Improving Side-Channel Attacks against Pairing-Based Cryptography

Damien Jauvart[1,2], Jacques J.A. Fournier[1], Nadia El Mrabet[3], and Louis Goubin[2]

[1] CEA-Tech PACA, Gardanne, France,
damien.jauvart2@cea.fr, jacques.fournier@cea.fr,
[2] UVSQ-PRiSM, Versailles, France,
louis.goubin@uvsq.fr
[3] EMSE, Gardanne, France,
nadia.el-mrabet@emse.fr

Abstract. Although the vulnerability of pairing-based algorithms to side-channel attacks has been demonstrated—pairing implementations were targeted on three different devices in a recent paper [41]—it nevertheless remains difficult to choose an adapted leakage model and detect points of interest. Our proposed approach evaluates the parameters of the attack and validates the data processing workflow. We describe weaknesses in the implementation of cryptographic pairings, and we show how information leakage can be fully exploited. Different leakage models, point-of-interest detection methods, and parameter dependencies are compared. In addition, practical results were obtained with a software implementation of twisted Ate pairing on Barreto–Naehrig curves with an ARM Cortex-M3 processor running at 50 MHz. We discuss countermeasures aimed at reducing side-channel leakage and review the available literature.

Keywords: pairing-based cryptography, twisted Ate pairing, Miller's algorithm, side-channel attack, points of interest, countermeasures

1 Introduction

Side-channel attacks, which aim to recover secret data, are a serious threat to cryptographic devices. With embedded systems, the attacker can easily gain physical access to the device. Thus, side-channel attacks are a high-level concern [13,26,27]. Because identity-based encryption (IBE) [6] systems are not immune to these threats, the vulnerability of pairings used in IBE systems should be investigated. The basic modular multiplication algorithm used during a pairing calculation was recently attacked through correlation power analysis (CPA) [5,41].

Over the past few years, several works have highlighted the threat posed by attacks that target precise arithmetic operations during pairing computations. Side-channel attacks are based on exploiting the link between known (possibly

malleable) data and secret data. A control device allows the attacker to execute a cryptographic algorithm with several known inputs. In IBE, such interactions appear during the decryption step. If the ciphertext to decrypt is $\{U, V\}$, then the first step consists in computing $e(s, U)$, where s is the secret key. The pairing algorithm then performs arithmetic operations between both sets of data. The attacks highlighted in [5,41] specifically target a modular multiplication algorithm. Once the target has been identified, a suitable leakage model must recreate the side-channel induced by calculating the targeted operation.

Studies on side-channel attacks share at least two important characteristics: the comparison of side-channel leakage models and the detection of points of interest associated with the models. The statistic tests that are used to detect points of interest can also be considered validators of the leakage model. In fact, if the statistical tool results in significant peaks, then the model can be validated. Our approach concerns a parameterized attack. Because of the large number of variables, we provide a detailed characterization of how side-channel attacks leak information concerning critical operations during pairings.

This study proposes a generic method for attacking pairing implementations and defines parameters to increase CPA efficiency (in terms of the number of measurement curves needed). To illustrate the application of our approach in the context of a cryptographic algorithm, we targeted one of the modular multiplications involved in the software implementation of an Ate pairing with the aim to retrieve (the secret) one of the two points in the pairing calculation. Compared with the best attacks on pairing calculations published so far [41], our results, based on taking real electromagnetic measurements on the chip of an embedded 32-bit ARM core processor, required significantly less computational time to retrieve the secret value.

The paper is organized as follows. Section 2 reviews existing research pertinent to the subject of the present paper, Section 3 gives some background information on pairing implementations, and Section 4 proposes an analysis of some general and specific techniques to defeat side-channel attacks. In addition, we describe our experimental results obtained with different techniques for the proposed attack scheme. Finally, possible countermeasures are discussed in Section 5, followed by our concluding statements in Section 6.

2 Related work

Side-channel attacks on cryptographic algorithms have been studied extensively for more than two decades. Attacks targeting public key algorithms such as RSA or elliptic curve cryptography (ECC) have mainly been of the simple power analysis (SPA) type, whose objective is to reveal the secret exponent (in RSA) or the secret scalar (in ECC) used in a signature/decryption scheme. These algorithms use "public" variables, a long precision message (in RSA) or a base point (in ECC), that do not need to be attacked. One of the rare exceptions to this is a CPA-like attack on the final subtraction of a Montgomery Modular Multiplication (MMM), as described in [36]. CPA-type attacks on public key algorithms

began to appear in attacks on implementations that were secured against SPA. For example, Joye [22] discusses this type of attack on protected versions of ECC. CPA attacks on algorithms such as RSA have been used to target protected implementations of the algorithm with a "horizontal" approach [10,33]: the approach is horizontal in the sense that the statistical correlation analysis is done on portions of the same measured side-channel curve to defeat the random mask that is used as a countermeasure.

