Roche [36] (a result that extends the previous analytical formula obtained for the "difference-of-means" test by Fei, Luo and Ding [13]). It is related to the various factors of the experimental setup, namely:

- the signal-to-noise ratio (which is also directly related to the NICV metric – recall Eqn. (5)),
- the sensitive variable expression, which discriminates more or less easily the correct key (which relates to an algorithmic parameter known as the *confusion coefficient* noted $\kappa$),
- the number of traces $m$.

The expression takes the following form:

$$\mathsf{SR} = \Phi(\sqrt{m}\boldsymbol{\Sigma}^{-1/2}\boldsymbol{\mu}) \quad .$$

In this expression, $\Phi$ is the cumulative distribution function of the multivariate Gaussian. Let $N_k$ be the number of key hypotheses. Then $\boldsymbol{\Sigma}$ is a $(N_k-1)\times(N_k-1)$ matrix, and $\boldsymbol{\mu}$ is column of length $(N_k-1)$. In the worst case, the leakage model is very discriminating. This means that the wrong key guesses are all equivalent and yield a null value for the distinguisher. It is a strong assumption, but for good ciphers, such as the AES, this pessimistic approximation is not too far from the reality [23]. In this case, $\boldsymbol{\Sigma}$ degenerates to a $1 \times 1$ matrix (a scalar), equal to: $\boldsymbol{\Sigma} = 2\sigma^2(\kappa_0 - \kappa_1)$. This consists in the application of the *disintegration theorem*, since the covariance matrix has not full rank. The problem is projected on a smaller matrix of size $\mathrm{rank}(\boldsymbol{\Sigma}) \times \mathrm{rank}(\boldsymbol{\Sigma})$ (in our case, $1 \times 1$). In this equation, $\kappa_0$ is a "generalized" confusion coefficient.

Also, $\boldsymbol{\mu}$ is a scalar, equal to $\kappa_0 - \kappa_1$. So, we have:

$$\mathsf{SR} = \Phi\left(\sqrt{m \times \frac{\kappa_0 - \kappa_1}{2\sigma^2}}\right) \quad . \tag{7}$$

**The ratio $\frac{\kappa_0-\kappa_1}{\sigma^2}$ is the signal-to-noise ratio; owing to Eqn. (5), it is equal to $\frac{1}{\frac{1}{\mathrm{NICV}}-1}$.** Confer to [18] for a similar relationship.

So, for a given confidence level in the result of the attack, expressed as targetted success rate $\mathsf{SR}$, we have that the lower bound on the number of traces to break the key is $m_{\mathsf{SR}}$, equal to:

$$m_{\mathsf{SR}} = \frac{2\sigma^2}{\kappa_0 - \kappa_1} \times \left(\Phi^{-1}\left(\mathsf{SR}\right)\right)^2 \quad , \tag{8}$$

where $\Phi^{-1}$ is the reciprocal function of $\Phi$ (recall that $\Phi$ is monotonic decreasing).

The function $\Phi$ can be computed from the "error function" (in C code, `erf`)": $\Phi(y) = \frac{1}{2}\left(\texttt{erf}(y/\sqrt{2}) + 1\right)$. It can be checked that $\Phi^{-1}(0.80) \approx 0.84$.

So, any CPA will require more traces to recover the key than $m_{\mathsf{SR}}$.

The equation (8) has a simple interpretation in a log-log graph, where the $X$ axis is $\sigma$ and the $Y$ axis is $m_{\mathsf{SR}}$. It simply cuts the plan into two halves, that are separated by a straight line (called "frontier"), of equation

$$\begin{aligned}
&\log_2\left(m_{\mathsf{SR}}^{\text{lower bound}}\right) \\
&= 2\log_2\left(\frac{\sigma}{\sqrt{\kappa_0 - \kappa_1}}\right) + 1 + 2\log_2\left(\Phi^{-1}\left(\mathsf{SR}\right)\right) \\
&= 2\log_2\left(\frac{\sigma}{\sqrt{\kappa_0 - \kappa_1}}\right) + 0.497 \text{ when } \mathsf{SR} = 80\%.
\end{aligned} \tag{9}$$

The possible attacks require a number of traces $m_{\mathsf{SR}}$, i.e., such as $m_{\mathsf{SR}}$ is above the "frontier". A more precise characterization follows.

In [36], the confusion coefficient for the CPA is defined as:

$$\begin{aligned}
\kappa_\delta &= \frac{1}{2^n}\sum_{x\in\mathcal{X}} L(x\oplus k) \times L(x\oplus k^\star) \quad \text{Where } \delta = k\oplus k^\star \\
&= \frac{1}{2^n}\sum_{x\in\mathcal{X}} L(x) \times L(x\oplus\delta) = \frac{1}{2^n}(L\otimes L)(\delta) \quad ,
\end{aligned} \tag{10}$$

where $\otimes$ is the convolution function.

We consider the ideal case where $L = w_H(S(X\oplus k)) - \frac{n}{2}$. This function is defined to be balanced, i.e., $\sum_{x\in\mathbb{F}_2^n} L(x) = 0$. In this case, for $\delta = 0$: $\kappa_0 = \frac{1}{2^n}\sum_{x\in\mathcal{X}}\left(w_H(x) - \frac{n}{2}\right)^2 = \frac{n}{4}$, which does not depend on the exact $S$.

The approximation that all incorrect keys yield a zero value for the distinguisher amounts to say that $\forall k \neq 0, \kappa_k = \frac{1}{2^n}(L\otimes L)(k) = 0$.

