Template attacks with partial profiles and Dirichlet priors: Application to timing attacks

Eloi De Cherisey, Sylvain Guilley, Olivier Rioul, Darshana Jayasinghe

Template Attacks with Partial Profiles and Dirichlet Priors: Application to Timing Attacks

Eloi de Chérisey[1], Sylvain Guilley[1,2], Olivier Rioul[1], and Darshana Jayasinghe[1]

[1] Télécom ParisTech, LTCI, CNRS, Université Paris-Saclay, 75 013 Paris, France.
[2] Secure-IC, 15 rue Claude Chappe, Bâtiment B, 35 510 Cesson-Sévigné, France.

ABSTRACT

In order to retrieve the secret key in a side-channel attack, the attacker computes distinguisher values using all the available data. A profiling stage is very useful to provide some *a priori* information about the leakage model. However, profiling is essentially empirical and may not be exhaustive. Therefore, during the attack, the attacker may come up on previously unseen data, which can be troublesome. A lazy workaround is to ignore all such novel observations altogether. In this paper, we show that this is not optimal and can be avoided. Our proposed techniques eventually improve the performance of classical information-theoretic distinguishers in terms of success rate.

1. INTRODUCTION

The field of cryptography is currently very sensitive as it deals with data protection and safety. Thus, in order to assess the security of cryptographic devices, it is crucial to know and test their weaknesses. For example, the Advanced Encryption Standard (AES) [1] is renowned as trustworthy from a mathematical point of view—there is currently no realistic way to cryptanalyze the AES-128. However, it is possible to break the 128-bit secret key byte by byte using side-channel analysis (SCA). SCA exploits the physical fact that the secret key leaks some information out of the device boundary through various "side-channels" such as power consumption or *timing*—number of clock cycles to perform a given operation. These leakages, correctly analyzed by SCA, yield the secret key of a device.

A good side-channel attack needs a good leakage model. Timing, for example, can be modeled easily when the implementation is unbalanced: Several successful attacks [2, 3, 4, 5] exploit a timing leakage in the conditional extra-reductions of Montgomery modular multiplications. Some conditional operations can also happen in AES, e.g. in field operations, as for instance discussed in [6, Alg. 1]. Even when the code is balanced—a recommended secure coding practice—some residual unbalances in timing can result from the hardware which executes the code. Indeed, processors implement speed optimization mechanisms such as memory caching and out-of-order execution. As a consequence, it is not possible to predict with certainty how timing leaks information. The attacker is then led to make predictions about the way the device leaks.

In this paper, we consider side-channel attacks that are performed in two phases:

1. a *profiling phase* where the attacker accumulates leakage from a device with a known secret key;
2. an *attacking phase* where the attacker accumulates leakage from the device with an unknown secret key.

This type of attack is known as a *template attack* [7]. It has been shown [7] to be very efficient under three conditions: (a) leakage samples are independent and identically distributed (i.i.d.); (b) the noise is additive white Gaussian; and (c) the secret key leaks byte by byte, which enables a divide-and-conquer approach. For some side-channels, such as power or electromagnetic radiations, condition (b) is met in practice. However, for timing attacks, the noise cannot be Gaussian as timing is discrete. Moreover, the noise source is non-additive in this case, since it arises from complex replacement policies in caches and processor-specific on-the-fly instructions reordering.

The first proposed profiled timing attack is the seminal timing attack of Kocher [8]. The same methodology can be used on AES, as noted by Bernstein in 2005 [9]. Further works used the same method [10, 11, 12]. To our best knowledge, all these works consist in profiling moments, such as the average timing under a given plaintext and key. However, it is known [7] that the best attacks should use maximum likelihood.

In this paper, as illustrated in Tab. 1, we focus on a profiling where the distribution is kept, and is not reduced to its moments. The attacker computes distributions using histogram methods. These distributions are then used to recover the correct secret key.

Table 1: State-of-the-art on profiled timing attacks

Profiling method	Reference articles
Moments	[9, 10, 11, 12]
Distributions	Our paper (Caution about *empty bins*)

The discrete nature of timing leakage leads to an *empty bin* issue which appears when a data value in the attacking phase has never been seen during the profiling phase. Based on profiling only, this data should have a zero probability, which can be devastating for the attack. One known workaround is to use kernel distribution methods [13] to estimate probabilities since the smoothing can be such that no empty bins remain. However, this method has very large computational complexity and its performance highly depends on *ad-hoc* choices of several parameters such as kernel type and bandwidth.

Contributions.

In this paper we show that even when all requirements (a), (b), (c) above are not present, timing attacks with incomplete profiling can be achieved successfully by adapting the maximum likelihood distinguisher and keeping the histogram method for probabilities estimation. We build six different distinguishers which are all good answers to the empty bin issue. For some of them, new histograms are built, such that the empty bin issue totally disappears. Furthermore, we compare these distinguishers and show which one of them is the best in every specific context. Finally, we provide some theoretical results proving how optimal some of the distinguishers can be.

Organization.
The paper is organized as follows. Section 2 provides mathematical tools to understand distinguishers and notations. Section 3 introduces new distinguishers that are suitable in the context of empty bins. Section 4 provides simulations for these distinguishers and Section 5 investigates real attacks on an ARM processor. Section 6 concludes.

2. MATHEMATICAL DERIVATIONS

2.1 Notations and Assumptions

We consider a side-channel attack with a profiling stage and use the following notations:

- during the profiling phase, a vector $\widehat{\mathbf{t}}$ of $\widehat{q}$ text bytes is sent and the profiler garners a vector of $\widehat{\mathbf{x}}$ measurements;
- during the attacking phase, a vector $\widetilde{\mathbf{t}}$ of $\widetilde{q}$ text bytes is sent and the attacker gathers a vector $\widetilde{\mathbf{x}}$ of leakage measurements—also customarily known as *traces*;
- we use simplified notations $\mathbf{t}$, q and $\mathbf{x}$ when discussing either profiling data or attacking data;
- the probability of a vector $\mathbf{x}$ with i.i.d. components x_i is denoted by $\mathbb{P}(\mathbf{x}) = \prod_i \mathbb{P}(x_i)$;
- we define the following sets:
  1. $\widehat{\mathscr{X}}$, $\widehat{\mathscr{T}}$, $\widetilde{\mathscr{X}}$ and $\widetilde{\mathscr{T}}$ are the sets of possible values of components $\widehat{x}$, $\widehat{t}$, $\widetilde{x}$ and $\widetilde{t}$, respectively;
  2. $\mathscr{X} = \widehat{\mathscr{X}} \cup \widetilde{\mathscr{X}}$ and $\mathscr{T} = \widehat{\mathscr{T}} \cup \widetilde{\mathscr{T}}$;
  3. $\mathscr{K}$ is the set of all possible values for the key k.
- k and t are made of n bits (in particular, they are "bytes" when $n = 8$).