"Vertical" CPA (statistical correlation analysis of several measured side-channel curves for different input values) is relevant to and mainly studied in pairing-based cryptography (PBC), which is a field of public-key cryptography. When pairings are used (e.g., in IBE schemes), one of the two points of the pairing calculation is the secret decryption key; hence, it makes sense to use (vertical) CPA to attempt to retrieve this key.

Several papers have addressed side-channel attacks on pairings of fields in characteristic 2 or 3. These studies are merely mentioned for reference, considering that our implementation is based on large prime fields. Page and Vercauteren [31] published the first paper describing physical attacks (passive side-channel attacks and active fault attacks) on pairing algorithms. They targeted the Duursma–Lee algorithm [16], which is used to compute Tate pairings on elliptic curves over finite fields in characteristic 3. Data manipulation during the Duursma–Lee algorithm involves the product of a secret data item and a value derived from the known input point. The authors propose an SPA-like attack on field multiplication algorithms that are implemented using the shift-and-add method. They additionally describe a DPA attack that aims to recover the secret one bit at a time. Kim *et al.* [24] proposed that timing, SPA, and DPA attacks used to target arithmetic operations also concern pairings over binary fields. In the context of Eta pairings over fields in characteristic 2, the targeted operation is $a(b+r)$, where a and b are derived from the secret, and r is derived from the known input. The authors conclude that, theoretically, the bitwise DPA proposed by Page and Vercauteren [31] would still be able to recover the secret point used in the pairing calculation. Pan and Marnane [32] proposed a practical CPA attack based on a Hamming distance model on an Eta pairing over a base field in characteristic 2 over supersingular curves.

One of the first papers describing side-channel attacks on pairings over large prime fields was proposed by Whelan and Scott [42], who used CPA to target the arithmetic operations to recover the secret: they calculated correlations between hypothetical outputs of the arithmetic operation $x \times k$ for all possible keys k and leakage traces. The resulting correlation curves were obtained for each key hypothesis; the correct one was the hypothesis with the highest peak. In the same paper, the authors discussed using word length (8, 16, 32, or 64) to represent long precision numbers; they further explain how partial correlation calculations can be used with CPA to target a portion of the word. El Mrabet *et al.* [17] later proposed the first practical side-channel attack on Miller's algorithm for a pairing over prime fields equal to 251. The tangent line equation was targeted because it involves a modular multiplication of a coordinate derived from a

public input point by a deterministic value derived from the secret point. Ghosh *et al.* [18] detailed a DPA attack on the modular subtraction in a Tate pairing over a Barreto–Naehrig elliptic curve [3]. Blömer *et al.* [5] then described side-channel attacks on modular additions and multiplications of finite field elements with large prime characteristics, showing that these attacks are possible even if the secret point is used as the first argument of the pairing calculation; their results were based on simulations. Unterluggauer and Wenger [41] have authored the most recent paper to investigate the use of SCA to target pairings. Using a CPA-like approach, as previously described in [42] for example, they targeted the modular operations during an Ate pairing to find the secret 16 bits at a time, taking advantage of the fact that the processor running the pairing calculation works with a 16-bit multiplier. Their configuration required more than 1500 measured curves to find the correct secret point.

3 Pairing-based cryptography

A pairing e is a bilinear and nondegenerate map such that $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$, where $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$ are cyclic groups of the same prime order r. Let q be a prime number, let E be an elliptic curve over $\mathbb{F}_q$, and let r be a prime divisor of $\#E(\mathbb{F}_q)$. Efficient pairing algorithms are realized with $\mathbb{G}_1$, $\mathbb{G}_2$ subgroups of an elliptic curve $\#E\left(\mathbb{F}_{q^k}\right)$ with a point at infinity $\mathcal{O}$, and $\mathbb{G}_3$ is the subgroup of the r^{th} roots of unity in $\mathbb{F}_{q^k}$, where k is the smallest integer such that r divides $(q^k - 1)$. A complete study of pairing-friendly elliptic curves can be found in [38]. The following properties complete the definition of a pairing:

- Nondegeneracy: $\forall P \in \mathbb{G}_1 \setminus \{\mathcal{O}\} \quad \exists Q \in \mathbb{G}_2$ such that $e(P, Q) \neq 1$, and $\forall Q \in \mathbb{G}_2 \setminus \{\mathcal{O}\} \quad \exists P \in \mathbb{G}_1$ such that $e(P, Q) \neq 1$,
- Bilinearity: $\forall a, b \in \mathbb{Z}, \forall P \in \mathbb{G}_1$ and $\forall Q \in \mathbb{G}_2$ then

$$e([a]P, [b]Q) = e(P, Q)^{ab}. \tag{1}$$

With the notation $[a]P = \underbrace{P + \ldots + P}_{a \text{ times}}$. More detailed definitions of pairings can be found in [39]; here, we are interested in physical attacks on cryptosystems that are based on Ate pairings.