So, $\kappa_0 = n/4$ (= 2 for AES, where $n = 8$) and $\kappa_1 = 0$, and thus Eqn. (11) becomes:

$$\log_2\left(m_{\mathsf{SR}}^{\text{lower bound}}\right) = 2\log_2\left(\sigma\right) - 0.503 \text{ when } \mathsf{SR} = 80\%. \tag{11}$$

To illustrate the relevance of the lower bound on the number of traces to break ($m_{\mathsf{SR}}$), we launch some simulations (cf. Fig. 2), repeated 100 times to gain a reasonable error margin on the success probability. The values in Fig. 2 are in fact the first time in the simulation that $\mathsf{SR} \geq 0.8$; sometimes, the experimental curve $\mathsf{SR}(m)$ is increasing and then decreasing. This explains why the values

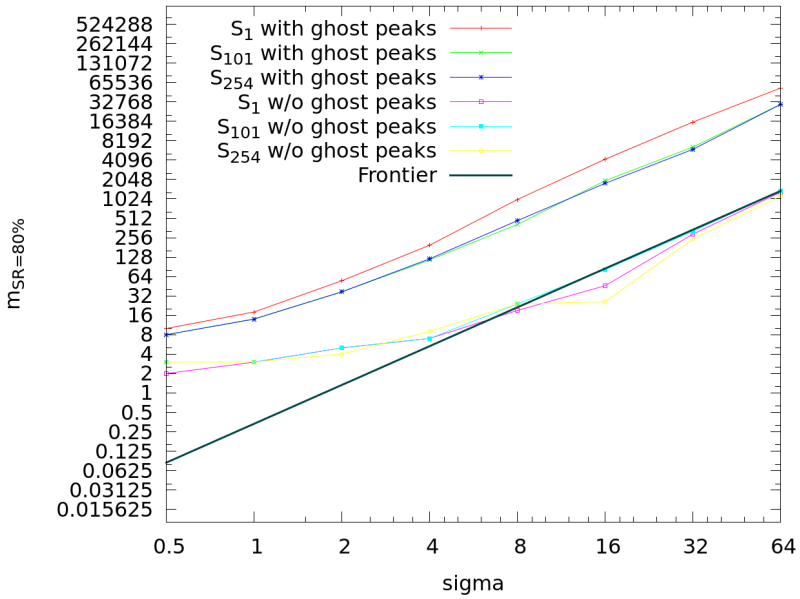



Figure 2: Simulated $m_{\mathsf{SR}=80\%}$

in the graph are sometimes "anticipated" success rates (i.e., lower than what would have been obtained by smoothing the experimental success rate curves before selecting the $m$ such as $\mathsf{SR}(m) = 0.8$).

We assume that the leakage is the Hamming weight of a sensitive variable, that has the form $S(X \oplus k)$. The hypothesis $\forall k \neq 0, \frac{1}{2^n}(L \otimes L)(k) = 0$ is all the more true as $S$ is cryptographically strong. So we investigate different types of bijective sboxes: $S(x) = a \odot x^d \oplus b$, where $a$ and $b$ are the constants also used in the AES. The operations $\oplus$ and $\odot$ are respectively the addition and the multiplication in the Galois field $\mathbb{F}_{2^8}$. We consider three values for $d$:

1. $d = 1$, i.e., an affine function (the least non-linear),
2. $d = 101$, i.e., a medium function, and
3. $d = 254$, i.e., a good function (it is the AES SubBytes).

Those three substitution boxes are respectively noted $S_1$, $S_{101}$ and $S_{254}$. In addition, for each sbox, we compute an *ideal* leakage model $L^{\text{no ghost peak}}(X \oplus k)$, such that:

$$L^{\text{no ghost peak}}(X \oplus k) = \begin{cases} w_H(S(X \oplus k^\star)) - \frac{n}{2} & \text{if } k = k^\star, \text{ or} \\ w_H(S(D)) - \frac{n}{2} & \text{otherwise,} \end{cases}$$

where $D$ is a dummy random variable, independent from $X$ and drawn uniformly in interval $[\![0, 2^n - 1]\!]$.

To summarize, in Fig. 2, we compute for the three different sboxes $S$ the two kinds of CPA:

- With *ghost peaks*:
  $\rho\left(w_H(S(X \oplus k^\star)) + N; w_H(S(X \oplus k))\right)$;
- Without *ghost peaks*:
  $\rho\left(w_H(S(X \oplus k^\star)) + N; L^{\text{no ghost peak}}(X \oplus k)\right)$.

Clearly, the attacks do not depend on the sbox in this later case, because we simply remove the rivals, which brings us to the case of distinguishing from two keys. Quite logically, this attack reaches the theoretical minimum. And we see that the less discriminating the sbox, the more traces are required.

**Remark 4** *The values $\kappa(0) - \kappa(1)$ in Eqn. (7) are computed using the leakage model of Eqn. (10) as: $\kappa(0) - \max_{\delta \neq 0} \kappa(\delta)$ for the three studied substitution boxes. When $n = 8$ (like in the case of AES), it can be proven that whatever the bijective substitution box, $\kappa(0) = n/4 = 2$. The value of $\max_{\delta \neq 0} \kappa(\delta)$ depends on the substitution box. Namely, we have:*

$$\begin{cases} \text{For } S = S_1\text{:} & \kappa(0) - \max_{\delta \neq 0} \kappa(\delta) = 0.5 \ , \\ \text{For } S = S_{101}\text{:} & \kappa(0) - \max_{\delta \neq 0} \kappa(\delta) = 1.5 \ , \\ \text{For } S = S_{254}\text{:} & \kappa(0) - \max_{\delta \neq 0} \kappa(\delta) = 1.625 \ . \end{cases}$$

*Thus, for a given success rate (say 80%) and a given noise variance $\sigma^2$, if the number of traces to break without substitution box (that is, $S = S_1$) is $m_1$, then:*