Here all sample components of one vector are i.i.d. and belong to some discrete set. Typically, $\mathscr{X}$ is a finite subset of $\mathbb{N}$ and $\mathscr{T}$ is equal to $\{0,1\}^n$.

In the profiling stage, the secret key $\widehat{k}^*$ is known and variable. In the attacking phase, the secret key $\widetilde{k}^*$ is unknown but fixed. Further, we assume that x_i depends only on t_i and k^* for all $i = 1, 2, \ldots, q$, in the form:

$$x_i = \psi(t_i \oplus k^*) \qquad (i = 1, 2, \ldots, q) \tag{1}$$

where $\oplus$ is the XOR (exclusive or) operator and ψ is an unknown function which may contain noise, masking and other hidden parameters[1].

Furthermore, in this paper, we use of the notation $n_{x,t}$ to denote the number of occurrences of (x,t). Thus we can write

$$\widehat{n}_{x,t} = \textstyle\sum_{i=1}^{\widehat{q}} \mathbb{1}_{\widehat{x}_i = x, \widehat{t}_i = t} \qquad \widehat{n}_x = \sum_{i=1}^{\widehat{q}} \mathbb{1}_{\widehat{x}_i = x},$$
$$\widetilde{n}_{x,t} = \textstyle\sum_{i=1}^{\widetilde{q}} \mathbb{1}_{\widetilde{x}_i = x, \widetilde{t}_i = t} \qquad \widetilde{n}_x = \sum_{i=1}^{\widetilde{q}} \mathbb{1}_{\widetilde{x}_i = x}.$$

where $\mathbb{1}_A = 1$ if A is true, $= 0$ otherwise.

Definition 2.1 (Probabilities). *We define three*[2] *different types of probabilities $\mathbb{P}$, $\widehat{\mathbb{P}}$ and $\widetilde{\mathbb{P}}$. $\mathbb{P}$ is the actual (real) underlying probability distribution, but it is generally not available and has to be estimated by either $\widehat{\mathbb{P}}$ or $\widetilde{\mathbb{P}}$.*

- *$\widehat{\mathbb{P}}$ is computed using the profiling data:*

$$\widehat{\mathbb{P}}(x,t) = \frac{1}{\widehat{q}} \sum_{i=1}^{\widehat{q}} \mathbb{1}_{\widehat{x}_i = x, \widehat{t}_i = t} = \frac{\widehat{n}_{x,t}}{\widehat{q}}, \tag{2}$$

$$\widehat{\mathbb{P}}(x) = \frac{1}{\widehat{q}} \sum_{i=1}^{\widehat{q}} \mathbb{1}_{\widehat{x}_i = x} = \frac{\widehat{n}_x}{\widehat{q}}. \tag{3}$$

- *$\widetilde{\mathbb{P}}$ is computed using the attacking data:*

$$\widetilde{\mathbb{P}}(x,t) = \frac{1}{\widetilde{q}} \sum_{i=1}^{\widetilde{q}} \mathbb{1}_{\widetilde{x}_i = x, \widetilde{t}_i = t} = \frac{\widetilde{n}_{x,t}}{\widetilde{q}}, \tag{4}$$

$$\widetilde{\mathbb{P}}(x) = \frac{1}{\widetilde{q}} \sum_{i=1}^{\widetilde{q}} \mathbb{1}_{\widetilde{x}_i = x} = \frac{\widetilde{n}_x}{\widetilde{q}}. \tag{5}$$

In practice, as the secret key leaks through the function via a XOR (Equation (1)), we shall often consider $\mathbb{P}(x, t \oplus k)$.

For a fair comparison between distinguishers, Standaert et al. [14] have put forward the *success rate* as a measure of efficiency of a given distinguisher.

Definition 2.2 (Success Rate). *The success rate* SR *is probability, averaged over all possible keys, of obtaining the correct key.*

$$\mathsf{SR} = \frac{1}{2^n} \sum_{k^*=0}^{2^n-1} \mathbb{P}_{k^*}(\widetilde{k} = k^*), \tag{6}$$

where $\widetilde{k}$ is the key guess obtained by the distinguisher during the attack.

It has been proven [15, Theorem 1, equation (3)] that for equiprobable keys the optimal distinguisher maximizes likelihood:

$$\mathscr{D}_{\mathsf{Optimal}}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{t}}) = \arg\max_{k \in \mathscr{K}} \mathbb{P}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k). \tag{7}$$

In real life, however, the attacker does not know the leakage model perfectly and thus $\mathbb{P}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k)$ is not available. In order to get an estimation of $\mathbb{P}$, we use the profiling data to build $\widehat{\mathbb{P}}$ defined in Equation (2). This is the classical *template attack*. The distinguisher becomes

$$\mathscr{D}_{\mathsf{Template}}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{t}}) = \arg\max_{k \in \mathscr{K}} \widehat{\mathbb{P}}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k). \tag{8}$$

This distinguisher is no longer optimal as it does not use the real distribution $\mathbb{P}$. However, if profiling tends to exhaustivity, $\widehat{\mathbb{P}}$ and $\mathbb{P}$ will be very close since by the law of large numbers,

$$\forall x, t \quad \widehat{\mathbb{P}}(x,t) \xrightarrow[\widehat{q} \to \infty]{} \mathbb{P}(x,t).$$

In practice, it is convenient to use the logarithm $\arg\max_{k \in \mathscr{K}} \log \widehat{\mathbb{P}}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k)$. In fact, since the samples are i.i.d., we have

$$\mathbb{P}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k) = \prod_{i=1}^{\widetilde{q}} \mathbb{P}(\widetilde{x}_i | \widetilde{t}_i \oplus k) \quad \text{and} \quad \widehat{\mathbb{P}}(\widetilde{\mathbf{x}} | \widetilde{\mathbf{t}} \oplus k) = \prod_{i=1}^{\widetilde{q}} \widehat{\mathbb{P}}(\widetilde{x}_i | \widetilde{t}_i \oplus k).$$

[1] The AES meets the secret and the text byte through a xor (`SubBytes`) executed in a fixed number of clock cycles. However, the rest of the AES consists in table lookups and other miscellaneous operations which are difficult to model and need different amounts of time to execute, hence the use of unknown function ψ.

[2] For the sake of evading the empty bin issue, we will also introduce yet another notation "$\mathbb{P}_\alpha$" in section 3.1 (Equation (15)).

Therefore, the attacker computes

$$\mathscr{D}_{\mathsf{Template}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \sum_{i=1}^{\widetilde{q}} \log \widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k) \tag{9}$$

where the logarithm was used to transform products into sums for a more reliable computation. However, we would like to avoid empty bins for which $\widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k) = 0$, since otherwise, Equation (9) would not be well defined.