For a 128-bit security level, Barreto–Naehrig (BN) curves [3] offer the highest security-level-to-computation-time ratio. Such curves take the form $E : y^2 = x^3 + b$ over a finite field $\mathbb{F}_q$, where $b \neq 0$ and q is a large prime integer.

For BN curves, the parameters q and r are defined as follows:

$$\begin{aligned} q(t) &= 36t^4 + 36t^3 + 24t^2 + 6t + 1, \\ r(t) &= 36t^4 + 36t^3 + 18t^2 + 6t + 1, \end{aligned} \tag{2}$$

for some $t \in \mathbb{Z}$ such that q is prime. Note that such curves have an embedded degree of $k = 12$.

The notation $E(\mathbb{F}_q)[r]$ is used to denote the $\mathbb{F}_q$-rational r-torsion group of E, (i.e., the set of points P in $E(\mathbb{F}_q)$ such that $[r]P = \mathcal{O}$).

Let $\mathbb{G}_1 = E(\mathbb{F}_q)[r] \cap \ker(\pi_q - [1])$, and let $\mathbb{G}_2 = E\left(\mathbb{F}_{q^{12}}\right)[r] \cap \ker(\pi_q - [q])$, where π_q is the Frobenius endomorphism $\pi_q : E \to E : (x, y) \mapsto (x^q, y^q)$, and let $e = k/d$, where d is the degree of the twist, here $d = 6$. Let t be the trace of the Frobenius map over E.

Ate pairing [14,20] over BN curves gives the map

$$\begin{aligned} e : \mathbb{G}_1 \times \mathbb{G}_2 &\to \mathbb{F}_{q^{12}}^{\star} \\ (P, Q) &\to f_{(t-1)^e, P}(Q)^{\frac{q^k-1}{r}}. \end{aligned} \tag{3}$$

If the curves admit a sextic twist, then the elements of $E\left(\mathbb{F}_{q^{12}}\right)$ can be on the twisted curve $E'\left(\mathbb{F}_{q^2}\right)$. This improves processing efficiency considerably because the first input point P is now stored as two integers in $\mathbb{F}_q$ instead of as twelve integers.

Miller [28] provides an efficient method for calculating such pairings: Miller's algorithm is the main part of the pairing computation.

We recall the computation of twisted Ate pairings over BN curves using Miller's loop in Algorithm 1.

Algorithm 1: Computation of twisted Ate pairings using Miller's loop over BN curves

Input : $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$, t the Frobenius trace of E
Output : $e(P, Q)$

$T \leftarrow P$;
$f \leftarrow 1$;
$n \leftarrow t - 1$; // $n = (n_{w-1} \dots n_0)_2$ radix 2 representation
for $i = w - 2$ **downto** 0 **do**
    $f \leftarrow f^2 \cdot l_{T,T}(Q)$;
    $T \leftarrow [2]T$;
    if $n_i == 1$ **then**
        $f \leftarrow f \cdot l_{T,P}(Q)$;
        $T \leftarrow T + P$;
    end
end
return $f^{\frac{q^{12}-1}{r}}$;

4 Analyzing information leakage in side-channel attacks

From a theoretical standpoint, the security level of cryptographic algorithms corresponds to the level of computational difficulty of a well-known mathematical problem. In practice, the implementation of those cryptographic algorithms

has to be tested for their resistance against physical attacks. Today, studies on physical attacks that aim to retrieve the secret keys used during cryptographic calculations represent a growing field of research, especially because cryptography is now being deployed in billions of connected objects.

Identity-based encryption (IBE) schemes solve several problems concerning the coupling of connected objects. In the context of pairing-based IBE implementations, the computational issues are solved by using pairing over elliptic curves. The principles of side-channel attacks are as follows: the decryption phase is calculated with a pairing between a point derived from the ciphertext (known) and a secret point, which constitutes the key. Hence, the aim of a side-channel attack is to target such pairing calculations in order to retrieve the secret key.

In pairing calculations, these critical operations are modular multiplications such as those identified in [5,31,41,42]. We describe how to identify these types of failures in Subsection 4.1.

In Subsection 4.2, we present a detailed study of the basic multiplication operation, which constitutes the basic building block of most public-key cryptographic algorithms, and provide a validated leakage model. So far, we have described a systematic method (based on predefined models) that finds the best parameters for using CPA to target a multiplication operation and, by extension, a modular multiplication, requiring only around 150 curves.

In the following, we use an efficient attack on a pairing computation to validate the usefulness of our approach.

4.1 Side-channel attack strategy to target Miller's algorithm

Operations that occur during the pairing computation involve both known and secret data. This is the case in Algorithm 1 for the computation of the tangent line (see Line 8 in Algorithm 1). This interaction takes the form of a modular multiplication.