- *the number of traces to recover the key with $L(x \oplus k) = w_H(S_{101}(x \oplus k))$ would be $m_1 \times \frac{0.5}{1.5} = \frac{1}{3} m_1 \approx 0.333 m_1$, and*
- *the number of traces to recover the key with $L(x \oplus k) = w_H(S_{254}(x \oplus k))$ would be $m_1 \times \frac{0.5}{1.625} = \frac{4}{13} m_1 \approx 0.308 m_1$.*

*This means that the choice of the substitution boxes impacts the number of traces to extract the key in a factor of about 1 to 3.*

## 3.4 Discussion

The mathematical background of NICV as a leakage detection technique was previously discussed. We learned that NICV has evident advantages over other methods because all its input parameters are public like side-channel traces and associated plaintexts/ciphertexts. Since public parameters are used for computation of NICV, there is no need for access to clone device which is a limiting requirement in template-based detection techniques. Another interesting observation is that the expression of NICV (Eq. (2)) does not contain $L(x)$. In other words, NICV is leakage model agnostic. Moreover from Eq. (4), we learn that NICV forms the envelope of all correlation coefficients for all leakage models. NICV provides the worst case leakage of a devise and therefore estimates the accuracy of leakage model used as illustrated in Fig. 1. Thus NICV has a clear application in comparing various leakage models.

# 4 Use Cases

We detailed the theoretical soundness and advantages of NICV as a leakage detection technique in Sec. 3. In this section, we apply NICV in practical side-channel evaluation scenarios. Several use cases of NICV are discussed in the following.

## 4.1 Accelerating Side-Channel Attacks

The main application of NICV is to find the interesting time samples for accelerating SCA. A simple trace of an AES execution can have millions of points. Therefore it is of interest for the evaluator to know few interesting points rather than attacking the whole trace. We first apply the metric on traces of an AES-128 implementation running on an FPGA which performs one round per clock cycle. These traces are small and contain only $1,000$ points. The comparison of our metric with a correlation coefficient computed with the good key is shown in Fig. 3. The correlation is based on Hamming distance model of the state register of the AES core. The model can be expressed as $w_H(val_i \oplus val_f)$ where $val_i$ and $val_f$ are initial and final value of the register. This leakage model is shown to be very efficient in CMOS technology. A relevant peak of NICV is seen at the same moment as in correlation peak. Thus NICV is able to detect the point of leakage in the trace. It can be noticed that NICV curve shows several other peaks apart from the correlation peak. As shown later in Sec. 4.2, other peaks in the NICV curve comes either from other leakage models or post-processing of cipher in the circuit.

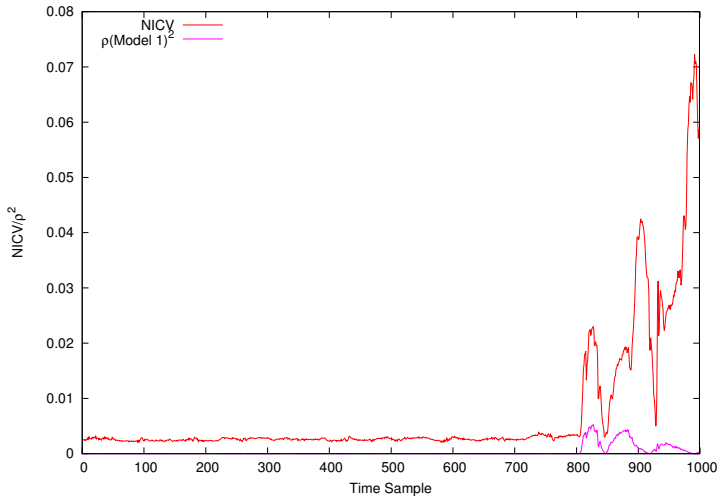



Figure 3: NICV vs Correlation for a AES-128 hardware implementation

Next we apply our metric on a software implementation of AES-256 running

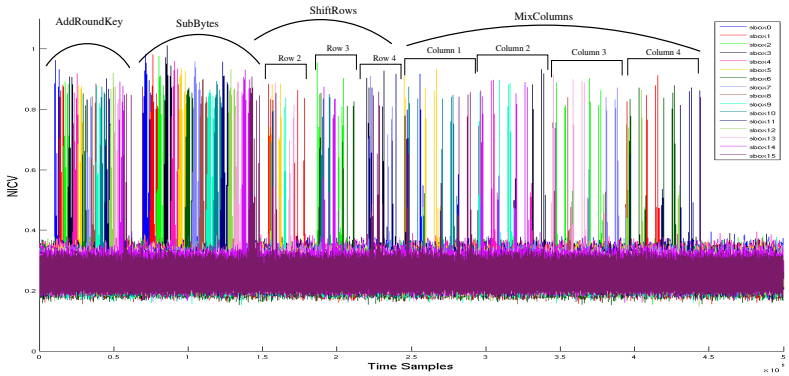



Figure 4: NICV computed for a AES-128 software implementation to detect each round operation

on an ATMEL AVR microcontroller. It is here that we can see the advantage of NICV. A single trace of this implementation contains 7 million points and needs roughly 5.3 Mbytes of disk space when stored in the most compressed format. These details are in respect to a LeCroy WaveRunner 6100A oscilloscope with a bandwidth of 1 GHz. We applied NICV on these traces to find the leakage points related to each of the 16 bytes of the AES. Fig. 4 shows the computation of NICV on the first round only (for better resolution of results). The computations of Sbox #0 for round 1 takes only $\approx 1,000$ time samples. Once the interesting time samples corresponding to each executed operation is known, the trace size is compressed from $7,000,000$ to $1,000$, i.e., a gain of roughly $7,000\times$ in attack time.

