Defensive Leakage Camouflage

E. Brier[1], Q. Fortier[2], R. Korkikian[3,4], K. W. Magld[5]
D. Naccache[2,4], G. Ozari de Almeida[3,4], A. Pommellet[2]
A. H. Ragab[5], and J. Vuillemin[2,5]

[1] Ingenico
1, rue Claude Chappe, BP 346, F-07503 Guilherand-Granges, France.
[2] École normale supérieure, Département d'informatique
45, rue d'Ulm, F-75230, Paris Cedex 05, France.
[3] Altis Semiconductor
224 Boulevard John Kennedy, F-91100, Corbeil-Essonnes, France.
[4] Sorbonne Universités – Université Paris II
12 place du Panthéon, F-75231, Paris Cedex 05, France.
[5] King Abdulaziz University, Jeddah, Saudi Arabia

Abstract. This paper considers the transfer of digital data over *leaky and noisy* communication channels. We develop defensive strategies exploiting the fact that noise prevents the attacker from accurately measuring leakage.
The defense strategy described in this paper pairs each useful data element k with a camouflage value v and simultaneously transmits both k and v over the channel. This releases an emission $e(k, v)$. We wish to select the camouflage values $v(k)$ as a function of k in a way that makes the quantities $e(k, v(k))$ as *indistinguishable* as possible from each other.
We model the problem and show that optimal camouflage values can be computed from side-channels under very weak physical assumptions. The proposed technique is hence applicable to a wide range of readily available technologies.
We propose algorithms for computing optimal camouflage values when the number of samples per trace is moderate (typically ≤ 6) and justify our models by a statistical analysis.
We also provide experimental results obtained using FPGAs.

1 Introduction

In addition to its usual complexity postulates, cryptography silently assumes that secrets can be physically protected in tamper-proof locations. All cryptographic operations are physical processes where data elements must be represented by physical quantities in physical structures.

At any given point in the evolution of a technology, the smallest logic devices must have a definite physical extent, require a certain minimum time to perform their function and dissipate a minimal switching energy when transiting from one state to another. Energy is also dissipated statically, *i.e.* in the absence of any switching.

During the last twenty years, the research community devised many sophisticated methods for retrieving secret information from circuits by measuring their side-channel emanations.

A number of authors, *e.g.* [1], rely on the isotropic switching-model in which all bits dissipate identical switching energies. This work does not assume any *a priori* side-channel model and totally relies on the analysis of actually measured[6] emissions.

[6] potentially anisotropic

While most previous works analyzed leakage from complex cryptographic computations, we focus on one of the simplest forms of leakage: the emanations of a bus through which bits are being sent. We make only two physical assumptions:

- Emanations can be measured with equal (in)accuracy by both the attacker and the defender.
- Leakage is a global function of data plus noise. The proposed methods are thus unadapted to settings in which individual channel bits are probed with precision.

The proposed methodology is hence applicable to a wide range of circuits having *leaky* buses.

The proposed countermeasure pairs each useful data element k with a camouflage value v and simultaneously transmits k and v through the channel. This releases a physical side-channel emanation $e(k, v)$ that can be measured by both the attacker and the defender.

We address the following question:

How can a defender pair each value of k with a corresponding value $v(k)$ that makes the $e(k, v(k))$ as *indistinguishable* as possible from each other?

The crux of this paper is the definition of *indistinguishability* given the measured emissions.

Section 2 introduces algorithms for computing optimal camouflage values from actual power traces. These algorithms are efficient when each trace contains a few samples (typically ≤ 6). Section 3 presents a statistical analysis justifying the intuition that the best v values are those concentrating the $e(k, v)$ into the smallest possible sphere containing representatives of all k values. Section 4 provides experimental results.

In a way, this work achieves some sort of cryptographic key exchange based on the existence of ambient noise and on a gap in measurement accuracy between the legitimate receiver and the attacker.

2 Models and Algorithms

Let $e(d)$ represent the side-channel (*e.g.* power consumption) resulting from the transfer of an n-bit data element d over an n-bit channel (*e.g.* a bus). $e(d)$ can be measured with equal precision by both the attacker and the defender.

The defender builds a set of 2^n side-channel measurements $\mathcal{E}$. Each $e(d) \in \mathcal{E}$ is generated by transmitting an n-bit data element d. The defender assigns s channel bits to the useful information k, and devotes the remaining $n-s$ bits to the transmission of $(n-s)$-bit camouflage values $v(k)$. We denote $d = k|v$ and call the k's *"keys"* or *"colors"*. Note that key bits and camouflage bits are not necessarily adjacent and might be interleaved.

Let $e(k, v) = e(d)$ be the emanation released by transmitting $d = k|v$.

The vector $V = [v(0), \ldots, v(2^s - 1)]$ of all camouflage values must be chosen to make all emanations $e(k, v(k))$ look "as similar as possible". Our goal is to infer V from $\mathcal{E}$.

We assign a unique *color* $k = \text{color}(e(k|v))$ to each $e(i) \in \mathcal{E}$. $\mathcal{E}$ is hence analogous to a multidimensional cloud of 2^n colored points (*i.e.* 2^s sets of colored points; each of these 2^s sets contains 2^{n-s} identically colored points).

A *color-spanning sphere* is a subset $\mathcal{B} \subset \mathcal{E}$ containing at least one emission of each color.

The defender will use the 2^n elements of $\mathcal{E}$ to select 2^s transmittable $k|v(k)$ values forming a color-spanning sphere $\mathcal{A}(V) \subset \mathcal{E}$. The attacker will only get to see traces belonging to $\mathcal{A}(V)$:

$$\mathcal{A}(V) = \bigcup_{k=1,\ldots,2^s-1} \{e \in \mathcal{E} : \text{color}(e) = k\}$$

The defender's goal is to minimize the *size* of the color-spanning sphere $\mathcal{A}(V)$ exposed to the attacker. *i.e.* infer from $\mathcal{E}$ a *smallest color-spanning sphere* $\mathcal{A}_{\text{optimal}}$ such that

$$\|\mathcal{A}_{\text{optimal}}\| = \min_V \|\mathcal{A}(V)\|$$

$\mathcal{A}_{\text{optimal}}$ has thus the least size for all choices of V.

The next section considers the simplest setting where emanations are scalars[7]. In that case the difference $|e - e'|$ between two scalars $e, e' \in \mathcal{E}$ can be used as a similarity measure for constructing $\mathcal{A}_{\text{optimal}}$ efficiently.