In our implementation, as is often the case in practice, the tangent line equation $l_{T,T}(Q)$ in Equation 4 is in mixed affine–Jacobian coordinates. The equation of the tangent at T is evaluated at the point Q.

$$l_{T,T}(Q) = \frac{2y_Q Y_T Z_T^3 - 2Y_T^2 - (3X_T^2 + aZ_T^4)(x_Q Z_T^2 - X_T)}{2Y_T Z_T^3}. \tag{4}$$

For optimization, this equation can be written in mixed system coordinates as suggested in [1,25]:

- P and Q are in affine coordinates.
- T is in Jacobian coordinates.

The point T is initialized with P by $X_T \leftarrow x_P, Y_T \leftarrow y_P$ and $Z_T \leftarrow 1$ before Miller's loop. Thus, for the first iteration T is equal to P. Therefore, if we recover T, then we will directly obtain the secret P. Even if the input point is either P or Q, we can see that the multiplication $2y_Q Y_T Z_T^3$ involves known and secret data. We then attack the modular multiplication as described in Section 4.2.

Case 1. P is the secret. In this case, we want to recover P (or T) with a side-channel attack. Our target is therefore the multiplication $(2y_Q) \cdot Y_T$. Knowledge of y_Q allows us to build a CPA to recover the coordinate $Y_T = Y_T^{(0)} + uY_T^{(1)} \in \mathbb{F}_{q^2}$. The multiplication $(2y_Q) \cdot Y_T$ applies to elements of $\mathbb{F}_q$, and $\mathbb{F}_{q^2}$ is similar to two multiplications in $\mathbb{F}_q$, that is, $(2y_Q) \cdot Y_T = (2y_Q) \cdot Y_T^{(0)} + u(2y_Q) \cdot Y_T^{(0)}$. Thus, a first CPA attack must target $(2y_Q) \cdot Y_T^{(0)}$ to recover $Y_T^{(0)}$, and a second attack then targets $Y_T^{(0)}$.

Case 2. Q is the secret. In order to recover the input point Q, we target the modular multiplication $(2y_Q) \cdot Y_T$. After recovering $2y_Q \in \mathbb{F}_q$, we have y_Q, and we use the elliptic curve equation to recover x_Q.

4.2 Our attack principle and practical applications

We have seen that targeting the pairing amounts to an attack on a modular multiplication. We are not concerned with the method used to compute this multiplication (see Booth [7], Toom-Cook [11,40], Karatsuba [23], Brickell [8], Montgomery [29], or Quisquater [15,35]) because it is unimportant which method is chosen. The algorithm goes through a step of smaller integer multiplication. The size of these integers depends on the architecture of the device, for example, an integer of 256 bits needs to be stored in $n_{\text{word}} = 8$ registers of 32 bits in a 32-bit architecture.

In the following, we describe the processing chain of our attack. The aim is to understand the leakages induced by the multiplier during processing of the multiplication of two "small" integers (32 bits, for instance).

Using correlation power analysis to target multiplication We target the secret input $k = (k_{n-1} \dots k_0)_2$ involved in $k \times x$. First, we record side-channel traces of this operation for several values of x. For all known inputs x and for all possible keys, we compute hypothetical outputs of the product $k \times x$. Then, we calculate correlations between the hypothetical outputs and the measured side-channel traces. To this end, we use the scheme detailed in Algorithm 2 to store two big matrices: the outputs and the traces.

Practical set-up In order to support the method, we put in parallel our practical results. The targeted device is an ARM Cortex M3 processor working on 32-bit length registers. To target the multiplication operation, we place a trigger in the C code before this operation for synchronisation. This step is used for recording the traces just during the targeted time interval. The electromagnetic emanation (EM) measurements were done using a Langer EMV-Technik LF-U 5 probe equipped with a Langer Amplifier PA303 BNC (30dB). The curves were collected using a Lecroy WaveRunner 640Zi oscilloscope. The acquisition frequency of the oscilloscope is 10^9 samples per second.

Algorithm 2: Using correlation power analysis to target multiplication

Input : $\mathcal{C}^{(l)}, \forall l = 1, \ldots, N$ the curves associated with $k \times x^{(l)}$ sampled on m points
Output : $\widehat{k}$ candidate for k

1 H is an empty matrix in $\mathcal{M}_{N \times 2^n}$;
2 T is an empty matrix in $\mathcal{M}_{N \times m}$;
3 **for** $l = 1$ **to** N **do**
4 $T(l, \cdot) \leftarrow C^{(l)}$; `// Store the traces`
5 **for** $j = 0$ **to** $2^n - 1$ **do**
6 $H(l, j+1) \leftarrow \phi(j * x^{(l)})$; `//` ϕ`(Hypothetical output)`
7 C is an empty matrix in $\mathcal{M}_{2^n \times m}$;
8 **for** $i = 1$ **to** m **do**
9 **for** $j = 1$ **to** 2^n **do**
10 $C(j, i) \leftarrow corr(T(\cdot, i), H(\cdot, j))$; `// Correlation between traces and predictions`
11 $(\widehat{k}, t) \leftarrow argmax_{i,j} |C|$;
12 **return** $\widehat{k}$;