One very interesting application of NICV that we found during our experiments is to reverse engineering. We computed NICV for all the 16 bytes of the plaintext and plotted the 16 NICV curves in Fig. 4 (depicted in different colors). By closely observing Fig. 4, we can distinguish individual operations from the sequence of byte execution. Each NICV curve (each color) shows all sensitive leakages related to that particular byte. Moreover, with a little knowledge of the algorithm, one can easily follow the execution of the algorithm. For example, the execution of all the bytes in a particular sequence indicates the SubBytes or AddRoundKey operation. Manipulation of bytes in sequence $\{1, 5, 9, 13\}$, $\{2, 6, 10, 14\}$ and $\{3, 7, 11, 15\}$ indicates the ShiftRows operations. The ShiftRows operation of AES shifts circularly 3 out of 4 rows with different constant. This can be clearly seen is Fig. 4: only three rows are manipulated and the bytes in the first row i.e., $\{0, 4, 8, 12\}$ are not used during this time. Similarly MixColumns can also be identified by just looking the bytes manipulated together. Moreover, detecting precise leakage points of each operation can help an attacker run collision attacks (see e.g., [27, 26]).

### 4.2 Testing Leakage Models

A common problem in SCA is the choice of leakage model which directly affects the efficiency of the attack. As shown in Sec. 3.1, the square of the correlation between modeled leakage ($L(X, K)$) and traces ($Y = L(X, K^\star) + N$) is smaller or equal to NICV, where $N$ represents a noise. The equality exists only if the modeled leakage is the same as the traces. We tested two different leakage models for the state register resent before the Sbox operation of AES, i.e., $w_H(val_i \oplus val_f) \in [\![0, 8]\!]$ (Model 1) and $val_i \oplus val_f \in [\![0, 255]\!]$ (Model 2). Similar models are built for another register intentionally introduced at the output of the Sbox, i.e., $w_H(S(val_i) \oplus S(val_f)) \in [\![0, 8]\!]$ (Model 3) and $S(val_i) \oplus S(val_f) \in [\![0, 255]\!]$ (Model 4). We implemented the AES on an FPGA and acquired SCA traces to compare the leakage models. Fig. 5 shows the square of correlation of four different leakage models with the traces against the NICV curve. It can be simply inferred from Fig. 5(b) that Model 4 performs the best while Model 2 is the worst. The gap between NICV and $\rho(\text{Model 4})^2$ is quite large, most probably due to reasons mentioned in Sec. 3.1. This means that there exist other leakage models which could perform better than Model 4. However, finding these models might not be easy because of limited knowledge of design and device characteristics available. Methods based on linear regression [11] can be leveraged to determine the most relevant leakage model.

### 4.3 Comparing Quality of Measurements

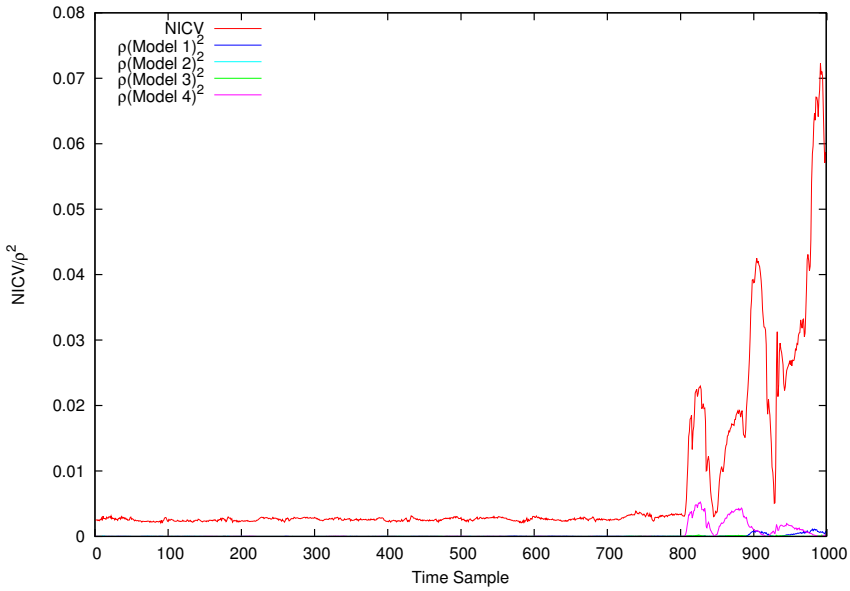
SNR is often used to estimate the quality of a measurement setup/traces to compare different measurement setups. The problem with SNR is that it is computed using a specific leakage model. NICV is a good candidate for quality comparison owing to the independence from choice of leakage model.

### 4.4 Ghost Peaks

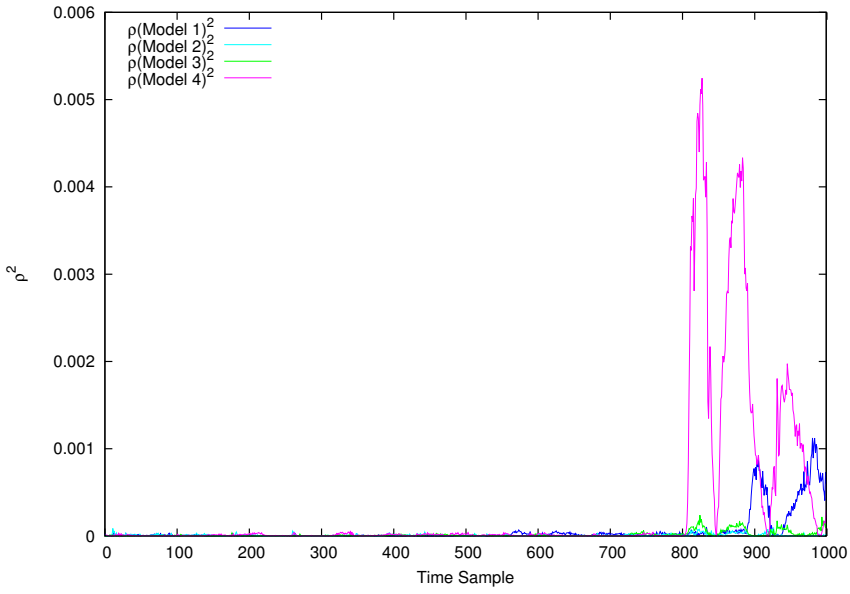
To further demonstrate the advantageous use of NICV, we implemented a bit-slice implementation (i.e., a single bit of plaintext is processed per cycle) of PRESENT algorithm on an ATMEL AVR microcontroller. Firstly we used NICV to localize the activity of an sbox in a multi-million sample trace. Figure 6 (top) shows the result of a CPA on the least significant bit with $10,000$ traces with an SNR of 15.8. Next we plot the NICV computed with respect to the input plaintext byte in Fig. 6 (bottom). If we compute the success rate of an attack on the whole trace, the ghost peak around sample 900 leads to failure of the attack. However if we compute a success rate on a window around samples where NICV is maximal, we are able to suppress the ghost peak. The success rate of an attack around sample 800, where NICV is maximal is shown in Fig. 7.

