ROSETTA for Single Trace Analysis

Recovery Of Secret Exponent by Triangular Trace Analysis

Christophe Clavier[1], Benoit Feix[1,2], Georges Gagnerot[1,2], Christophe Giraud[3], Mylène Roussellet[2], and Vincent Verneuil[2]

[1] XLIM-CNRS, Université de Limoges, France
firstname.familyname@unilim.fr
[2] INSIDE Secure, Aix-en-Provence, France
firstname-first-letterfamilyname@insidefr.com
[3] Oberthur Technologies, Pessac, France
c.giraud@oberthur.com

Abstract. In most efficient exponentiation implementations, recovering the secret exponent is equivalent to disclosing the sequence of squaring and multiplication operations. Some known attacks on the RSA exponentiation apply this strategy, but cannot be used against classical blinding countermeasures. In this paper, we propose new attacks distinguishing squaring from multiplications using a single side-channel trace. It makes our attacks more robust against blinding countermeasures than previous methods even if both exponent and message are randomized, whatever the quality and length of random masks. We demonstrate the efficiency of our new techniques using simulations in different noise configurations.

Keywords: Exponentiation, Side-Channel Analysis, Collision, Correlation, Blinding.

1 Introduction

Although crypto-systems are proven secure against theoretical cryptanalysis, they can be easily broken if straightforwardly implemented on embedded devices such as smart cards. Indeed, the so-called *Side-Channel Analysis* (SCA) takes advantage of physical interactions between the embedded device and its environment during the crypto-system execution to recover information on the corresponding secret key. Examples of such interactions are the device power consumption [16] or its electromagnetic radiation [10]. SCA can be mainly divided into two kinds: *Simple Side-Channel Analysis* (SSCA) and *Differential Side-Channel Analysis* (DSCA). The first kind aims at recovering information on the secret key by using only one execution of the algorithm whereas DSCA uses several executions of the algorithm and applies statistical analysis to the corresponding measurements to exhibit information on the secret key.

Amongst crypto-systems threatened by SCA, RSA [20] is on the front line since it is the most widely used public key crypto-system, especially in embedded

Published in S. Galbraith and M. Nandi (Eds.), Progress in Cryptology – INDOCRYPT 2012, LNCS vol. 7668, pp. 140-155, Springer, 2012.

environment. Therefore, many researchers have published efficient side-channel attacks and countermeasures specific to RSA over the last decade. Due to the constraints of the embedded environment, countermeasures must not only resist each and every SCA known so far but must also have the smallest impact in terms of performance and memory consumption. Nowadays, the most common countermeasure to prevent SSCA on RSA consists in using an exponentiation algorithm where the sequence of modular operations leaks no information on the secret exponent. Examples of such exponentiation are the square-and-multiply-always [8], the Montgomery ladder [13], the Joye ladder [12], the square-always [7] or the atomic multiply-always exponentiation [4]. The latter is generally favorite due to its very good performance compared to the other non-atomic methods. Regarding DSCA prevention, most common countermeasures consist in blinding the modulus and/or the message, and the exponent [14,8]. Their effect is to randomize the intermediate values manipulated during the exponentiation as well as the sequence of squarings and multiplications. In this paper we denote by *blinded exponentiation* an exponentiation using the atomic implementation presented in [4] where modulus, message and exponent are blinded.

Today blinded exponentiation remain resistant to most SCA techniques. Only the Big Mac attack presented by Walter [24] theoretically threatens this implementation, although no practical result has been ever published. Other attacks introduced later partially threaten this implementation. First, Amiel et al. [1] show how to exploit the average Hamming weight difference between squaring and multiplication operations to recover the secret exponent. Their technique is efficient when the modulus and the message are blinded. However it requires many exponentiation traces using a fixed exponent, so this attack can be thwarted by the randomization of the exponent. To circumvent the blinded exponentiation, they suggested to apply their attack on a single trace but did not try it in practice. Clavier et al. present in [5] the so-called *Horizontal Correlation Analysis*. They apply DSCA using the Pearson correlation coefficient [3] on a single exponentiation side-channel trace. The exponent randomization has no effect against this attack. Modulus and message blinding are efficient only if random masks are large enough (32 bits or more).