Statistical tests to evaluate leakage models Ideally, CPA will recover the secret k if the leakage model ϕ is well chosen. In Algorithm 2, Line 6 can take numerous forms. Because we assume that the device leakage follows a Hamming weight (HW) model [13,27,30], the Hamming weight is a classic choice for ϕ. At the beginning, we considered two HW models for ϕ:

- $c = k \times x = (c_{2n-1} \ldots c_0)_2$ then $\phi_1(k, x) = \sum_{i=0}^{n-1} c_i$,
- $c = k \times x = (c_{2n-1} \ldots c_0)_2$ then $\phi_2(k, x) = \sum_{i=0}^{2n-1} c_i$.

Note that by taking the n least significant bits in the ϕ_1 model, we take in fact the bits of $a \times b \mod 2^n$.

We evaluate both models by computing the t-test (also known as the sum of squared pairwise t-differences [SOST] [19]). To this end, we use the fixed key and the variable plaintext obtained through our 1000 trace measurements. For each trace, we compute the supposed leakage $\phi(k, x)$, and we add the trace to the associated set. The size of each set is stored in $\eta_{\phi,i}, i = 1, \ldots, N_\phi$. In our case, $n = 8$; thus, for ϕ_1, there are $N_{\phi_1} = 9$ sets (9 possible HWs), and $N_{\phi_2} = 17$ for ϕ_2. We compute the mean $m_{\phi,i}$ for $i = 1, \ldots, N_\phi$ and the variance $\sigma^2_{\phi,i}$ of each set for ϕ_1 and ϕ_2. Thus, we are able to compute the SOST value for both models:

$$SOST_\phi = \sum_{i,j=1}^{N_\phi} \left(\frac{m_{\phi,i} - m_{\phi,j}}{\sqrt{\frac{\sigma^2_{\phi,i}}{\eta_{\phi,i}} + \frac{\sigma^2_{\phi,j}}{\eta_{\phi,j}}}} \right)^2 \quad \text{for } i \geq j \tag{5}$$

Figure 1 illustrates our experimental results. The leak is visibly confirmed for model ϕ_2: the peak is clearly always higher in this second case.

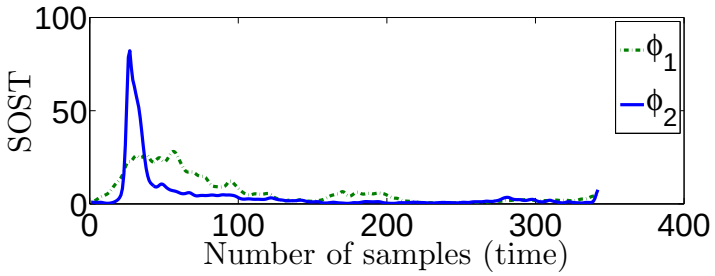

Fig. 1: The sum of squared pairwise t-differences (SOST)

Divide and conquer Using the previously described leakage model, we apply CPA in sequence to all four bytes of k in order to retrieve all 32 bits of k. First, we want to recover the 8 least significant bits of k, this is the attack of Algorithm 2, with either model ϕ_1 or ϕ_2. In this first instance of CPA, the 2^8-key hypotheses consider different values for plaintexts x. We thus obtain 2^8 values for the coefficient correlations. At this stage, we define an α-parameter, which means we will retain the key α-hypothesis corresponding to the best α-correlations. Then, for each of the α-hypotheses, we use CPA to target the following 8 bits of k. For each α, we also retain the best α-candidates. At the end of this step, the $\alpha \times \alpha$ key hypotheses correspond to the 16 least significant bits of k. We perform this process a third time to select candidates for the 24 least significant bits of k. The fourth step is identical to step three, and the candidate $\widehat{k}$ for k is the key corresponding to the best correlation found after this fourth instance of CPA.

Effects of the α-parameters. Even though our comparison is based on differing α-values—$\alpha = 64$ with our method and $\alpha = 5$ for the attack proposed in [41]—Unterluggauer *et al.* specify that varying the α-parameter did not affect success for their attack. In fact, they observed no significant difference between $\alpha = 5$ and $\alpha = 10$.

Figure 2 shows the evolution of the success rate with respect to the number of traces used and the α-parameters of the CPA targeting the first 32-bit word. For each database size, the height of the bars of the corresponding column increases with α.

For example, for a database with 80 traces and with $\alpha \geq 40$ the success rate of the attack is greater than 80%. For 110 traces and $\alpha \geq 28$ the success rate of the attack is greater than 95%.