### 4.5 Accelerating SCA on Asymmetric Key Cryptography

Asymmetric key cryptography consists in computing exponentiations. For example, in RSA [30], the computation consists in $X^d$ (modulo $N$) from $X$. For




(a)




(b)

Figure 5: (a) NICV vs $\rho^2$ of four different models, (b) and its zoom

the sake of simplicity, let us consider a right-to-left exponentiation. Such ex-

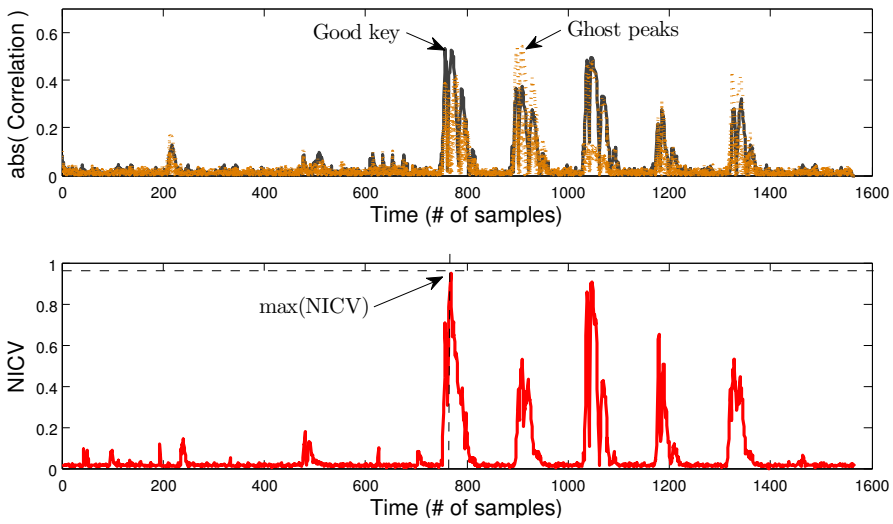



Figure 6: NICV vs Correlation depicting ghost peaks for a software implementation of PRESENT protected by DPL (Dual-rail Precharged Logic)

ponentiation is illustrated in Alg. 1, where $N$ is the modulus (e.g., that fits on 1024 bits), and $R[1]$ and $R[2]$ are two 1024 bit temporary registers. Let us call $d_i$ the 1024 bits of $d$. We assume $d_0 = 1$.

Hence the number $X^3$ will be computed (in $R[1]$; refer to line 5) if and only if $d_1 = 1$. This conditional operation is at the basis of the SCA on RSA [25]: if a correlation between the traces $Y$ and the prediction $L(X) = X^3$ exists, then $d_1 = 1$; otherwise, $d_1 = 0$. For this alternative to be tested with NICV, one should compute $Y|X^3$, where $X^3$ (modulo $N$) is a large number (e.g., $1,024$ bits). To be tractable, small parts of $X^3$ like the least significant byte (LSB) shall be used instead of $X^3$. In this case, a leakage can be detected by computing $\mathsf{Var}\left[\mathbb{E}\left[Y|\mathrm{LSB}(X^3)\right]\right]/\mathsf{Var}\left[Y\right]$. The corresponding attack would use the prediction function $L(X) = \mathrm{LSB}(X^3)$.

For sure, the test is relevant only if the bit $d_1$ is set in the private key $d$. But if it is not, then maybe $d_2$ is set. In this case, a leakage can be detected by computing
$\mathsf{Var}\left[\mathbb{E}\left[Y|\mathrm{LSB}(X^5)\right]\right]/\mathsf{Var}\left[Y\right]$. Similarly, if $d_1 = d_2 = 0$, it is plausible that $d_3 = 1$, and thus $X^9$ is computed. Thus, it is sufficient, in order to detect a leakage to compute
$\mathsf{Var}\left[\mathbb{E}\left[Y|\mathrm{LSB}(X^{2^i+1})\right]\right]/\mathsf{Var}\left[Y\right]$ for a couple of small $i > 0$. Any significant peak indicates a potential vulnerability.