2.1 One Dimension

Assume that the $e(d)$ are scalars (*e.g.* execution times or a unique power measurement per trace). Acquire the 2^n reference traces:

$$\mathcal{E} = \{e(0), \ldots, e(2^n - 1)\}$$

A given choice of $V = [v(0), \ldots, v(2^s - 1)]$ restricts the attacker's information to

$$\mathcal{A}(V) = \{e(0, v(0)), \ldots, e(2^{s-1}, v(2^{s-1}))\}$$

The defender's goal is to minimize:

$$\|\mathcal{A}(V)\| = \max \mathcal{A}(V) - \min \mathcal{A}(V) = \max_k(e(k, v(k))) - \min_k(e(k, v(k)))$$

Let $\mathcal{P} = [p_0 \leq p_1 \leq \cdots \leq p_{2^n-1}]$ be the $e(i) \in \mathcal{E}$ sorted (with repetitions) by increasing scalar values. A *color-spanning segment* is an interval of $\mathcal{P}$ containing at least one p_i of each color.

A straightforward algorithm for finding $\mathcal{A}_{\text{optimal}}$ consists in working with two pointers start and end representing the beginning and the end of the segment under evaluation. When execution begins, start and end point at p_0. While [start,end] is not a color-spanning segment end is moved to the right. When end reaches p_{2^n-1} start is moved by one position to the right (*i.e.* from p_i to p_{i+1}) and end is moved back to start. Throughout this process, whenever a shorter color-spanning segment is found, it is recorded. The complexity of this algorithm is quadratic in the cardinality of $\mathcal{E}$, *i.e.* $O(2^{2n})$.

More clever approaches allow to solve the problem in $\tilde{O}(2^n)$. To do so build the 2^s sorted sequences (with repetitions) of emissions for each color:

$$\mathcal{P}^k = [p_0^k \leq \ldots \leq p_{2^{n-s}-1}^k] \text{ for } k = 1, \ldots, 2^s - 1$$

[7] *e.g.* execution times or a unique power measurement per trace.

Represent the *color-spanning* segments by a binary search tree $\mathcal{T}$ of size 2^s.

At step 0, initialize the tree to $\mathcal{T}_0 = \{p_0^0, \ldots, p_0^{2^s-1}\}$ and proceed by 2^s-way merging.

At stage t, the color-spanning tree is

$$\mathcal{T}_t = \left\{ p_{\lambda_t^0}^0, \ldots, p_{\lambda_t^{2^s-1}}^{2^s-1} \right\}$$

where the λ_t^k denote the merge pointers.

Let $\underline{m}$ and $\overline{m}$ denote (respectively) the minimal and maximal scalars in $\mathcal{T}_t$. We denote by ϕ_t the minimal (*i.e.* best) segment length found at step t.

If $t = 0$ or $\overline{m} - \underline{m} < \phi_{t-1}$, then update $\phi_t = \overline{m} - \underline{m}$ else $\phi_t = \phi_{t-1}$.

Let $\underline{m} = p_{\lambda_t^c}^c$ and let $m = p_{\lambda_t^c+1}^c$ be the next emission of the same color. The next tree $\mathcal{T}_{t+1}$ is obtained by replacing $\underline{m}$ by m in $\mathcal{T}_t$. *i.e.* we increase $\lambda_{t+1}^c = \lambda_t^c + 1$ and stall all other merge pointers $\lambda_{t+1}^k = \lambda_t^k$ for $k \neq c$.

The algorithm terminates (at some step $\tau < 2^n$) when it fails to find a successor m to $\underline{m}$. The length of the minimal color-spanning segment is then ϕ_τ.

Complexity: Partitioning $\mathcal{E}$ to 2^s color subsets and sorting these subsets to get the $\mathcal{P}^k$ costs $O(n2^n)$.

Binary search trees [5] support the operations (insert, find-min, extract-min and find-max) required by the structure $\mathcal{T}$, each of these operations requires $O(s)$ time. It follows that the 2^s-way merge runs in $O(s2^n)$ and hence the above algorithm has an overall complexity of $\tilde{O}(2^n)$.

2.2 Higher Dimensions

We now consider the general case where e is a T-dimensional vector, *e.g.* a power consumption sampled at T different instants. $\mathcal{E}$ is now a T-dimensional cloud of colored points (Fig. 1) and the color spanning interval is a T-dimensional sphere. We need to determine the smallest sphere containing at least one point of each color *i.e.* the smallest color-spanning sphere $\mathcal{A}_{\text{optimal}}$ (Fig. 2, right).

The cloud of points is contained in some minimal enclosing T-dimensional rectangle $\mathcal{R}$, whose sides are parallel to the hyperspace's T axes (Fig. 3, right).

Divide and Conquer This problem lends itself to divide and conquer resolution.

Let $\mathcal{B}$ be some[8] initial color spanning sphere of radius r. Let ℓ denote the length of the rectangle $\mathcal{R}$ along some dimension x. Split $\mathcal{R}$ along the x axis into two overlapping sub-rectangles of lengths $\frac{\ell}{2}+r$ as shown by Figure 4. Let $\mathcal{R}_{\text{right}}$ and $\mathcal{R}_{\text{left}}$ be the two equally sized sub-rectangles obtained that way (Fig. 5).

By construction, $\mathcal{A}_{\text{optimal}}$ is fully contained in either $\mathcal{R}_{\text{right}}$ or $\mathcal{R}_{\text{left}}$. So, we recursively apply the process to $\mathcal{R}_{\text{right}}$ and $\mathcal{R}_{\text{left}}$ until splitting diminishes the rectangles' sizes only negligibly[9]. At that point we solve each of the smaller instances (by any chosen method) and output the

[8] not necessarily optimal, *cf.* Fig. 3, left.

[9] After the w-th splitting the rectangles' sides are of size $\ell_w = (\ell - 2r)2^{-w} + 2r$. Hence splitting can last forever. We suggest to stop splitting when $\ell_w < 3r$ *i.e.* after $\lfloor \log_2(\ell/r - 2) \rfloor$ iterations.

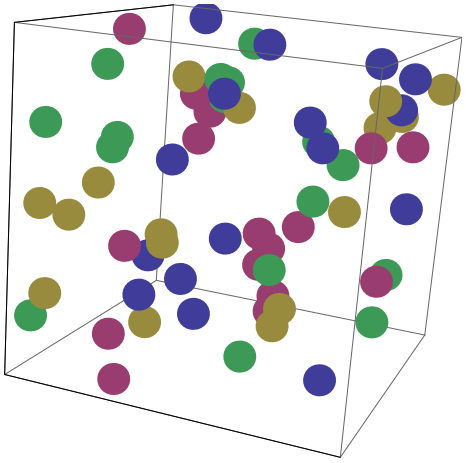

Fig. 1. Power trace representation in 3 dimensions.