2.2 About Empty Bins

The empty bin issue appears when there exists $i \in \{1,\ldots,\widetilde{q}\}$ and $k \in \mathscr{K}$ such that $\widetilde{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k) > 0$ and $\widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k) = 0$. This may even happen for the *correct* key hypothesis, leading to a wrong key guess during the attack.

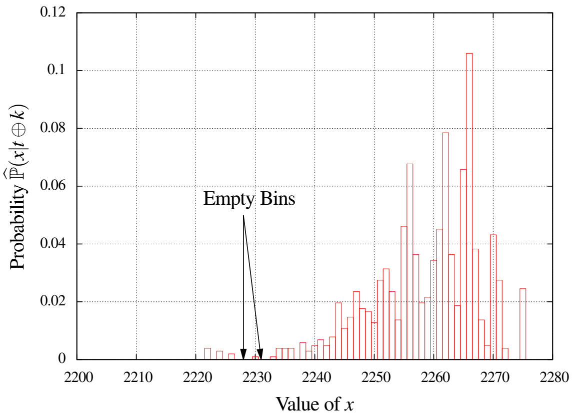

Figure 1: Empirical probability $\widehat{\mathbb{P}}(x|t\oplus k)$ for $t = 0$ and $k = 67$ and $\widehat{q} = 2560000$

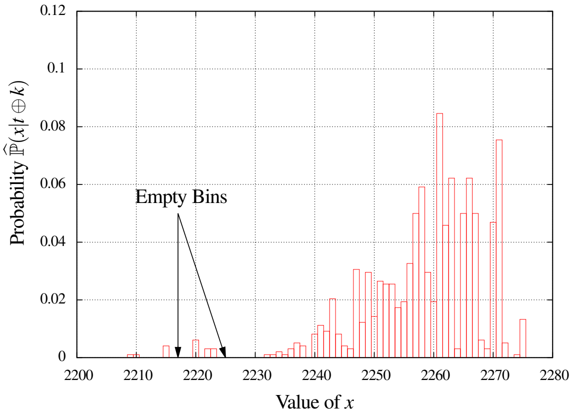

Figure 2: Empirical probability $\widehat{\mathbb{P}}(x|t\oplus k)$ for $t = 0$ and $k = 149$ and $\widehat{q} = 2560000$

Figures 1 and 2 show how empty bins can look like after a profiling phase[3]. We notice that some parts of the histograms are left blank, some of them indicated by arrows. These timing values x are the "empty bins". Notice that no additional "binning" is needed as in the case of continuous distributions. The figures also show that the noise is not Gaussian as can be observed from the shape of the distribution.

The shortcoming of empty bins can be seen when evaluating the likelihood. The attacker encounters a zero probability, which makes the product vanish for the probability of a given key guess, even if many traces are used. This multiplication by zero is not inherent to the attack; it is rather a profiling artifact. In fact, with more profiling traces, the empty bin would likely be populated. Thus the empty bin issue is a mere side-effect of insufficient profiling, which results in an attack failure if it is encountered in the computation of the likelihood of the correct key.

3. DISTINGUISHERS WHICH TOLERATE EMPTY BINS

3.1 Building Distributions or Models

Before presenting the novel distinguishers in Subsection 3.2, we need to define yet another other type of distribution known as a Dirichlet *a posteriori* in a Bayesian approach.

The Dirichlet A Posteriori

In order to avoid zero probabilities, we use a method based on Dirichlet Prior calculations [16, Section 1]. This method leads to a new distribution denoted by $\overline{\mathbb{P}}_\alpha$, where $\alpha > 0$ is a user-defined parameter whose value (typically $= 1$) will be discussed next.

Let $\mathscr{X}$ be the set of possible values for x and $\mathscr{T}$ be the set of possible values for t. For any x,, we set $p_{x,t} = \mathbb{P}(x,t)$ their joint probability and $\mathbf{p} = (p_{x,t})_{x,t}$. Prior to obtaining any trace, $p_{x,t}$ is completely unknown and we consider a Bayesian approach to estimate $p_{x,t}$.

1. We consider the following *a priori*: without further information, we suppose that for all x,t,

   $$\overline{\mathbb{P}}_\alpha(x,t) = \frac{\alpha_{x,t}}{\sum_{x',t'} \alpha_{x',t'}},$$

   where $\alpha_{x,t} > 0$ is an *a priori* parameter. To simplify we may choose $\alpha_{x,t} = \alpha$ constant for all x,t. Let us suppose that $\mathbf{p}$ follows a Dirichlet (prior) distribution, whose probability density function is

   $$f(\mathbf{p}) = \frac{\Gamma(\sum_{x,t} \alpha_{x,t})}{\prod_{x,t} \Gamma(\alpha_{x,t})} \prod_{x,t} p_{x,t}^{\alpha_{x,t}-1}, \tag{10}$$

   where Γ is the Gamma function defined for $x > 0$ as

   $$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t}\, \mathrm{d}t. \tag{11}$$

   The Dirichlet distribution can also be written as

   $$f(\mathbf{p}) = \mathscr{N}_\alpha \prod_{x,t} p_{x,t}^{\alpha_{x,t}-1}, \tag{12}$$

   where $\mathscr{N}_\alpha = \frac{\Gamma(\sum_{x,t} \alpha_{x,t})}{\prod_{x,t} \Gamma(\alpha_{x,t})}$ is a normalization factor. Notice that the prior distribution is *uniform* when $\alpha_{x,t} = \alpha = 1$ for all x,t.

2. Then suppose we know $\widehat{\mathbf{x}}$, $\widetilde{\mathbf{x}}$, $\widehat{\mathbf{t}}$ and $\widetilde{\mathbf{t}}$. We can now compute the *a posteriori* probability

   $$\mathbb{P}(x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \int f(\mathbf{p},x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}})\, \mathrm{d}p.$$

   By Bayes' rule,

   $$f(\mathbf{p},x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \mathbb{P}(x,t|\mathbf{p},\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) f(\mathbf{p}|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}).$$

[3] Figures obtained with the STM Discovery Board presented in Section 5. The unit of x is the "clock cycle".