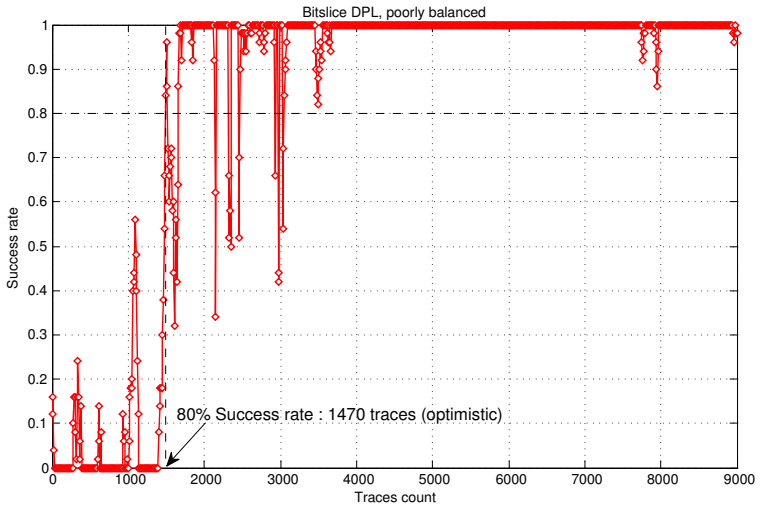



Figure 7: Success rate of CPA on a software implementation of PRESENT protected by DPL after suppression of ghost peaks using NICV

If, for example, the 10 NICV quantities $\mathsf{Var}\left[\mathbb{E}\left[Y|\mathrm{LSB}(X^{2^i+1})\right]\right]/\mathsf{Var}\left[Y\right]$, for $1 \leq i \leq 10$, are computed (without knowing the key $d$), then a vulnerability is detected with probability $1 - 2^{-10}$ (indeed, $2^{-10}$ is the probability of having $d_1 = \ldots = d_{10} = 0$). This methodology is illustrated in Fig. 8.

The methodology has been validated in practice, on RSA with an exponent $d = (\ldots 111)_2$, i.e., finishing with three bits at 1. In Fig. 9, we have computed NICV by partitioning on $\mathrm{LSB}(X^3) \in \{0, \ldots, 255\}$. The NICV indeed features a peak, which validates that $d_1 = 1$, and that the device leaks. As the peak is really sharp, the leak is strong (our implementation is an ATMega163 smartcard, which is known to leak a lot). For cross-checking, we also plotted in Fig. 9:

- NICV on $\mathrm{LSB}(X^5)$, which does not feature a peak (as expected). It would have had a peak if the exponent would have been $d = (\ldots 101)_2$.
- NICV on $\mathrm{LSB}(X^7)$, which has a peak, as expected.

# 5 Conclusions and Perspectives

We presented NICV as a leakage detection technique for side-channel leakage. NICV uses public information like plaintext or ciphertext for detecting leakage and trace compression. Therefore NICV has a low computation footprint which can be considered as the worst case leakage analysis which *envelops* correlation coefficient of all possible leakage models. However NICV cannot be used directly

**Algorithm 1:** Unprotected right-to-left 1024 bit RSA implementation

**Input** : $X \in \mathbb{Z}_N, d = (d_{1023}, \cdots, d_0)_2$
**Output**: $X^d \in \mathbb{Z}_N$

```
1 R[1] ← 1
2 R[2] ← X
3 for i ∈ [[0, 1023]] do
4 |   if d_i = 1 then
5 |   |   R[1] ← R[2] · R[1]                    /* Multiply */
6 |   end
7 |   R[2] ← R[2] · R[2]                        /* Square */
8 end
9 return R[1]
```

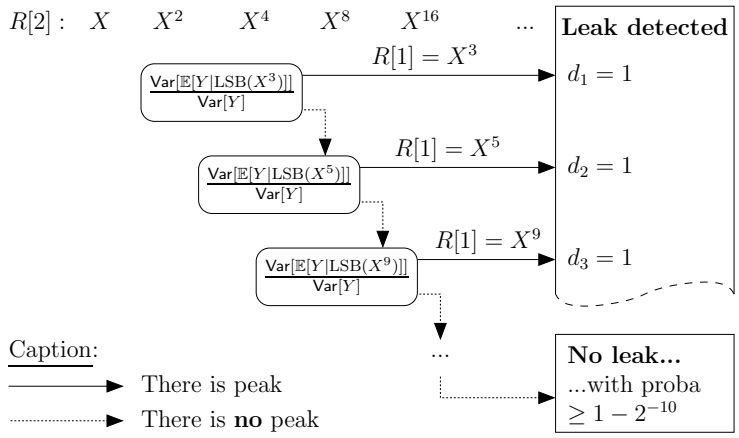



Figure 8: Illustration of the application of NICV to RSA

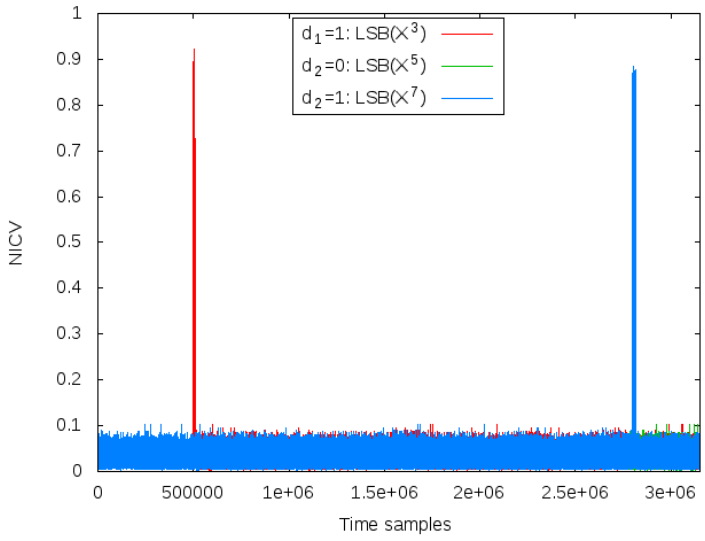



Figure 9: Result of the methodology of NICV applied to RSA (first two steps of Fig. 8)

as a distinguisher for an attack. Unlike templates, NICV can operate on the same set of traces which are used for attack. We can also use NICV to theoretically estimate the lower bound on number of traces to attack an implementation with CPA. We demonstrated the power of NICV in several use cases related to SCA like detecting relevant time samples, comparing leakage models, detecting ghost peaks, etc.

Future works can focus on extending the power of NICV in detecting higher-order leakage and extensive application to asymmetric key cryptography.