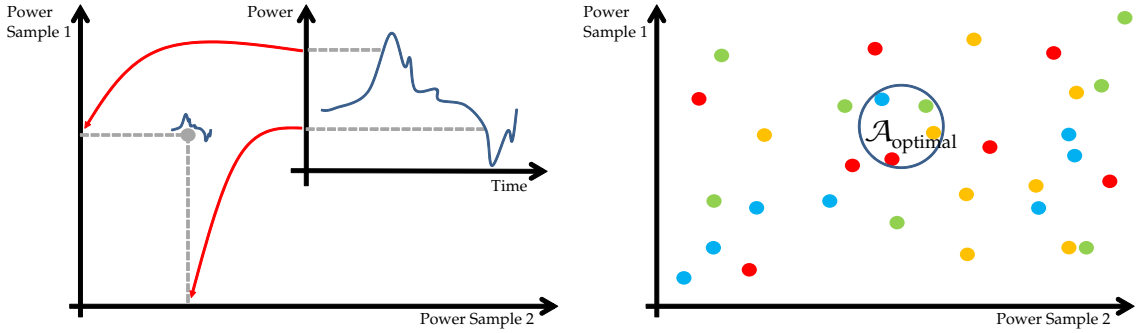

Fig. 2. Left: Mapping curves into points. Right: Problem instance and its optimal solution $\mathcal{A}_{\text{optimal}}$.

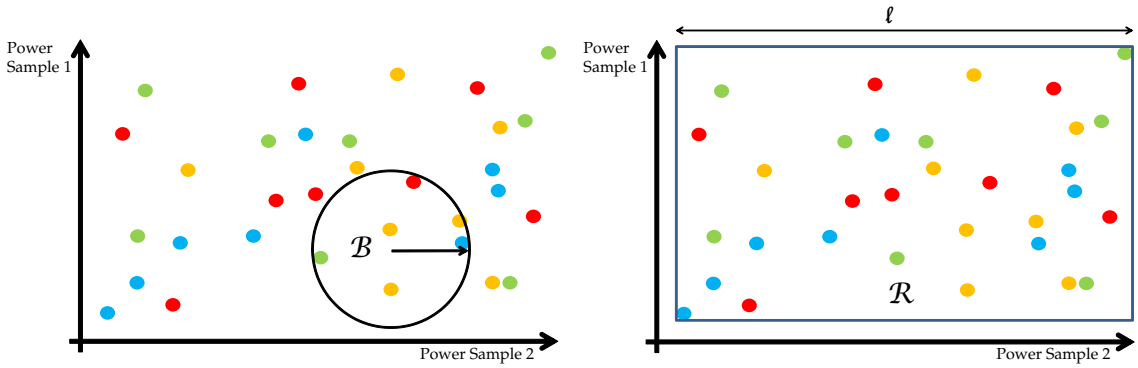

Fig. 3. Left: Step 1, find any color spanning sphere $\mathcal{B}$. Right: Step 2, define the rectangle $\mathcal{R}$.

smallest solution of all, which is indeed the smallest color-spanning sphere in $\mathcal{R}$ *i.e.* the smallest color-spanning sphere $\mathcal{A}_{\text{optimal}}$ of the original problem.

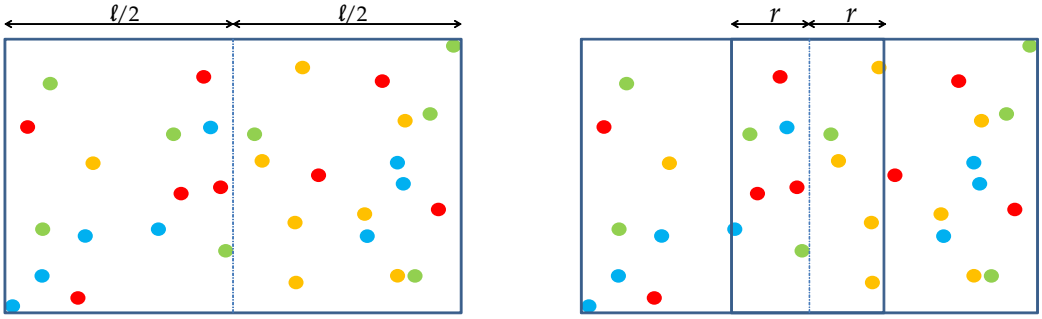

Fig. 4. Left: Step 3, split $\mathcal{R}$ into two overlapping rectangles $\mathcal{R}_{\text{right}}$ and R_{left} of length $\frac{\ell}{2} + r$.

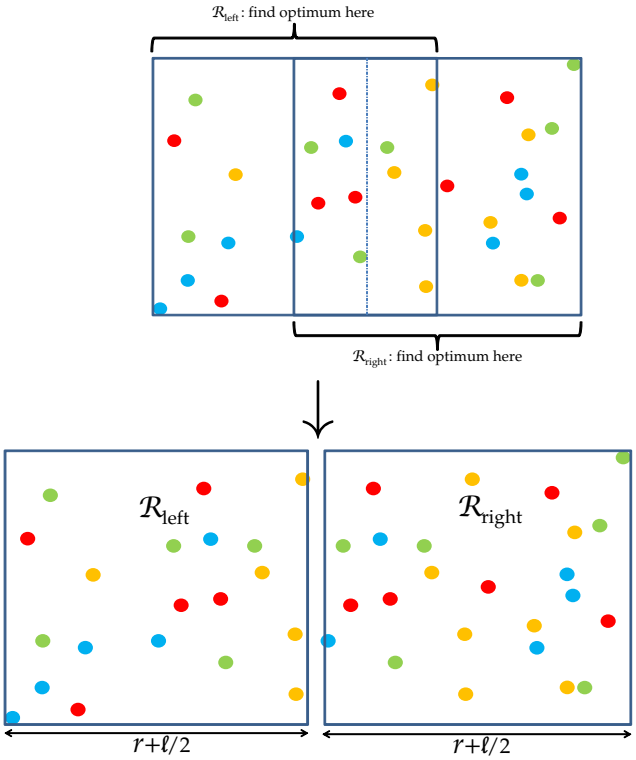

Fig. 5. Recursive problem size reduction.

Note that splitting can take place along several orthogonal axes simultaneously.

While practically very useful, this algorithm fails in a number of pathological cases (*e.g.* when $\mathcal{B}$ is too large to split $\mathcal{R}$). Luckily this is a well-studied problem: [2] describes a simple linear-time algorithm in two dimensions and Welzl [3] shows how to solve the problem in linear

time for all dimensions, considering that the number of dimensions is a fixed problem parameter. Complexity is however exponential in the number of dimensions.