As components x_i and t_i are i.i.d., we can write

$$\begin{aligned} f(\mathbf{p},x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) &= \mathbb{P}(x,t|\mathbf{p})\cdot f(\mathbf{p}|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}},t) \\ &= p_{x,t}\cdot f(\mathbf{p}|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) \end{aligned}$$

Again by Bayes' rule,

$$\begin{aligned} f(\mathbf{p}|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) &= \frac{\mathbb{P}(\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widetilde{\mathbf{t}},\widehat{\mathbf{t}}|\mathbf{p})f(\mathbf{p})}{\mathbb{P}(\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widetilde{\mathbf{t}},\widehat{\mathbf{t}})} \\ &= \frac{\prod_{x',t'\in\mathscr{X}\times\mathscr{T}} p_{x',t'}^{\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}(k)}}{\mathbb{P}(\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widetilde{\mathbf{t}},\widehat{\mathbf{t}})} f(\mathbf{p}) \\ &= \frac{\mathscr{N}_\alpha}{\mathbb{P}(\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widetilde{\mathbf{t}},\widehat{\mathbf{t}})} \prod_{x',t'\in\mathscr{X}\times\mathscr{T}} p_{x',t'}^{\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\alpha_{x',t'}-1}. \end{aligned}$$

We recognize another Dirichlet distribution with parameters $\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\alpha_{x',t'}$. Let $\mathscr{N}_{\alpha'} = \frac{\Gamma(\sum_{x',t'}\alpha_{x',t'}+\widetilde{n}_{x',t'}+\alpha_{x',t'})}{\prod_{x,t}\Gamma(\alpha_{x,t}+\widetilde{n}_{x',t'}+\alpha_{x',t'})}$ be the new normalization constant for this distribution. We finally obtain

$$f(\mathbf{p},x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = p_{x,t}\cdot\mathscr{N}_{\alpha'} \prod_{x',t'\in\mathscr{X}\times\mathscr{T}} p_{x',t'}^{\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\alpha_{x',t'}-1}. \tag{13}$$

Therefore,

$$\mathbb{P}(x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \int p_{x,t}\cdot\mathscr{N}_{\alpha'} \prod_{x',t'\in\mathscr{X}\times\mathscr{T}} p_{x',t'}^{\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\alpha_{x',t'}-1}\,\mathrm{d}p.$$

which is known as the Dirichlet *a posteriori*.

3. The integral can be easily expressed in terms of the Gamma function:

$$\mathbb{P}(x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \frac{\Gamma(\sum_{x',t'}\alpha_{x,t}+\widehat{n}_{x',t'}+\widetilde{n}_{x',t'})}{\prod_{x',t'}\Gamma(\alpha_{x,t}+\widehat{n}_{x',t'}+\widetilde{n}_{x',t'})} \times \frac{\prod_{x',t'}\Gamma(\alpha_{x,t}+\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\delta_{x,t})}{\Gamma(\sum_{x',t'}\alpha_{x,t}+\widehat{n}_{x',t'}+\widetilde{n}_{x',t'}+\delta_{x,t})}$$

which simplifies to

$$\mathbb{P}(x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \frac{\widehat{n}_{x,t}+\widetilde{n}_{x,t}+\alpha_{x,t}}{\widehat{q}+\widetilde{q}+\sum_{x',t'}\alpha_{x',t'}}.$$

This new distribution will now be noted:

$$\bar{\mathbb{P}}_\alpha(x,t) = \mathbb{P}(x,t|\widehat{\mathbf{x}},\widetilde{\mathbf{x}},\widehat{\mathbf{t}},\widetilde{\mathbf{t}}) = \frac{\widehat{n}_{x,t}+\widetilde{n}_{x,t}+\alpha_{x,t}}{\widehat{q}+\widetilde{q}+\sum_{x',t'}\alpha_{x',t'}}. \tag{14}$$

It is important to notice that for all $(x,t)\in\mathscr{X}\times\mathscr{T}$, one has $\bar{\mathbb{P}}_\alpha(x,t)>0$. In other words, $\bar{\mathbb{P}}_\alpha$ has **no empty bin issue**.

4. With $\bar{\mathbb{P}}_\alpha(x,t)$ we can calculate

$$\begin{aligned} \bar{\mathbb{P}}_\alpha(t) = \sum_x \bar{\mathbb{P}}_\alpha(x,t) &= \sum_x \frac{\widehat{n}_{x,t}+\widetilde{n}_{x,t}+\alpha_{x,t}}{\widehat{q}+\widetilde{q}+\sum_{x',t'}\alpha_{x',t'}} \\ &= \frac{\widehat{n}_t+\widetilde{n}_t+\sum_t\alpha_{x,t}}{\widehat{q}+\widetilde{q}+\sum_{x',t'}\alpha_{x',t'}} = \frac{\widehat{n}_t+\widetilde{n}_t+\alpha_t}{\widehat{q}+\widetilde{q}+\sum_{x'}\alpha_{x'}}. \end{aligned}$$

where $\alpha_t = \sum_x \alpha_{x,t}$. The resulting conditional probability[4] is

$$\bar{\mathbb{P}}_\alpha(x|t) = \frac{\bar{\mathbb{P}}_\alpha(x,t)}{\bar{\mathbb{P}}_\alpha(t)} = \frac{\widehat{n}_{x,t}+\widetilde{n}_{x,t}+\alpha_{x,t}}{\widehat{n}_t+\widetilde{n}_t+\alpha_t}. \tag{15}$$

The Learned MIA Model

When $\widehat{q}$ is small, the model cannot be profiled accurately and $\widehat{\mathbb{P}}$ is a bad approximation of $\mathbb{P}$. However, these profiled values $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{t}}$ can still be useful yet they require a more robust distinguisher.

Distinguishers that compute models using profiling have already been proposed. For example, [17] computes a correlation on moments. However, correlations analysis may be sensitive to model errors [18]. Mutual Information Analysis (MIA) yields a distinguisher that can be robust when models are not perfectly known [18, Section 4] but it requires at least a vague estimation of the leakage model.

Since our function ψ is unknown, we can create a first-order model $\widehat{\psi}$ with the profiled data as

$$\widehat{\psi}(t\oplus\widehat{k}^*) = \mathsf{Step}\Big(\frac{1}{n_t}\sum_{i \text{ s.t. } \widehat{t}_i=t}\widehat{x}_i\Big) \qquad (\forall t\in\mathscr{T}). \tag{16}$$

The Step function is a function that ensures the non-injectivity of the model. The simplest way to define Step would be the following:

$$\mathsf{Step}(x) = \frac{\lfloor d\cdot x\rfloor}{d} \qquad (x\in\mathbb{R})$$

where $d>0$—the greater d, the smaller the step size. This parameter d has to be small enough in order to make the model non-injective [19]. In our case, we choose, for all our experiments, $d=1$. With such a model, it is possible to compute a MIA which successfully distinguishes the correct key.

3.2 Robust distinguishers

In this subsection, we present several distinguishers that tackle null probabilities. Some of these solutions seem quite obvious while others are deduced from the notions presented in the preceding Subsection 3.1.

Hard Drop Distinguisher

The first naive method consists in removing all the traces which, for any key guess, have a zero probability.

DEFINITION 3.1 (HARD DROP DISTINGUISHER). *The hard drop distinguisher is defined as followed:*

$$\mathscr{D}_{\mathsf{Hard}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \sum_{i\in\mathscr{I}} \log\widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i\oplus k), \tag{17}$$

where $\mathscr{I}$ is defined as

$$\mathscr{I} = \Big\{ i\in\{1,\dots,\widetilde{q}\} \mid \forall k\in\mathscr{K},\, \widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i\oplus k)>0 \Big\}. \tag{18}$$

Recall that $\widehat{\mathbb{P}}$, defined in Equation (2), is an empirical histogram estimated on profiled data $\widehat{\mathbf{x}}$ (along with corresponding texts $\widehat{\mathbf{t}}$).