## Acknowledgements

The authors thank Yip Ching Yu Henry (DSO National Laboratories, Singapore), Yongbin Zhou (State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences), Hirofumi Sakane (AIST, Japan) and Emmanuel Prouff (ANSSI, France) for interesting discussions about the leakage metrics, and especially for comments regarding the comparison of NICV and TVLA. Besides, we thank Youssef Souissi (Secure-IC S.A.S.) for more examples of NICV, used as *correlation ratio*.

# References


[1] M. A. E. Aabid, S. Guilley, and P. Hoogvorst. Template Attacks with a Power Model. Cryptology ePrint Archive, Report 2007/443, December 2007. http://eprint.iacr.org/2007/443/.

[2] C. Archambeau, É. Peeters, F.-X. Standaert, and J.-J. Quisquater. Template Attacks in Principal Subspaces. In *CHES*, volume 4249 of *LNCS*, pages 1–14. Springer, October 10-13 2006. Yokohama, Japan.

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

[4] É. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In *CHES*, volume 3156 of *LNCS*, pages 16–29. Springer, August 11–13 2004. Cambridge, MA, USA.

[5] S. Chari, J. R. Rao, and P. Rohatgi. Template Attacks. In *CHES*, volume 2523 of *LNCS*, pages 13–28. Springer, August 2002. San Francisco Bay (Redwood City), USA.

[6] O. Choudary and M. G. Kuhn. Efficient Template Attacks. Cryptology ePrint Archive, Report 2013/770, 2013. http://eprint.iacr.org/2013/770.

[7] C. C. Consortium. Common Criteria (*aka* CC) for Information Technology Security Evaluation (ISO/IEC 15408), 2013.
Website: http://www.commoncriteriaportal.org/.

[8] J. Cooper, G. Goodwill, J. Jaffe, G. Kenworthy, and P. Rohatgi. Test Vector Leakage Assessment (TVLA) Methodology in Practice, Sept 24–26 2013. International Cryptographic Module Conference (ICMC), Holiday Inn Gaithersburg, MD, USA.

[9] J.-S. Coron, P. C. Kocher, and D. Naccache. Statistics and Secret Leakage. In *Financial Cryptography*, volume 1962 of *Lecture Notes in Computer Science*, pages 157–173. Springer, February 20-24 2000. Anguilla, British West Indies.

[10] J.-L. Danger, N. Debande, S. Guilley, and Y. Souissi. High-order Timing Attacks. In *Proceedings of the First Workshop on Cryptography and Security in Computing Systems*, CS2 '14, pages 7–12, New York, NY, USA, 2014. ACM.

[11] J. Doget, E. Prouff, M. Rivain, and F.-X. Standaert. Univariate side channel attacks and leakage modeling. *J. Cryptographic Engineering*, 1(2):123–144, 2011.

[12] R. J. Easter, J.-P. Quemard, and J. Kondo. Text for ISO/IEC 1st CD 17825 – Information technology – Security techniques – Non-invasive attack mitigation test metrics for cryptographic modules, March 22 2014. Prepared within ISO/IEC JTC 1/SC 27/WG 3. (Online).

[13] Y. Fei, Q. Luo, and A. A. Ding. A Statistical Model for DPA with Novel Algorithmic Confusion Analysis. In E. Prouff and P. Schaumont, editors, *CHES*, volume 7428 of *LNCS*, pages 233–250. Springer, 2012.

[14] B. Gierlichs, K. Lemke-Rust, and C. Paar. Templates vs. Stochastic Methods. In *CHES*, volume 4249 of *LNCS*, pages 15–29. Springer, October 10-13 2006. Yokohama, Japan.

[15] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi. A testing methodology for side-channel resistance validation, September 2011. NIST Non-Invasive Attack Testing Workshop, http://csrc.nist.gov/news_events/non-invasive-attack-testing-workshop/papers/08_Goodwill.pdf.

[16] S. Guilley, S. Chaudhuri, L. Sauvage, P. Hoogvorst, R. Pacalet, and G. M. Bertoni. Security Evaluation of WDDL and SecLib Countermeasures against Power Attacks. *IEEE Transactions on Computers*, 57(11):1482–1497, nov 2008.

[17] S. Guilley, R. Nguyen, and L. Sauvage. Non-Invasive Attacks Testing: Feedback on Relevant Methods, Sept 24–26 2013. International Cryptographic Module Conference (ICMC), Holiday Inn Gaithersburg, MD, USA.

[18] S. Hajra and D. Mukhopadhyay. SNR to Success Rate: Reaching the Limit of Non-Profiling DPA. Cryptology ePrint Archive, Report 2013/865, 2013. http://eprint.iacr.org/2013/865/.

[19] A. Heuser, W. Schindler, and M. Stöttinger. Revealing side-channel issues of complex circuits by enhanced leakage models. In W. Rosenstiel and L. Thiele, editors, *DATE*, pages 1179–1184. IEEE, 2012.

[20] I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics, 2002. ISBN: 0387954422.

[21] P. Karsmakers, B. Gierlichs, K. Pelckmans, K. D. Cock, J. Suykens, B. Preneel, and B. D. Moor. Side channel attacks on cryptographic devices as a classification problem. COSIC technical report, 2009.

[22] V. Lomné, E. Prouff, and T. Roche. Behind the scene of side channel attacks. In K. Sako and P. Sarkar, editors, *ASIACRYPT (1)*, volume 8269 of *LNCS*, pages 506–525. Springer, 2013.

[23] S. Mangard. Hardware Countermeasures against DPA – A Statistical Analysis of Their Effectiveness. In *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2004. San Francisco, CA, USA.

[24] S. Mangard, E. Oswald, and F.-X. Standaert. One for All - All for One: Unifying Standard DPA Attacks. *Information Security, IET*, 5(2):100–111, 2011. ISSN: 1751-8709 ; Digital Object Identifier: 10.1049/iet-ifs.2010.0096.