A key choice is the initial sphere $\mathcal{B}$: we want $\mathcal{B}$ to be small enough to significantly reduce the divide and conquer's search space. Yet, we want $\mathcal{B}$ to remain easy to compute.

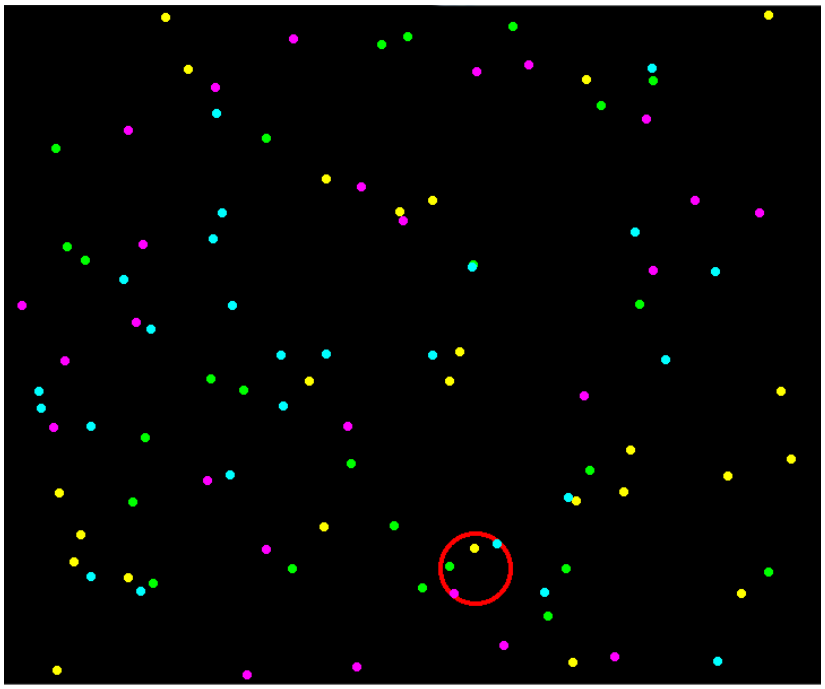

Fig. 6. Program output example in 2 dimensions.

Heuristics: In our implementation we used the following method to construct $\mathcal{B}$: let p_0 be a point (for example the closest point to the center of $\mathcal{R}$) of color 0. After computing $p_1, \ldots, p_k$, we select as p_{k+1} the point of color $k+1$ at minimal distance from the barycenter of the cloud $p_1 \cdots p_k$. The resulting $\mathcal{B}$ is not necessarily optimal, (*cf.* Figure 7) but turns out to be much better than selecting any random color-spanning sphere.

2.3 Implementations

Algorithms were implemented in C++[10] in a straightforward manner. A function

```
bool smallest_ball(points, space, output)
```

splits space and points as explained above (using a sphere found by `find_ball_barycenter`) and calls recursively `smallest_ball` on the smaller spaces, until this process stops to significantly decrease the problem size. We then use Miniball[11], a C++ software for computing

[10] the code is available at http://perso.ens-lyon.fr/quentin.fortier/color_ball.html
[11] http://www.inf.ethz.ch/personal/gaertner/miniball.html

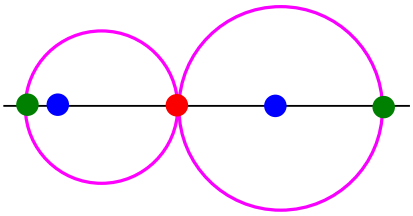

Fig. 7. The optimal sphere (left) is different from the sphere found by the barycenter heuristic (right) if the heuristic considers first the red, then the blue and finally the green points.

smallest enclosing spheres of points in arbitrary dimensions (without requiring spheres to be color spanning) using brute force. The description of Miniball can be found in [4, 3].

Timings were measured on a Dell Inspiron 1520[12]. Code was compiled using Visual C++ 2008 with all optimization flags set for maximal speed.

Total number of points	**2 colors**	**3 colors**	**4 colors**	**5 colors**
10^2	8 ms	11 ms	43 ms	211 ms
10^3	96 ms	221 ms	833 ms	7 s
10^4	946 ms	3 s	11 s	81 s
10^5	10 s	31 s	145 s	953 s
10^6	109 s	327 s		

Table 1. Running time for points randomly chosen in the 3-dimensional unit cube, averaged over 10 runs

Total number of points	**2 colors**	**3 colors**	**4 colors**	**5 colors**
10^2	11 ms	39 ms	309 ms	2 ms
10^3	164 ms	1 s	10 s	147 s
10^4	2 s	16 s	160 s	
10^5	27 s	188 s	37 min	
10^6	287 s	32 min	$>$ 1 hour	$>$ 1 hour

Table 2. Running time for points randomly chosen in the 4-dimensional unit cube, averaged over 10 runs

Experimental running times seem to confirm that the algorithm is linear in the number of points and exponential in the number of colors.

[12] Intel Core 2 Duo T7300 processor, 2.0GHz, 4MB L2 cache, 2Go memory.

3 Why Euclidean Distances?

Let $\{m_{0,t}, \ldots, m_{n-1,t}\}$ be a database of n reference power consumption traces measured over some discrete time interval $t \in [0; T-1]$. Sample $m_{i,t}$ corresponds to the power consumption caused by the manipulation of data element i at instant t. Let μ_t be the average power consumption at time t and σ_t the standard deviation at time t:

$$\mu_t = \frac{1}{n}\sum_{i<n} m_{i,t} \qquad\qquad \sigma_t = \sqrt{\frac{1}{n}\sum_{i<n}(m_{i,t}-\mu_t)^2}.$$

Let a_t be an unidentified power measurement made by an attacker. The attacker's problem consists in finding the $m_{k,t}$ that *best reassembles* a_t. This section justifies why for doing so, an attacker would naturally compute for $i < n$ the quantities:

$$\text{score}(i) = \sum_{t<T} \frac{(a_t - m_{i,t})^2}{\sigma_t^2}, \tag{1}$$

and output the guess k corresponding to the $m_{k,t}$ whose score is the lowest *i.e.*:

$$\text{score}(k) = \min_{i<n}(\text{score}(i)).$$

This formula is justified in the next section for t-wise independent $m_{i,t}$'s.

In general, samples may be correlated, for instance when the same secret bit is manipulated at two different instants. We analyze this general case later and propose an explicit score minimization formula (2) taking into account intra-sample correlations.