Resource comparison with Unterluggauer et al. [41]. Our strategy consists in dividing $32 = 4 \times 8$ bits for a case with $\alpha = 64$ (a large α-value). By contrast,

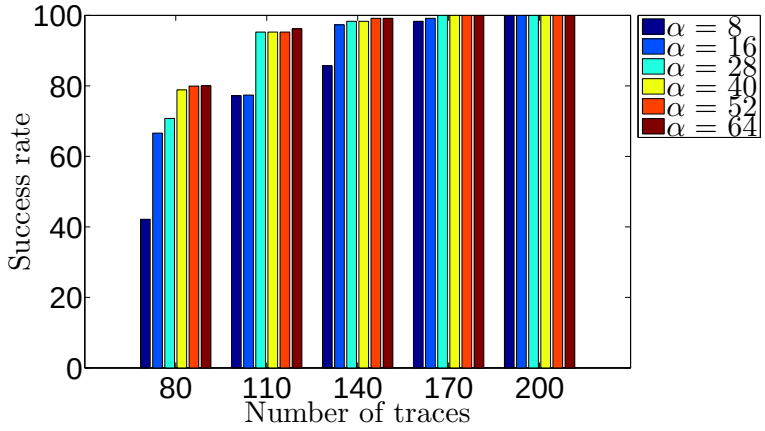

Fig. 2: Success rate for different database sizes and α -values

Unterluggauer *et al.* divided $32 = 2{\times}16$ bits with $\alpha = 5$. The resource comparison given in Table 1 quantifies the differences between both methods; "time" denotes the number of enumerated subkeys, and "memory" represents the resources used to store the subkeys.

	Unterluggauer *et al.*	**Our method (with $\alpha = 64$)**
Time	$2^{18} < 2^{16} + 5 \times 2^{16} < \mathbf{2^{19}}$	$2^{15} < 2^8 + \alpha \times 2^8 + \alpha \times 2^8 + \alpha \times 2^8 < \mathbf{2^{16}}$
Memory	$2^{18} < 5 \times 2^{16} < \mathbf{2^{19}}$	$\alpha \times 2^8 = \mathbf{2^{14}}$

Table 1: Resource comparison

4.3 Practical attack on the pairing algorithm

We implemented a twisted Ate pairing over BN curves in a real environment on an ARM Cortex M3 processor by manipulating 256-bit-long integers in $n_{\text{word}} = 8$ words of 32 bits. We ran our algorithm and carried out experimental side-channel attacks using the same setup as described above. We chose to put the secret in the second input point Q. Therefore, thanks to our knowledge of P, we were able to use our attack to recover the secret, word for word, as previously described in Section 4.1.

Our method allowed us to recover the eight 32-bit words of the secret point of a pairing calculation using only around 150 curves. To enable a comparison with previously published practical results [41], we implemented the method described in [41] and ran the analysis on our curves. In terms of the required

number of curves, our method only used 150 traces, compared with 1500 in [41]. Moreover, because we are working on 8-bit words ([41] are working on 16-bit words), our method is much faster. In addition, optimum leakage models can be identified based on our characterization of the multiplication calculation.

5 Countermeasures and prospects

In this paper, we have shown how a thorough study of side-channel attacks—from leakage to multiplication—can be used to improve the attack. We were able to carry out an optimized side-channel attack on a pairing algorithm. Consequently, we are interested in how to protect implementations from such attacks. There are many methods to protect an implementation, for example the physical countermeasures. It based on create noise around the execution of sensitive operations. Here we are interrelated in the "mathematical" countermeasures.

Several countermeasures have already been proposed to protect pairing-based cryptographic algorithms against the kind of side-channel attacks described in the present paper. Most of these countermeasures aim to eliminate any predictable link between the manipulated data and the known input. In practice, pairing computations use various randomization levels. One category of countermeasures consists in randomizing the inputs before the pairing computation, another consists in adding a random mask directly to Miller's algorithm. In addition, a method based on arithmetic randomization can be adapted for pairing-based algorithms.

Input randomization. Page and Vercauteren [31] proposed two countermeasures for their passive attack. The first one is based on the pairing bilinearity. Let a and b be two random values; thus, $e([a]P,[b]Q)^{1/ab} = e(P,Q)$. For each pairing computation, it is therefore possible to take different values for a and b and compute $e([a]P,[b]Q)^{1/ab}$. Evidently, this method is very costly in terms of computation time. Moreover, the randomization itself can be a target for side-channel attacks. In fact, some papers [4,9,10,33,34] have proposed horizontal attacks, which constitute a threat for protected exponentiation with a single trace.

The authors of [31] proposed another method (applicable, for example, to cases where P is secret) that consists in adding the mask to the point Q in the following way: select a random point $R \in \mathbb{G}_2$ and compute $e(P,Q+R)e(P,R)^{-1}$ instead of $e(P,Q)$, with different values of R at every call to e.