Other attacks on the RSA exponentiation are not mentioned in our study as they do not apply to the blinded exponentiation.

In this paper we propose new attacks on the blinded exponentiation which make use of a single execution trace. We achieve this by introducing two new distinguishers — the Euclidean distance and the collision correlation applied to the long-integer multiplication — which allow to efficiently distinguish a squaring from a multiplication operation without the knowledge of the message or the modulus.

Roadmap In Section 2, we recall some basics on RSA implementations on embedded devices. In particular, we describe the attacks presented in [1,5,24] and we show that one of them can be extended using the collision-correlation technique. In Section 3, we present the principle of the so-called *Rosetta* analysis

using two different distinguishers. In Section 4, we put into practice our attack and we demonstrate its efficiency using simulated side-channel traces of long-integer operations using a 32×32-bit multiplier. Moreover, we also compare our technique with previous attacks and show that it is more efficient especially on noisy measurements. We discuss in Section 5 possible methods to counteract Rosetta analysis. Finally, Section 6 concludes this paper.

2 Background

In this section, after presenting some generalities on RSA implementation in the context of embedded environment, we present three of the most efficient side-channel attacks published so far on RSA: the *Big Mac attack* published by Walter at CHES 2001 [24], the one published by Amiel et al. at SAC 2008 [1] and the *Horizontal Correlation Analysis* published at ICICS 2010 by Clavier et al. [5]. Also, we explain how the latter can be extended using a collision-correlation technique.

2.1 RSA Implementation

The standard way of computing an RSA signature S of a message m consists of a modular exponentiation with the private exponent: $S = m^d \bmod N$. The corresponding signature is verified by comparing the message m with the signature S raised to the power of the public exponent: $m \stackrel{?}{=} S^e \bmod N$.

In order to improve its efficiency, the signature is often computed using the Chinese Remainder Theorem (CRT). Let us denote by d_p (resp. d_q) the residue $d \bmod p-1$ (resp. $d \bmod q-1$). To compute the signature, the message is raised to the power of d_p modulo p then to the power of d_q modulo q. The corresponding results S_p and S_q are then combined using Garner's formula [11] to obtain the signature: $S = S_q + q(q^{-1}(S_p - S_q) \bmod p)$.

If used exactly as described above, RSA is subject to multiple attacks from a theoretical point of view. Indeed, it is possible under some assumptions to recover some information on the plaintext from the ciphertext or to forge fake signatures. To ensure its security, RSA must be used according to a protocol which mainly consists in formatting the message. Examples of such protocols are the encryption protocol OAEP and the signature protocol PSS, both of them being proven secure and included in the standard PKCS #1 V2.1 [19]. Note that, as they do not require the knowledge of the exponentiated value, the new attacks described in this paper also apply when either OAEP or PSS scheme is used.

From a practical point of view, the RSA exponentiation is also subject to many attacks if straightforwardly implemented. For instance, SSCA, DSCA or collision analysis can be used to recover the RSA private key. SSCA aims at distinguishing a difference of behavior when an exponent bit is a 0 or a 1.

DSCA allows a deeper analysis than SSCA by exploiting the dependency which exists between side-channel measurements and manipulated data values [2]. To this end, thousands of measurements are generally combined using

a statistical distinguisher to recover the secret exponent value. Nowadays, the most widespread distinguisher is the Pearson linear correlation coefficient [3].

Finally, collision analysis aims at identifying when a value is manipulated twice during the execution of an algorithm.

Algorithm 1 presents the classical atomic exponentiation which is one of the fastest exponentiation algorithms protected against the SPA.

Algorithm 1 Atomic Multiply-Always Exponentiation

Input: $x, n \in \mathbb{N}$, $d = (d_{v-1}d_{v-2}\dots d_0)_2$
Output: $x^d \bmod n$

1: $R_0 \leftarrow 1$
2: $R_1 \leftarrow x$
3: $i \leftarrow v - 1$
4: $k \leftarrow 0$
5: **while** $i \geq 0$ **do**
6: $\quad R_0 \leftarrow R_0 \times R_k \bmod n$
7: $\quad k \leftarrow k \oplus d_i$ [$\oplus$ stands for bitwise X-or]
8: $\quad i \leftarrow i - \neg k$ [$\neg$ stands for bitwise negation]
9: **return** R_0

When correctly implemented, Alg. 1 defeats SSCA since squarings cannot be distinguished from other multiplications on a side-channel trace, as depicted by Fig. 1.