3.1 Multivariate Normal Distributions

Equation (1) stems from the assumption that, for any fixed i, successive measurements of $m_{i,t}$ follow an independent normal distribution with mean μ_t and standard deviation σ_t, and hence abide by the probability density function:

$$f_{m_t}(x) = \frac{1}{\sigma_t\sqrt{2\pi}} \exp\Big(-\frac{(x-\mu_t)^2}{2\sigma_t^2}\Big)$$

When the $m_{i,t}$'s are independent, the probability density of all measurements $t < T$ can be expressed, for $\boldsymbol{x} = [x_0 \cdots x_{T-1}]$ as a T-dimensional multivariate distribution:

$$f_{\boldsymbol{m}}(\boldsymbol{x}) = \prod_{t<T} f_{m_t}(x_t) = \frac{1}{(2\pi)^{T/2}\prod\limits_{t<T}\sigma_t} \exp\Big(-\sum_{t<T}\frac{(x_t-\mu_t)^2}{2\sigma_t^2}\Big).$$

Note that in the previous equation μ_t and σ_t are the expected value and standard deviation of $m_{i,t}$ over *all* data elements i. For a measurement $m_{i,t}$ corresponding to a specific data element i, in addition, we also assume that $m_{i,t}$ follows a normal distribution with mean $\tilde{\mu}_t = m_{i,t}$ and standard deviation $\tilde{\sigma}_t$; we also assume that the standard deviation $\tilde{\sigma}_t$ around $m_{i,t}$ is the same

for all data elements. In this case, the measurement m_t corresponding to data element i has the following distribution:

$$f_{\boldsymbol{m}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{T/2} \prod\limits_{t<T} \tilde{\sigma}_t} \exp\Big(- \sum_{t<T} \frac{(x_t - m_{i,t})^2}{2\tilde{\sigma}_t^2} \Big)$$

Additionally, we assume that the standard deviation $\tilde{\sigma}_t$ of m_t around $m_{i,t}$ is proportional to the standard deviation σ_t of m_t when all data values are considered, *i.e.* we assume $\tilde{\sigma}_t = \alpha \cdot \sigma_t$ for all $0 \leq t \leq T-1$ for some $\alpha \in \mathbb{R}$. In this case, the probability density function of the m_t's for data i can be written as:

$$f_i(\boldsymbol{m}) = \frac{1}{(2\pi)^{T/2}\alpha^T \prod\limits_{t<T} \sigma_t} \exp\Big(- \sum_{t<T} \frac{(m_t - m_{i,t})^2}{2\alpha^2\sigma_t^2} \Big) \;\propto\; \exp\Big(- \frac{\text{score}(i)}{2\alpha^2} \Big)$$

where score(i) is given by equation (1). The probability to obtain measurements m_t from data i is thus a decreasing function of score(i). Given measurement $\boldsymbol{m}$, the most probable candidate is therefore the one with the lowest score.

3.2 Multivariate Normal Distribution: Taking Correlation into Account

We denote by Σ the covariance matrix of the measurements, defined as follows:

$$\Sigma = \text{var}(\boldsymbol{m}) = \text{var}\begin{pmatrix} m_1 \\ \vdots \\ m_T \end{pmatrix} = \begin{pmatrix} \text{var}(m_1) & \text{cov}(m_1 m_2) & \cdots & \text{cov}(m_1 m_T) \\ \text{cov}(m_1 m_2) & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \text{cov}(m_1 m_T) & \cdots & \cdots & \text{var}(m_T) \end{pmatrix}$$

where $\text{cov}(X,Y) = \text{E}(XY) - \text{E}(X)\text{E}(Y)$ and $\text{var}(X) = \text{cov}(X,X) = \text{E}(X^2) - \text{E}(X)^2$.

We assume that the measurements follow a T-dimensional multivariate distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ. The probability density function can then be expressed as:

$$f_{\boldsymbol{m}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{T/2}|\Sigma|^{1/2}} \exp\Big(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\text{tr}}\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big).$$

where $|\Sigma|$ is the determinant of Σ and M^{tr} is the transposed of matrix M. The mean $\boldsymbol{\mu}$ is a T-vector and Σ is a $T \times T$-matrix.

Note that in the previous equation $\boldsymbol{\mu}$ and Σ are the expected value and covariance matrix of measurements for *all* data elements i. As previously for measurements corresponding to a specific data element i, we assume that these measurements follow a T-multivariate normal distribution with mean $\tilde{\mu}_t = m_{i,t}$ and covariance matrix $\tilde{\Sigma}$.

If we further assume that matrix $\tilde{\Sigma}$ is identical for all data elements, the measurement $\boldsymbol{m}$ for data i then obeys the multivariate distribution:

$$f_{\boldsymbol{m}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{T/2}|\tilde{\Sigma}|^{1/2}} \exp\Big(- \tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{m_{i,\cdot}})^{\text{tr}}\tilde{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m_{i,\cdot}})\Big).$$

As previously, let us additionally assume that the covariance matrix satisfies $\tilde{\Sigma} = \alpha \cdot \Sigma$ for some $\alpha \in \mathbb{R}$. In this case, the probability density function is expressed by:

$$f_{\boldsymbol{m}}(\boldsymbol{x}) = \frac{1}{(2\pi\alpha)^{T/2}|\Sigma|^{1/2}} \exp\Big(-\tfrac{1}{2\alpha}(\boldsymbol{x} - \boldsymbol{m_{i,.}})^{\mathrm{tr}}\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{m_{i,.}})\Big).$$

This can finally be written as

$$f_{\boldsymbol{m}}(x) = \frac{1}{(2\pi\alpha)^{T/2}|\Sigma|^{1/2}} \exp\Big(-\frac{\mathrm{score}(i)}{2\alpha}\Big)$$

where

$$\mathrm{score}(i) = (\boldsymbol{m} - \boldsymbol{m_{i,.}})^{\mathrm{tr}}\Sigma^{-1}(\boldsymbol{m} - \boldsymbol{m_{i,.}}) \tag{2}$$

It follows that equation (2) is a generalization of equation (1) where correlations are taken into account. In other words, to take correlations into account acquire a_t and compute for every i the score as per equation (2), sort the scores by increasing values and bet on the smallest.

Example To illustrate the procedure, we consider the bivariate case where the covariance matrix between variables X and Y is:

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

where $\mathrm{var}(X) = \sigma_x^2$, $\mathrm{var}(Y) = \sigma_y^2$, $\mathrm{cov}(X, Y) = \rho\sigma_x\sigma_y$ and ρ is the correlation between X and Y. In this case, we find:

$$\Sigma^{-1} = \frac{1}{1-\rho^2}\begin{bmatrix} \frac{1}{\sigma_x^2} & \frac{-\rho}{\sigma_x\sigma_y} \\ \frac{-\rho}{\sigma_x\sigma_y} & \frac{1}{\sigma_y^2} \end{bmatrix}$$

and the probability density function can be written as

$$f(x, y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} - \frac{2\rho xy}{\sigma_x\sigma_y}\right]\right).$$

In this case, equation (2) gets simplified as follows:

$$s_i = \frac{(a_1 - m_{i,1})^2}{\sigma_1^2} + \frac{(a_2 - m_{i,2})^2}{\sigma_2^2} - \frac{2\rho(a_1 - m_{i,1})(a_2 - m_{i,2})}{\sigma_1\sigma_2}$$

where $\sigma_1 = \mathrm{var}(m_1)$, $\sigma_2 = \mathrm{var}(m_2)$ and ρ is the correlation between m_1 and m_2.