Based on this countermeasure, Blömer *et al.* [5] proposed to improve the Tate pairing. For a reduced Tate pairing, they note that the set of the second input argument is the equivalence class ${}^{E(\mathbb{F}_{q^k})}/_{rE(\mathbb{F}_{q^k})}$. They therefore choose a random point $R \in E(\mathbb{F}_{q^k})$ with order l and coprime to r. Thus, $Q + R \sim Q$. Hence, $e(P,Q+R) = e(P,Q)$. This method avoids the second pairing computation that is used to find the same result without a mask.

Randomization of intermediate variables. In 2005, Scott [37] proposed a countermeasure that involves randomizing the Miller variable. In this case, we would

multiply instructions 5 and 8 in Algorithm 1 by a random $\lambda \in \mathbb{F}_q$, eliminated by the final exponentiation. This countermeasure is ineffective against our attack.

Kim *et al.* [24] use the third countermeasure proposed by Coron [12] (based on random projective coordinates) in order to protect the Eta pairing in characteristic 2. However, this countermeasure can be adapted to pairing algorithms that are based on large prime field characteristics. At the beginning of the algorithm, the authors implement this randomization based on the homogeneity of projective or Jacobian coordinates.

Arithmetic randomization. All previous attacks on pairing algorithms have targeted arithmetic operations. The ability to secure multiplications was originally investigated in [21] to protect ECDSA against side-channel attacks with the aim to prevent all possible predictions during a modular multiplication. A "mask" is randomly chosen before processing a multiplication, rendering any hypothesis concerning the output of the internal modular multiplication impossible. Another masking technique proposed in [4] also aims to eliminate any predictable link between known and secret data directly in the arithmetic operations.

In addition, the well-known residue number system can be used to protect arithmetic operations [2].

Although arithmetic protection seems to be a robust method to protect against side-channel attacks, overhead costs must be evaluated. In fact, significant costs are associated with permutation changes in randomized multiplication and with base refreshing in RNS implementations.

Because none of these methods have been validated in the literature, we will apply the proposed countermeasures to our attack to measure their effectiveness. Different α-parameters were used in our attempts to defeat these countermeasures.

6 Conclusion

In this paper, we propose a revised version of the CPA attack provided in [41]. In fact, our investigation constitutes one of the first attempts to experimentally validate side-channel attacks on pairing-based algorithms. The paper makes two principal contributions: 1) We established the differences between two leakage models and described how to choose the appropriate model. The model is selected on the basis of using statistical tools applied to the multiplication of integers. Such tools also allowed us to find the points of interest used in future attacks. 2) We executed an attack on 32-bit multiplication, for which it was necessary to compute partial correlations (of just 8 bits, for example). Because the correlations are only partial, they are very sensitive to noise contained in the signals; to solve this problem, we introduced an α-parameter. The value of this parameter was varied in some experiments, which considerably improved the effectiveness of our attack. We demonstrated that our proposed attack method is less resource intensive (memory and processing time), even though the results obtained here

focused exclusively on one chip. Consequently, through our detailed analysis, we achieved a substantial increase in the efficiency of side-channel attacks on pairing-based algorithms. We also discussed the countermeasures that can be used to thwart such an attack and considered their potential flaws.

Acknowledgments

This work was supported in part by the EUREKA Catrene programme under contract CAT208 MobiTrust and by a French DGA-MRIS scholarship.

References

1. J. Bajard and N. El Mrabet. Pairing in cryptography: an arithmetic point of view. *Advanced Signal Processing Algorithms, Architectures, and Implementations*, 2007.
2. J.-C. Bajard, L. Imbert, P.-Y. Liardet, and Y. Teglia. Leak Resistant Arithmetic. In *CHES*, pages 62–75. 2004.
3. P. S. L. M. Barreto and M. Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. SAC'05, pages 319–331, 2005.
4. A. Bauer, E. Jaulmes, E. Prouff, and J. Wild. Horizontal and vertical side-channel attacks against secure RSA implementations. *CT-RSA*, pages 1–17, 2013.
5. J. Blömer, P. Günther, and G. Liske. Improved Side Channel Attacks on Pairing Based Cryptography. *COSADE*, pages 154–168, 2013.
6. D. Boneh and M. Franklin. *Identity-Based Encryption from the Weil Pairing*, volume 32. Springer Berlin Heidelberg, 2001.
7. A. D. Booth. A Signed Binary Multiplication Technique, 1951.
8. E. F. Brickell. A Fast Modular Multiplication Algorithm with Application to Two Key Cryptography. In *Advances in Cryptology*, pages 51–60. Springer, 1983.
9. C. Clavier, B. Feix, G. Gagnerot, C. Giraud, M. Roussellet, and V. Verneuil. Rosetta for single trace analysis. In *INDOCRYPT 2012*, pages 140–155. 2012.
10. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Horizontal correlation analysis on exponentiation. *Information and Communications Security*, pages 46–61, 2010.
11. S. Cook. On the minimum computation time of functions. *Transactions of the American Mathematical Society*, 142(23):291–291, 1969.
12. J. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. *Cryptographic Hardware and Embedded Systems*, pages 292 – 302, 1999.
13. J.-S. Coron, P. Kocher, and D. Naccache. Statistics and secret leakage. In *Financial Cryptography*, pages 157–173. Springer, 2000.
14. A. J. Devegili, M. Scott, and R. Dahab. Implementing cryptographic pairings over Barreto-Naehrig curves. In *Pairing 2007*, pages 197–207. Springer, 2007.
15. J.-F. Dhem, M. Joye, and J.-J. Quisquater. Normalisation in diminished-radix modulus transformation. *Electronics Letters*, 33(23):1931, 1997.
16. I. Duursma and H. Lee. Tate Pairing Implementation for Hyperelliptic Curves $y^2 = x^p - x + d$. *Advances in cryptology - AsiaCrypt 2003*, 4:111–123, 2003.
17. N. El Mrabet, G. Di Natale, Flottes, and M. Lise. A Practical Differential Power Analysis Attack Against the Miller Algorithm. *PRIME*, pages 308–311, 2009.
18. S. Ghosh and D. Roychowdhury. Security of prime field pairing cryptoprocessor against differential power attack. pages 16–29. Springer, 2011.