The Hard Drop Distinguisher, as the name indicates, drops some data. In very noisy cases, it may even drop most of the data.

Soft Drop Distinguisher

The second possibility is to drop values only for some keys. However, it has to be done carefully because dropping a value in a product implicitly implies a probability value of one. For this reason, instead of removing the trace, we replace the zero probability by a constant which is smaller than the smallest probability.

DEFINITION 3.2 (SOFT DROP DISTINGUISHER). *We define the Soft Drop Distinguisher as*

$$\mathscr{D}_{\mathsf{Soft}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \sum_{i \text{ s.t. } \widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i\oplus k)>0} \log\widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i\oplus k) \;+\; \sum_{i \text{ s.t. } Ph(\widetilde{x}_i|\widetilde{t}_i,k)=0} \log\gamma, \tag{19}$$

[4] We should normally have used the notation $\widetilde{\widehat{\bar{\mathbb{P}}}}_\alpha$ instead of $\bar{\mathbb{P}}_\alpha$, but found this too heavy and confusing, hence the use of $\bar{\mathbb{P}}_\alpha$.

where $\gamma \in \mathbb{R}_+^*$ is a constant such that $\forall i,k \in \{1,\ldots,\widetilde{q}\} \times \mathscr{K}, \quad \gamma \leq \widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k)$. This means that we penalize data with zero probability. The smaller γ, the harder the penalty.

The Dirichlet Prior Distinguisher

The Dirichlet Prior Distinguisher uses the Dirichlet *a posteriori* distributions presented in Subsection 3.1.

DEFINITION 3.3 (THE DIRICHLET DISTINGUISHER). *We define the Dirichlet Distinguisher as:*

$$\mathscr{D}_{\mathsf{Dirichlet}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \bar{\mathbb{P}}_\alpha(\widetilde{\mathbf{x}}|\widetilde{\mathbf{t}} \oplus k). \tag{20}$$

REMARK 1. *As can be seen in the construction of the Dirichlet a posteriori, the Dirichlet distinguisher is α-dependent. It is important to evaluate the influence of α over the success rate. In practice, $\alpha = 1$ seems a natural choice since the corresponding prior is uniform, which minimizes the impact of the* a priori. *In contrast, another value of α like $1/2$ can be interpreted as an* a priori *bin count. We may also consider scenarios where $\alpha \approx 0$ to have the least possible impact to the modified values of the histogram.*

Offline-Online Profiling

The Dirichlet Prior Distinguisher is parameterized by α. As we discussed in Remark 1, we can choose any α so long as it is strictly positive (the Dirichlet distribution would not be defined if $\alpha = 0$). However, it is interesting to study its asymptotical behavior as α vanishes:

$$\lim_{\alpha\to 0} \bar{\mathbb{P}}_\alpha(x|t) = \frac{\widehat{n}_{x,t} + \widetilde{n}_{x,t}}{\widehat{n}_t + \widetilde{n}_t}.$$

This distribution can be denoted as $\bar{\mathbb{P}}_0(x|t)$ and resembles a profiling stage that would start offline and continue online.

DEFINITION 3.4 (OFFLINE-ONLINE PROFILING). *The Offline-Online Profiled (OOP) distinguisher is defined as:*

$$\mathscr{D}_{\mathsf{OOP}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \bar{\mathbb{P}}_0(\widetilde{\mathbf{x}}|\widetilde{\mathbf{t}} \oplus k) \tag{21}$$

The OOP distinguisher seems easier than the Dirichlet prior distinguisher since α is no longer in use. Of course, it also solves the empty bin issue since for all $(x,t) \in \mathscr{X} \times \mathscr{T}$, one has $\bar{\mathbb{P}}_0(x,t) > 0$.

Learned MIA Distinguisher

The Learned MIA Distinguisher is made with the profiled model function presented in Subsection 3.1.

DEFINITION 3.5 (THE LEARNED MIA DISTINGUISHER). *The Learned MIA Distinguisher is defined as:*

$$\mathscr{D}_{\mathsf{MIA\_Learned}} = \arg \max_{k\in\mathscr{K}} \widetilde{\mathrm{I}}\left(\widetilde{\mathbf{x}}; \widehat{\psi}(\widetilde{\mathbf{t}} \oplus k)\right), \tag{22}$$

where $\widetilde{\mathrm{I}}$ is the empirical mutual information [20].

Empty Bin Distinguisher

The empty bin Distinguisher is yet another intuitive solution based on the idea that instead of avoiding null probabilities, we may take only these into account. It is the key guess with the least number of null probabilities that "should" be the correct key.

DEFINITION 3.6. *The Empty Bin Distinguisher is defined as:*

$$\mathscr{D}_{\mathsf{Empty\_Bin}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\min_{k\in\mathscr{K}} \sum_{i=1}^{\widetilde{q}} \mathbb{1}_{\widehat{\mathbb{P}}(\widetilde{x}_i|\widetilde{t}_i \oplus k)=0}. \tag{23}$$

The Empty Bin Distinguisher assumed that missing data contain more information than actual (measured) data. More precisely, a drop should normally not happen unless the guessed key is wrong, hence the key guess with the least drops should be the correct key. Obviously, this distinguisher is not effective anymore if no drop occurs for at least two key guesses.

Further Remarks.

All these distinguishers use a profiling phase. Before comparing them, we would like to make *a priori* discussion about their respective efficiencies. As the Hard Drop Distinguisher does not take into account some data, we may suppose that it will be the one with the least success rate for a given number of traces. The OOP Distinguisher takes into account two types of data: profiling and attacking data. Therefore, it should be more efficient than other distinguishers. Lastly, we build the Learned MIA Distinguisher in order to prevent model errors, such as inaccurate profiling. In that case, we suppose that Learned MIA should work better with few data during the profiling stage.

4. SIMULATED RESULTS

In this section, we present results obtained on a simulated model. With these results, we can give a comparison of the proposed distinguishers.

4.1 Presentation of the Simulated Model

The simulated model is built as follows:

$$\begin{aligned} x_i &= \mathrm{H_w}(\mathrm{SubBytes}(t_i \oplus k^*)) + u_i \\ &= \phi(t_i \oplus k^*) + u_i = y_i(k^*) + u_i, \end{aligned} \tag{24}$$

where u_i is a discrete uniformly distributed noise $u_i \sim \mathscr{U}(-\sigma,\sigma)$, SubBytes is the AES substitution box function, and $\mathrm{H_w}$ is the Hamming weight of a byte.