4 Experiments

4.1 Measurements

This section describes our experimental results using the Altera EP2C20F484C7N FPGA present on the Cyclone II Starter Development Kit (SDK). Fig.8 shows the circuit used to measure the power consumption of a memory read + register store operation. The circuit consisted of a

RAM, a multiplexer, eight registers, slide switches (DIP) and buttons. Identical data was simultaneously written into eight identical registers to increase power signature.

Power was measured using a 1GHz oscilloscope (TDS 684B) and a Tektronix P6247 differential probe (1GHz bandwidth). The SDK's two GPIO pins (power and ground) were connected via the differential probe. Apart from DC signal rejection no filtering or power trace post processing was done.

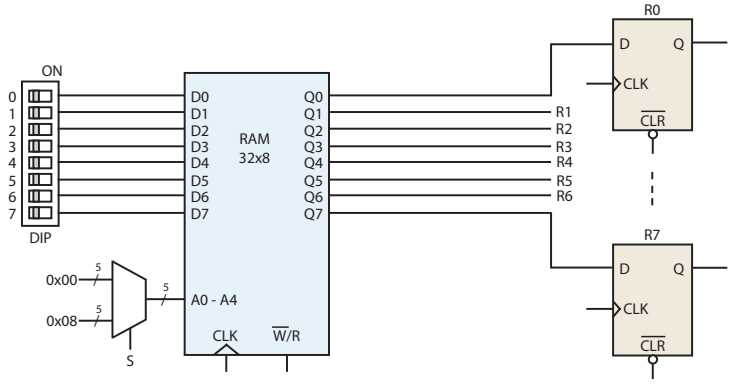

Fig. 8. The experimental circuit used for power consumption measurements

The experimental protocol was defined as follows:

- The DIP's eight slide switches were manually set to 0x00.
- Address 0x00 was latched on address bus A=[A0,...,A4] using the multiplexer's control bit S. This caused the value 0x00 to be written into RAM address 0x00.
- For $d = 0$ to 255:
  - The DIP's eight slide switches were manually set to d.
  - Pressing the board's KEY0 button triggered the following sequence of events 1000 times (averaged to remove noise):
    1. RAM write ($\overline{W}$) was activated and bit S was used to latch address 0x08 on bus A. This caused d to be written to RAM address 0x08 (1 cycle).
    2. RAM read was activated (R) and bit S was used to latch address 0x00 on bus A. This caused 0x00 to be read-out of RAM and clear all data previously present on the bus and in the registers (3 cycles).
    3. The RAM's CLK signal was disabled.
    4. Bit S was used to latch address 0x08 on bus A.
    5. The oscilloscope was triggered.
    6. The RAM's CLK signal was enabled for one cycle only causing d to appear on bus [R0,...,R7]. The RAM's CLK signal was immediately re-disabled to avoid a double-reads and freeze d on bus [R0,...,R7].
    7. At the next clock cycle, d appeared at the Q output pins of the eight registers.
    8. The clock was left running for one more cycle to acquire any signal tails due to capacitive discharges.

- A 2500-sample averaged power measurement $e'(d)$ was recorded.
- Three samples corresponding to instants t_0, t_1, t_2 were extracted from $e'(d)$ to form $e(d)$. $e(d)$ was recorded[13] as a file `trace_d.d` used for camouflage calculations.

The described state-flow could only be interrupted by power-off or by pressing `KEY0`. A finite state-machine (FSM) diagram will appear in the final version of this paper. A characteristic power trace is shown in Figure 9.

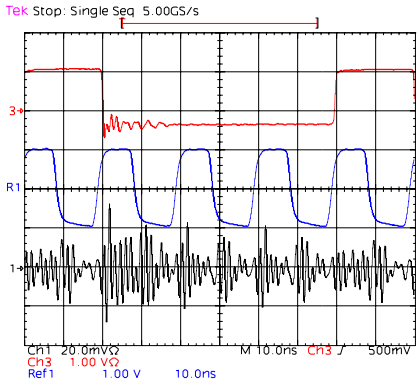

Fig. 9. Power trace of the circuit on Fig.8

The obtained results confirm very wall both our analysis and intuition. However, for various technical reasons, we are not entirely satisfied with this first measurement campaign. We thus plan to refine our setting and provide new experimental results in the final paper.

4.2 Analysis

Figure 12 represent the 256 values ($n = 8$) obtained experimentally as 8 color families (*i.e.* 32 points per family). The experimental data is available upon request.

Our goal is to consider this data as 2^i colors $\times\, 2^{8-i}$ points for $i = 1, \ldots, 7$, select the optimal bus bits on which k should be encoded, compute the $v(k)$ in all cases and check if the results indicate, as we conjecture, that similar Hamming weight words yield the best encoding.

For two colors (*i.e.* a 1-bit k) the two most similar bus values are `0x7F` and `0xF9` for which:

$$\text{distance}(e(\texttt{0x7F}), e(\texttt{0xF9})) = \text{distance}(\{28601, 28795, 28794\}, \{29115, 28789, 28876\}) = 26.94$$

For four colors (*i.e.* a 2-bit k) we get:

binary value of k	**optimal** k **and** $v(k)$	**observed side channel** $e(k, v(k))$
00	0xB4=10110100	$e(\texttt{0xB4}) = \{28704, 28232, 28278\}$
01	0xD9=11011001	$e(\texttt{0xD9}) = \{28652, 28107, 28315\}$
10	0x96=10010110	$e(\texttt{0x96}) = \{28716, 28159, 28293\}$
11	0x6B=01101011	$e(\texttt{0x6B}) = \{28670, 28280, 28380\}$

[13] 3 big-endian values stored in ASCII in decimal format. Each sample is represented by two bytes (oscilloscope's precision).