[25] T. S. Messerges, E. A. Dabbish, and R. H. Sloan. Power Analysis Attacks of Modular Exponentiation in Smartcards. In Ç. K. Koç and C. Paar, editors, *CHES*, volume 1717 of *LNCS*, pages 144–157. Springer, 1999.

[26] A. Moradi, S. Guilley, and A. Heuser. Detecting Hidden Leakages. In I. Boureanu, P. Owesarski, and S. Vaudenay, editors, *ACNS*, volume 8479. Springer, June 10-13 2014. 12th International Conference on Applied Cryptography and Network Security, Lausanne, Switzerland.

[27] A. Moradi, O. Mischke, and T. Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *CHES*, volume 6225 of *Lecture Notes in Computer Science*, pages 125–139. Springer, August 17-20 2010. Santa Barbara, CA, USA.

[28] E. Prouff, M. Rivain, and R. Bevan. Statistical Analysis of Second Order Differential Power Analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009.

[29] M. Renauld, F.-X. Standaert, N. Veyrat-Charvillon, D. Kamel, and D. Flandre. A Formal Study of Power Variability Issues and Side-Channel Attacks for Nanoscale Devices. In *EUROCRYPT*, volume 6632 of *LNCS*, pages 109–128. Springer, May 2011. Tallinn, Estonia.

[30] R. L. Rivest, A. Shamir, and L. M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[31] A. Roche, G. Malandain, X. Pennec, and N. Ayache. The Correlation Ratio as a New Similarity Measure for Multimodal Image Registration. In W. M. W. III, A. C. F. Colchester, and S. L. Delp, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI'98, First International Conference, Cambridge, MA, USA, October 11-13, 1998, Proceedings*, volume 1496 of *Lecture Notes in Computer Science*, pages 1115–1124. Springer, 1998.

[32] W. Schindler, K. Lemke, and C. Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In LNCS, editor, *CHES*, volume 3659 of *LNCS*, pages 30–46. Springer, Sept 2005. Edinburgh, Scotland, UK.

[33] Y. Souissi, N. Debande, S. Mekki, S. Guilley, A. Maalaoui, and J.-L. Danger. On the Optimality of Correlation Power Attack on Embedded Cryptographic Systems. In I. G. Askoxylakis, H. C. Pöhls, and J. Posegga, editors, *WISTP*, volume 7322 of *Lecture Notes in Computer Science*, pages 169–178. Springer, June 20-22 2012.

[34] 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.

[35] F.-X. Standaert, B. Gierlichs, and I. Verbauwhede. Partition vs. Comparison Side-Channel Distinguishers: An Empirical Evaluation of Statistical Tests for Univariate Side-Channel Attacks against Two Unprotected CMOS Devices. In *ICISC*, volume 5461 of *LNCS*, pages 253–267. Springer, December 3-5 2008. Seoul, Korea.

[36] A. Thillard, E. Prouff, and T. Roche. Success through Confidence: Evaluating the Effectiveness of a Side-Channel Attack. In G. Bertoni and J.-S. Coron, editors, *CHES*, volume 8086 of *Lecture Notes in Computer Science*, pages 21–36. Springer, 2013.

[37] C. Whitnall, E. Oswald, and F.-X. Standaert. The Myth of Generic DPA ... and the Magic of Learning. In J. Benaloh, editor, *CT-RSA*, volume 8366 of *Lecture Notes in Computer Science*, pages 183–205. Springer, 2014.

[38] D. W. Zimmerman, B. D. Zumbo, and R. H. Williams. Bias in Estimation and Hypothesis Testing of Correlation. *Psicológica*, 24:133–158, 2003.

# A NICV *vs.* TVLA

NICV and TVLA are two methods to quantify the leakage of a cryptographic module (CM). They have been discussed recently at the first International Cryptographic Module Conference (09/2013), respectively in presentations [17] and [8].

TVLA consists in operating the CM with *a fixed key* (specified), exercised under a *fixed input message*, and under *varying messages* [15, Sec. 3]. Then, a T-test (See Eqn. (1), for $n = 2$ classes) is applied on both sets of measurements.

## A.1 Differences

TVLA requires the definition of a set of keys and messages to be tested. This has some side effects. First, it is not always possible to set a key or an input message in a CM; indeed, the key is often initialized randomly or agreed by a protocol, and in most modes of operation, the input message depends upon an initialization vector (IV) that cannot be chosen. Second, this is restrictive as a general method, because new or regional algorithms are created frequently, hence the key / message database of TVLA must keep track of this evolution. Third and the criteria for the selection of keys & inputs is delicate to explain, hence possibly suspicious. At the opposite, with NICV, the device can be tested with its operational key and without chosen inputs.

TVLA tests whether the leakage varies for *two* different input types of the CM (hence the computation of a *difference*), namely varying vs fixed input

messages, whereas NICV tests whether the leakage varies of *a multiplicity of* different inputs (hence the computation of a *variance*). Also, NICV can focus on each input bit ($n = 1$) or byte ($n = 8$).

## A.2 Common Points

As underlined in Remark 2, SOSD and the inter-class variance are proportional. Thus SOST (normalized version of SOSD) and NICV are related, for binary partitionings (i.e., when there are only $n = 2$ classes).

Also, both NICV and TVLA can have an estimation of their bias: the confidence level of NICV is discussed in [23, 38], and in [8] for TVLA.

As common drawbacks, simple countermeasures (like desynchronization) almost zero NICV and SOST, despite efficient preprocessing techniques (like realignment) manage to undo them. Therefore NICV and TVLA do not prove the absence of leakage, but provide a systematic way to detect obvious design errors. We notice that both metrics can be used throughout the development process of a CM, as a non-regression test.