This very simple leakage is used to compare distinguishers in the case the attacker has no information about the model.

REMARK 2 (OPTIMAL DISTINGUISHER). *The optimal distinguisher (7) can be easily calculated if the model is perfectly known, as*

$$\mathscr{D}_{\mathsf{Optimal}}(\widetilde{\mathbf{x}},\widetilde{\mathbf{t}}) = \arg\max_{k\in\mathscr{K}} \prod_{i=1}^{\widetilde{q}} \delta_\sigma(\widetilde{x}_i - \mathrm{H_w}(\mathrm{SubBytes}(\widetilde{t}_i \oplus k))), \tag{25}$$

where δ_σ is defined such that $\delta_\sigma(x) = 1$ if $|x| \leq \sigma$ and 0 otherwise. In Figures 3, 4 and 5, we include the optimal distinguisher for reference, to show how far the other curves are from the fundamental limit of performance.

By construction, the leakage simulation (24) generates some traces with zero probability but notice that there is no i such that $\mathbb{P}(x_i|t_i,k) = 0$ for the correct key guess. This academic example is useful to compare the distinguishers defined in Section 3.

4.2 Attack Results

We computed the success rates (6) of the various attacks for for $\sigma = 24$, $n = 4$ bits, and $\widehat{q}$ ranging from small to high values.

The only difference between Figures 3, 4 and 5, is that we have increased the number of data during the profiling stage. When profiling is bad (Figure 3), the best distinguisher is the Offline-Online profiling distinguisher, while the Learned MIA Distinguisher is not as good as was expected. When $\widehat{q} = 1\,600$ (Figure 4), all distinguishers improve. Finally, when profiling is good ($\widehat{q} = 4\,000$, Figure 5) the best distinguisher is now the Empty Bin distinguisher, followed by the Soft Drop distinguisher and the Offline-Online profiling.

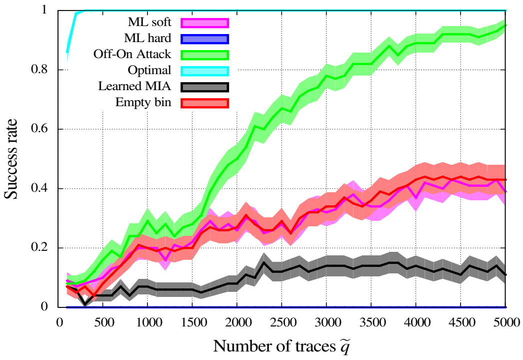

Figure 3: SR **for** $\widehat{q} = 320$ **and** $\sigma = 24$ **on synthetic measurements**

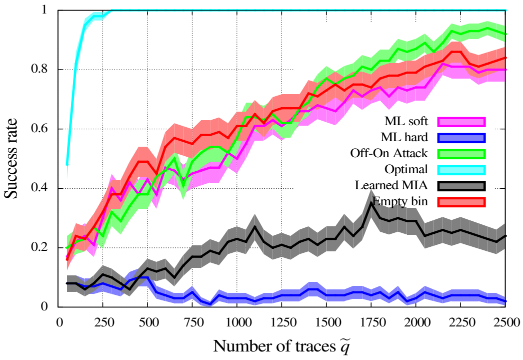

Figure 4: SR **for** $\widehat{q} = 1\,600$ **and** $\sigma = 24$ **on synthetic measurements**

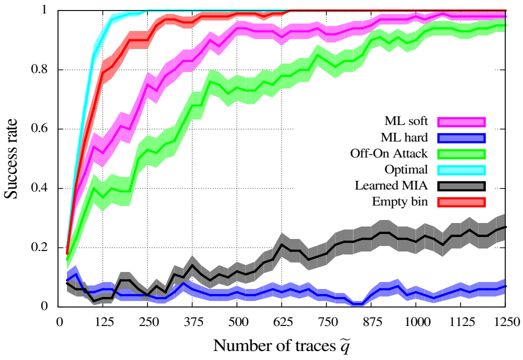

Figure 5: SR **for** $\widehat{q} = 4\,000$ **and** $\sigma = 24$ **on synthetic measurements**

REMARK 3. *In this very special case, we can show that the Empty Bin Distinguisher can accurately approximate the Optimal Distinguisher. Indeed, the actual probability is such that for all* $(x,t) \in \mathscr{X} \times \mathscr{T}$,

$$\mathbb{P}(x|y(k)) = \begin{cases} \frac{1}{2\sigma+1} & \text{if } -\sigma \le x - \phi(t \oplus k) \le \sigma, \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

which is constant if x is in the appropriate interval. For the Empty Bin Distinguisher,

$$\widehat{\mathbb{P}}(x|y(k)) > 0 \implies \mathbb{P}(x|y(k)) = \frac{1}{2\sigma+1}$$

due to the leakage model. Therefore, we can predict that at least $\widehat{q} = (2\sigma+1)|\mathscr{Y}|\frac{1}{\min \mathbb{P}(y)} = 3\,920$ *profiling traces are needed to make sure that the Empty Bin Distinguisher becomes as efficient as the Optimal Distinguisher. As profiling consists in random draws* with replacement, *the* $\mathscr{D}_{\text{Empty\_Bin}}$ *distinguisher is found very close to the* $\mathscr{D}_{\text{Optimal}}$ *distinguisher with* $\widehat{q} = 4\,000$ *profiling traces.*

5. RESULTS ON REAL DEVICES

We have chosen to carry out a timing attack on an STM32F4 discovery board [21]. One interesting aspect is that we do not make any assumption on the model. In real life, the leakage model happens to be much more complex than the one employed in simulations (e.g., Equation (24)). As will be seen, in practice empty bins appear even for the correct key guess and for a "good" profiling phase. This observation differs from the ideal case of our simulations carried out in the preceding Section 4.

5.1 The ARM processor

We used a STM32F4 discovery board by STMicroelectronics. It contains an STM32F407VGT6 microcontroller which has an ARM cortex-M4 MCU with 1 MB flash memory for instructions and data, and a 192 KB Random Access Memory (RAM). The RAM is divided into three sections: one of 16 KB, another one of 112 KB, and a last one consisting of 64 KB Core Coupled Memory (CCM). The CCM has a zero flash wait state and is often used to store critical data such as data from the operating system. Since the RAM is divided into three regions the users are unable to make use of the 192 KB RAM as a continuous memory block.

STM32F4 microcontrollers contain a proprietary prefetch module (Adaptive Real-Time memory accelerator - ART accelerator). ART accelerator contains an instruction cache which has 64 lines and a data cache which contains 8 lines. The line size of both instruction cache and data cache is 128-bits. The precise details about ART accelerator (cache replacement policy and cache associativity) are not mentioned as the module is an intellectual property of STMicroelectronics.