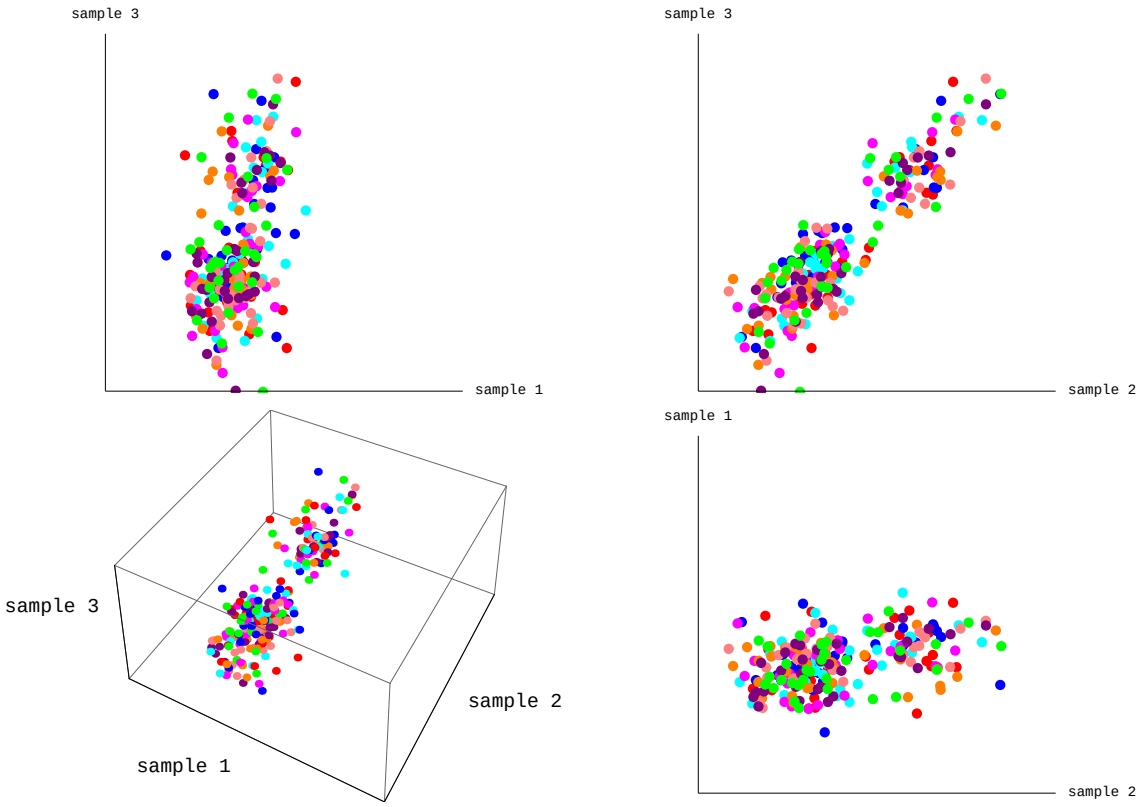

Fig. 10. Experimental results for $n = 8$. 3D and projected representations of the 256 experimental measurements (represented as 8 color families of 32 points).

$e(\texttt{0xB4}), e(\texttt{0xD9}), e(\texttt{0x96}), e(\texttt{0x6B})$ are contained in a sphere of radius $\sqrt{\frac{17239}{2}} \cong 92.84$ centered at $c = \{28661, 28193.5, 28347.5\}$ where:

$$\begin{aligned} \text{distance}(c, e(\texttt{0xB4})) &= 90.34 \quad & \text{distance}(c, e(\texttt{0xD9})) &= 92.84 \\ \text{distance}(c, e(\texttt{0x96})) &= 84.77 \quad & \text{distance}(c, e(\texttt{0x6B})) &= 92.84 \end{aligned}$$

The positions are illustrated in Figure 11 where points were re-scaled to $[0, 1]$ using the affine transform $\text{rescale}(\{x, y, z\}) = \{u(x), u(y), u(z)\}$ where $u(\ell) = (\ell - 28107)/609$:

$$\begin{aligned} \text{rescale}(e(\texttt{0xB4})) &= \{0.98, 0.21, 0.28\} \quad & \text{rescale}(e(\texttt{0xD9})) &= \{0.89, 0.00, 0.34\} \\ \text{rescale}(e(\texttt{0x96})) &= \{1.00, 0.09, 0.31\} \quad & \text{rescale}(e(\texttt{0x6B})) &= \{0.92, 0.28, 0.45\} \end{aligned}$$

5 Conclusion and Further Research

This works raises a number of interesting questions. A first natural generalization is the translation of our analysis to an infinite number of dimensions (in terms of metrics on function spaces and distances between functions).

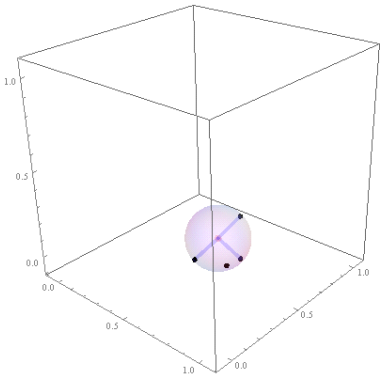

Fig. 11. Display of the re-scaled optimal solution rescale(e(0xB4)), rescale(e(0xD9)), rescale(e(0x96)), rescale(e(0x6B)).