19. B. Gierlichs, K. Lemke-Rust, and C. Paar. Templates vs. stochastic methods. In *CHES*, pages 15–29. Springer, 2006.
20. F. Hess, N. P. Smart, and F. Vercauteren. The Eta pairing revisited. *IEEE Transactions on Information Theory*, 52:4595–4602, 2006.
21. M. Hutter, M. Medwed, D. Hein, and J. Wolkerstorfer. Attacking ECDSA-Enabled RFID devices. *Applied Cryptography and Network Security*, pages 519–534, 2009.
22. M. Joye. Elliptic curves and side-channel analysis. *ST Journal of System Research*, 4(1):17–21, 2003.
23. A. Karatsuba and Y. Ofman. Multiplication of Multidigit Numbers on Automata. In *Soviet physics doklady*, volume 7, page 595, 1963.
24. T. H. Kim, T. Takagi, D.-G. Han, H. W. Kim, and J. Lim. Side Channel Attacks and Countermeasures on Pairing Based Cryptosystems over Binary Fields. *Cryptology and Network Security*, pages 168–181, 2006.
25. N. Koblitz and A. Menezes. Pairing-based cryptography at high security levels. *Cryptography and Coding*, 3796 LNCS:13–36, 2005.
26. P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. *Advances in Cryptology - CRYPTO'99*, pages 1–10, 1999.
27. R. Mayer-Sommer. Smartly analyzing the simplicity and the power of simple power analysis on smartcards. In *CHES*, pages 78–92. Springer, 2000.
28. V. Miller. Use of elliptic curves in cryptography. *Advances in Cryptology - CRYPTO 85 Proceedings*, 218:417–426, 1986.
29. P. L. Montgomery. Modular multiplication without trial division, 1985.
30. E. Oswald. *On side-channel attacks and the application of algorithmic countermeasures*. na, 2003.
31. D. Page and F. Vercauteren. Fault and Side-Channel Attacks on Pairing Based Cryptography. 2004.
32. W. Pan and W. Marnane. A correlation power analysis attack against Tate pairing on FPGA. *Reconfigurable Computing: Architectures, Tools and Applications*, pages 340–349, 2011.
33. G. Perin, L. Imbert, P. Maurine, and L. Torres. Vertical and horizontal correlation attacks on RNS-based exponentiations. *Journal of Cryptographic Engineering*, pages 1–15, 2015.
34. G. Perin, L. Imbert, L. Torres, and P. Maurine. Attacking randomized exponentiations using unsupervised learning. In *COSADE*, pages 144–160. Springer, 2014.
35. J.-J. Quisquater. Presentation at the rump session of Eurocrypt 90. 1990.
36. H. Sato, D. Schepers, and T. Takagi. Exact Analysis of Montgomery Multiplication. INDOCRYPT'04, pages 290–304. Springer, 2004.
37. M. Scott. Computing the Tate pairing. *Topics in Cryptology - CT-RSA 2005*, pages 293–304, 2005.
38. M. Scott. On the efficient implementation of pairing-based protocols. In *Cryptography and Coding*, pages 296–308. Springer, 2011.
39. J. H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, 2nd edition, 2009.
40. A. L. Toom. The complexity of a scheme of functional elements realizing the multiplication of integers. *Soviet Mathematics Doklady*, 3:714–716, 1963.
41. T. Unterluggauer and E. Wenger. Practical Attack on Bilinear Pairings to Disclose the Secrets of Embedded Devices. *ARES*, pages 69–77, 2014.
42. C. Whelan and M. Scott. Side Channel Analysis of Practical Pairing Implementations: Which Path Is More Secure? *VIETCRYPT 2006*, pages 99–114, 2006.