The STM32F407VGT6 microcontroller does not have either a CPU cycle counter or a performance register to measure a cycle accurate time. However the Data Watchpoint and Trace (DWT) unit has a cycle accurate 32 bit counter (DWT_CYCCNT register) which can be used for measuring the duration of critical operations. When processor runs at 168 MHz, the DWT_CYCCNT register will overflow at every 25.5 seconds providing enough time window to measure the encryption / decryption time for an adversary to measure the elapsed time without timer overflowing. In practice, we collected timing data repeatedly within the ARM, and then dump it as large data buffers sporadically. This modus operandi allowed us to reach about 10 000 measurements per second.

We use OpenSSL (version 1.0.2) AES as the cryptographic library, where the SubBytes function is implemented with large 1 KB T-boxes. The AES timing acquisition is illustrated in Figure 6. Before each encryption we reset DWT_CYCCNT register. This yields the exact timing of the AES execution. Of course, a real attacker would measure a noisy timing using an external "chronometer". However, our attack models the best case for an attacker, hence bounds the security of the analyzed implementation.

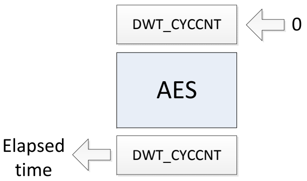

Figure 6: Measuring elapsed time for AES encryption

Time deviations for different configurations of instruction cache and data cache are shown in Figure 7. We observe a huge time difference when data cache is turned Off / On. Even though both IC and DC are turned off, time still varies as the plaintext is changed.

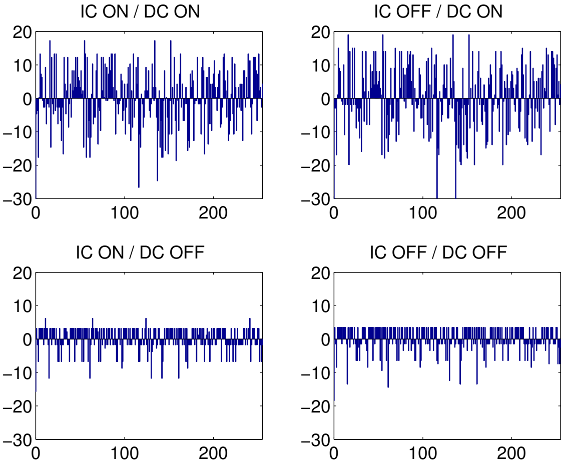

Figure 7: Time deviations for different configurations of Instruction Cache (IC) and Data Cache (DC). The horizontal axis holds $t \in \{0,1\}^8$ and the vertical axis consists in corresponding $x \in \mathbb{Z}$ for a given fixed key k^*

5.2 Attack Results

As already noticed above, the leakage model is mostly unknown. We only suppose that the text byte is mixed with the key through a XOR operation. As a consequence, the optimal distinguisher (giving the limit of performance) is not known. The SNR of the leakage is $\mathsf{Var}(\mathbb{E}(x|t))/\mathbb{E}(\mathsf{Var}(x|t)) = 0.4$.

In Figure 8, we notice that Learned MIA is the best distinguisher in the case of poor profiling. The Hard Drop Distinguisher is not succeeding at all since it drops about 90% of the data.

Figure 9 presents the success rate for a better profiling stage. We notice the following interesting improvements:

- The Learned MIA distinguisher is only slightly better than in Figure 8. To reach 80% success rate, 1 100 traces are needed as compared to 1 250 traces previously.
- The Soft Drop and Offline-Online distinguishers are the best distinguishers in this scenario, with a small advantage for the Soft Drop distinguisher.
- The Hard Drop distinguisher remains unsuccessful.

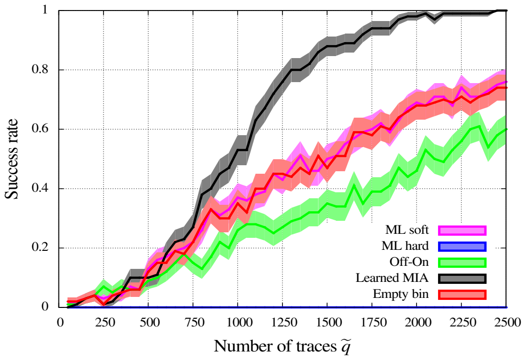

Figure 8: SR **for** $\widehat{q} = 25\,600$ **on real-world measurements**

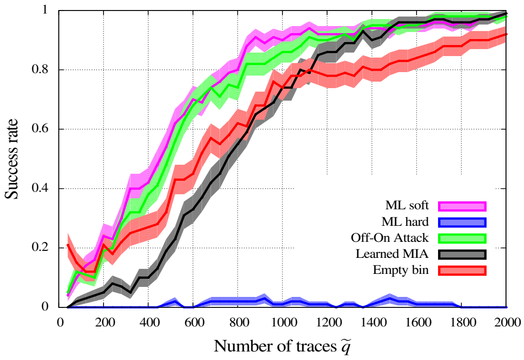

Figure 9: SR **for** $\widehat{q} = 256\,000$ **on real-world measurements**

We notice that the Soft Drop Distinguisher has been established using the γ parameter defined in Equation 19 such that $\gamma = 1/\widetilde{q}$.

Figure 10 is the continuation of Figure 9 with much more traces in the profiling stage. The resulting profiling is very good and one may consider that the approximation of $\mathbb{P}$ is tight. In this case, Soft Drop and OOP Distinguishers are both very successful which seems natural regarding the fact that $\widehat{\mathbb{P}}$ has converged to the actual probability $\mathbb{P}$.

As a conclusion to this study on the STM32F4 discovery board, we have learned the following comparisons between the proposed distinguishers:

- when the profiling stage is poor, the best distinguisher is the Learn MIA Distinguisher;
- when there is enough data in the profiling stage, the best distinguisher is the Soft Drop Distinguisher, closely followed by the OOP Distinguisher;
- the Empty Bin Distinguisher converges to the optimal success rate, but is not as efficient as previously in Section 4. This can be explained by the fact that we skip a lot of data in the computation;
- the Hard Drop Distinguisher is the slowest to converge to 100% success rate.

REMARK 4. *When comparing Figures 9 and 10, we notice that the Empty Bin distinguisher does not improve as the number of profiling traces increases. An explanation that there is no more empty bins to be filled between these two situations; then only a more precise estimation of the probability would make the difference.*

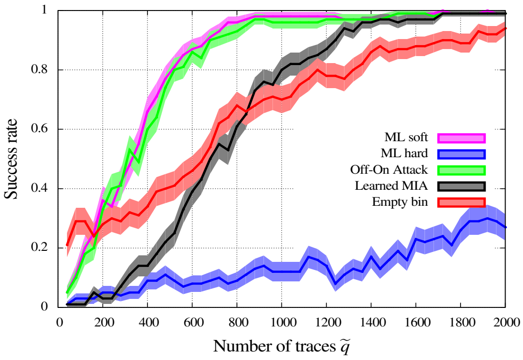

Figure 10: SR for $\widehat{q} = 2\,560\,000$ on real-world measurements

6. CONCLUSION AND PERSPECTIVES

We have derived several "information-theoretic" distinguishers as possible solutions to the empty bin issue. Some of them, like the Dirichlet Prior and the Offline-Online distinguishers, required the computation of novel distributions. We have shown in particular that the empty bins, previously believed to be an annoyance and dropped accordingly, can turn out to be valuable assets for the attacker as long as they are treated carefully.