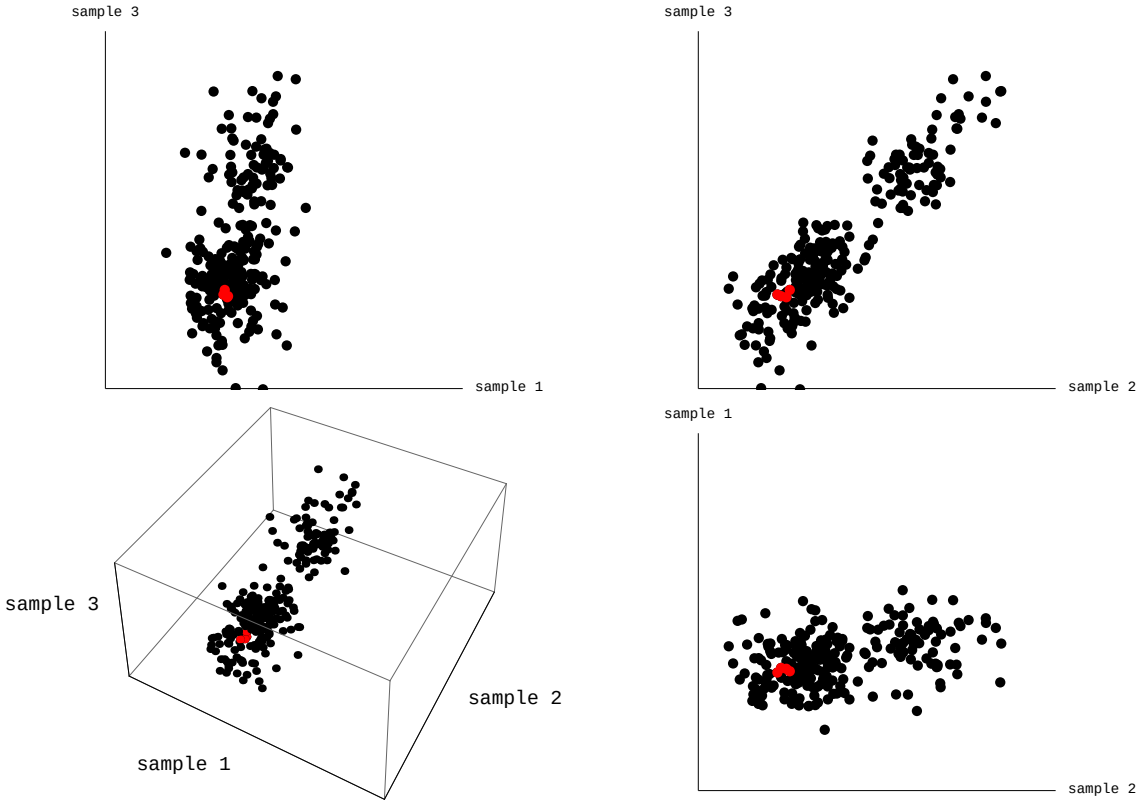

Fig. 12. Experimental results for $n = 8$. Position of the optimal solution e(0xB4), e(0xD9), e(0x96), e(0x6B).

A second line of research consists in introducing more complex information encoding schemes. Here the defender detects the 2^s most similar traces in $\mathcal{E} = \{e(0), \ldots, e(2^n - 1)\}$, *e.g.* using clustering. Let $\mathcal{L}$ be the subset (cluster) of these most similar traces:

$$\mathcal{L} = \{e(\ell(1)), \ldots, e(\ell(2^s - 1))\} \subset \mathcal{E}$$

The communicating parties assign[14] to the transmitted information the encoding:

$$\ell(k) = \text{encode}(k) \qquad\qquad k = \text{decode}(\ell(k))$$

For four colors (*i.e.* a 2-bit k) using our experimental data, we get:

binary value of k	**optimal encode**(k)	**observed side channel** $e($**encode**$(k))$
00	0x96=10010110	$e(\texttt{0x96}) = \{28716, 28159, 28293\}$
01	0xB4=10110100	$e(\texttt{0xB4}) = \{28704, 28232, 28278\}$
10	0xD0=11010000	$e(\texttt{0xD0}) = \{28703, 28238, 28247\}$
11	0xD9=11011001	$e(\texttt{0xD9}) = \{28652, 28107, 28315\}$

$e(\texttt{0x96}), e(\texttt{0xB4}), e(\texttt{0xD0}), e(\texttt{0xD9})$ are contained in a sphere of radius $\sqrt{\frac{12193}{2}} \cong 78.08$ centered at $c = \{28677.5, 28172.5, 28281\}$. This solution (shown in green in Figure 13) shares three points with the previous solution (Figure 12) shown in red.

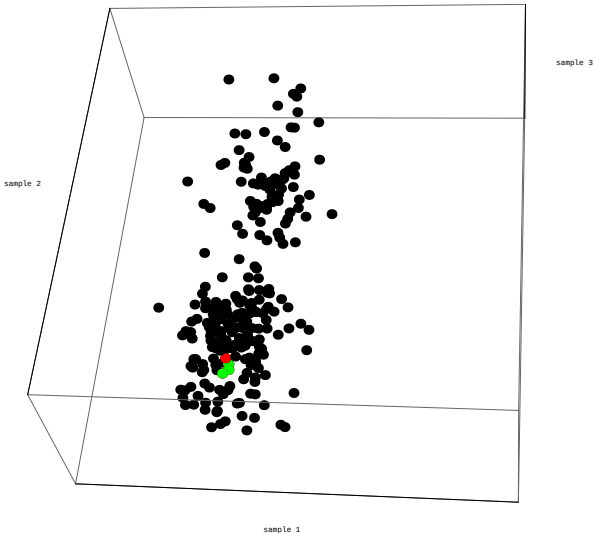

Fig. 13. Experimental results for $n = 8$. Position of the optimal solution $e(\texttt{0xB4}), e(\texttt{0xD9}), e(\texttt{0x96}), e(\texttt{0xD0})$.

Along the same line of ideas, a further refinement consists in buying an easier computation of camouflage values at the cost of extra assumptions on the power consumption model. Assume for instance an isotropic consumption model where emanations are proportional to the Hamming weight of the transmitted data. Here all $\binom{n}{w}$ emissions of weight w cause identical emanations. The largest binomial has weight $w = n/2$, and it is bounded by

$$\frac{2^n}{\sqrt{2n}} < b_n = \binom{n}{\frac{n}{2}} < \frac{2^n}{\sqrt{\pi n/2}}.$$

Assigning $c_n = \lceil \log_2 \sqrt{2n} \rceil$ implies that $2^{n-c_n} \leq 2^n/\sqrt{2n} < b_n$, *i.e.* $2^s < b_n$ for $s = n - c_n$. We can thus choose a distinct configuration of weight $n/2$ to encode each secret key k. It follows

[14] *e.g.* using a lookup table.

that $c_n = (3 + \log_2 n)/2$ bits are sufficient to perfectly hide the emanations from $s = n - c_n$ keys over the n bits of an isotropic bus.

If the noise level is high enough then the implementer may use the fact that

$$\log\left(\binom{n}{\frac{n}{2}}\right) \simeq \log\left(\binom{n}{\frac{n}{2} \pm \gamma}\right) \quad \text{for moderate } \gamma \text{ values}$$

and increase bandwidth at the cost of a carefully controlled security risk.

6 Acknowledgments

Part of this work was supported under grant 12-15-1432-HiCi from King Abdulaziz University.

References

1. E. Brier, C. Clavier, F. Olivier, *Optimal Statistical Power Analysis* Cryptology ePrint Archive, Report 152, http://eprint.iacr.org/, 2003
2. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Springer, 1997
3. E. Welzl, *Smallest Enclosing Disks (Balls and Ellipsoids)*, New Results and New Trends in Computer Science, volume 555 of LNCS, Springer, pp. 359-370, 1991
4. B. Gärtner, *Fast and Robust Smallest Enclosing Balls*, Proc. of the 7th Annual European Symposium on Algorithms (ESA), volume 1643 of LNCS, Springer, pp. 325-338, 1999
5. D. Knuth, *The Art of Computer Programming, vol. 3, Sorting and Searching*, Addison Wesley, 2nd edition, 1998