We have also compared the various distinguishers under two frameworks: a simulated test with synthetic leakage and real-world timing attacks. In both cases, we noticed that the attacks outcome depends on the quality of the profiling stage. A good profiling improves the results, where the best distinguisher seems to be the Soft Drop Distinguisher. A poor profiling makes the traditional distinguishers beak down. More sophisticated solutions like Offline-Online Profiling and Learned MIA distinguishers are very useful in this case.

The interesting aspect on the studied timing attack is that one does not have to make any assumption on the leakage model. In addition, the main advantage of the new distinguishers is that the empty bin issue is completely solved. We also introduced distinguishers which can jointly exploit offline and online side-channel measurements. As an interesting perspective, our approach could advantageously be analyzed using the "perceived information" metric recently introduced by Standaert et al. in [22, Eqn. (1)].

Another perspective would be to compare our information-theoretic attacks with attacks based on machine learning techniques. Surprisingly, and contrary to results reported in other papers, our preliminary results show that SCA based on support vector machines [23] has poor performance, even when profiling with very few traces ($\widehat{q}$ is small), which may be due to the univariate nature of the leakage.

REFERENCES

[1] J. Daemen and V. Rijmen, *The Design of Rijndael: AES – The Advanced Encryption Standard*. Springer, 2002.

[2] W. Schindler, "A Timing Attack against RSA with the Chinese Remainder Theorem," in *CHES* (Ç. K. Koç and C. Paar, eds.), vol. 1965 of *Lecture Notes in Computer Science*, pp. 109–124, Springer, 2000.

[3] W. Schindler, "Optimized timing attacks against public key cryptosystems," *Statistics & Risk Modeling*, vol. 20, no. 1-4, pp. 191–210, 2002. DOI: 10.1524/strm.2002.20.14.191.

[4] D. Brumley and D. Boneh, "Remote timing attacks are practical," in *Proceedings of the 12th USENIX Security Symposium, Washington, D.C., USA, August 4-8, 2003*, USENIX Association, 2003.

[5] B. B. Brumley and N. Tuveri, "Remote Timing Attacks Are Still Practical," in *ESORICS* (V. Atluri and C. Díaz, eds.), vol. 6879 of *Lecture Notes in Computer Science*, pp. 355–371, Springer, 2011.

[6] 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, (New York, NY, USA), pp. 7–12, ACM, 2014.

[7] S. Chari, J. R. Rao, and P. Rohatgi, "Template Attacks," in *CHES*, vol. 2523 of *LNCS*, pp. 13–28, Springer, August 2002. San Francisco Bay (Redwood City), USA.

[8] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Advances in Cryptology - CRYPTO '96, Santa Barbara, California, USA, August 18-22, 1996, Proceedings* (N. Koblitz, ed.), vol. 1109 of *LNCS*, pp. 104–113, Springer, 1996.

[9] D. J. Bernstein, "Cache-timing attacks on AES," April 14 2005. http://cr.yp.to/antiforgery/cachetiming-20050414.pdf

[10] C. Rebeiro and D. Mukhopadhyay, "Boosting Profiled Cache Timing Attacks With A Priori Analysis," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 6, pp. 1900–1905, 2012.

[11] M. Weiß, B. Heinz, and F. Stumpf, "A cache timing attack on AES in virtualization environments," in *Financial Cryptography and Data Security - 16th International Conference, FC 2012, Kralendijk, Bonaire, Februray 27-March 2, 2012, Revised Selected Papers* (A. D. Keromytis, ed.), vol. 7397 of *LNCS*, pp. 314–328, Springer, 2012.

[12] S. Bhattacharya, C. Rebeiro, and D. Mukhopadhyay, "Hardware prefetchers leak: A revisit of SVF for cache-timing attacks," in *45th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2012, Workshops Proceedings, Vancouver, BC, Canada, December 1-5, 2012*, pp. 17–23, IEEE Computer Society, 2012.

[13] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, pp. 1065–1076, 09 1962.

[14] F.-X. Standaert, T. Malkin, and M. Yung, "A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks," in *EUROCRYPT*, vol. 5479 of *LNCS*, pp. 443–461, Springer, April 26-30 2009. Cologne, Germany.

[15] A. Heuser, O. Rioul, and S. Guilley, "Good Is Not Good Enough - Deriving Optimal Distinguishers from Communication Theory," in *CHES 2014, Busan, South Korea, September 23-26, 2014. Proceedings* (L. Batina and M. Robshaw, eds.), vol. 8731 of *LNCS*, pp. 55–74, Springer, 2014.

[16] B. A. Frigyik, A. Kapila, and M. R. Gupta, "Introduction to the Dirichlet Distribution and Related Processes," Tech. Rep. 206, 2010.

[17] A. Moradi and F. Standaert, "Moments-correlating DPA," *IACR Cryptology ePrint Archive*, vol. 2014, p. 409, June 2 2014.

[18] N. Veyrat-Charvillon and F.-X. Standaert, "Mutual Information Analysis: How, When and Why?," in *CHES*, vol. 5747 of *LNCS*, pp. 429–443, Springer, September 6-9 2009. Lausanne, Switzerland.

[19] E. Prouff and M. Rivain, "Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis," in *ACNS* (Springer, ed.), vol. 5536 of *LNCS*, pp. 499–518, June 2-5 2009. Paris-Rocquencourt, France.

[20] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel, "Mutual information analysis," in *CHES, 10th International Workshop*, vol. 5154 of *LNCS*, pp. 426–442, Springer, August 10-13 2008. Washington, D.C., USA.

[21] "STM32F4DISCOVERY Discovery kit with STM32F407VG MCU." http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419?sc=internet/evalboard/product/252419.jsp [Accessed March 19, 2016].

[22] M. Renauld, D. Kamel, F.-X. Standaert, and D. Flandre, "Information Theoretic and Security Analysis of a 65-Nanometer DDSLL AES S-Box," in *CHES* (B. Preneel and T. Takagi, eds.), vol. 6917 of *LNCS*, pp. 223–239, Springer, 2011.

[23] A. Heuser and M. Zohner, "Intelligent Machine Homicide - Breaking Cryptographic Devices Using Support Vector Machines," in *COSADE* (W. Schindler and S. A. Huss, eds.), vol. 7275 of *LNCS*, pp. 249–264, Springer, 2012.