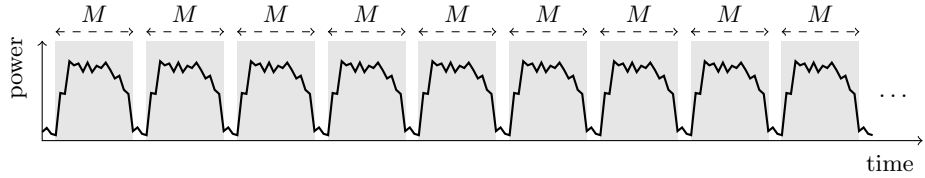

Fig. 1. Atomic multiply-always side-channel leakage

To prevent the implementation of RSA exponentiation from DSCA, the two main countermeasures published so far are based on message and exponent blinding [8,14]. Instead of computing straightforwardly $S = m^d \bmod n$, one rather computes $\tilde{S} = (m + k_0 \cdot n)^{d+k_1 \cdot \varphi(n)} \bmod 2^\lambda \cdot n$ where φ denotes the Euler's totient and k_0 and k_1 are two λ-bit random values, then finally reduce $\tilde{S}$ modulo N to obtain S. Using such a blinding scheme with a large enough λ (32 bits are generally considered as a good compromise between security and cost overhead), the relationship between the side-channel leakages occurring during an exponentiation and the original message and exponent values is hidden to an adversary, therefore circumventing DSCA.

As the modular exponentiation consists of a series of modular multiplications, it relies on the efficiency of the modular multiplication. Many methods have been published so far to improve the efficiency of this crucial operation. Amongst these methods, the most popular are the Montgomery, Knuth, Barrett, Sedlack or Quisquater modular multiplications [17,9]. Most of them have in common that the long-integer multiplication is internally computed by repeatedly calling a smaller multiplier operating on t-bit words. A classic example is given in Alg. 2 which performs the schoolbook long-integer multiplication using a t-bit internal multiplier giving a $2t$-bit result. The decomposition of an integer x in t-bit words is given by $x = (x_{\ell-1}x_{\ell-2}\dots x_0)_b$ with $b = 2^t$ and $\ell = \lfloor \log_b(x) \rfloor + 1$.

Algorithm 2 Schoolbook Long-Integer Multiplication

Input: $x = (x_{\ell-1}x_{\ell-2}\dots x_0)_b, y = (y_{\ell-1}y_{\ell-2}\dots y_0)_b$
Output: $x \times y$
1: **for** $i = 0$ **to** $2\ell - 1$ **do**
2: $\quad z_i \leftarrow 0$
3: **for** $i = 0$ **to** $\ell - 1$ **do**
4: $\quad R_0 \leftarrow 0$
5: $\quad R_1 \leftarrow x_i$
6: $\quad$ **for** $j = 0$ **to** $\ell - 1$ **do**
7: $\quad\quad R_2 \leftarrow y_j$
8: $\quad\quad R_3 \leftarrow z_{i+j}$
9: $\quad\quad (R_5R_4)_b \leftarrow R_3 + R_2 \times R_1 + R_0$
10: $\quad\quad z_{i+j} \leftarrow R_4$
11: $\quad\quad R_0 \leftarrow R_5$
12: $\quad z_{i+\ell} \leftarrow R_5$
13: **return** z

In the rest of this section we recall some previously published attacks on atomic exponentiations which inspired our new technique detailed in Section 3.

2.2 Attacks Background

Distinguishing Squarings from Multiplications in Atomic Exponentiation In [1] Amiel et al. present a specific DSCA aimed at distinguishing squaring from other multiplications in the atomic exponentiation. They observe that the average Hamming weight of the output of a multiplication $x \times y$ has a different distribution whether:

- the operation is a squaring performed using the multiplication routine, i.e. $x = y$, with x uniformly distributed in $\left[0, 2^{\ell t} - 1\right]$;
- or the operation is an actual multiplication, with x and y independent and uniformly distributed in $\left[0, 2^{\ell t} - 1\right]$.

Thus, considering a device with a single long-integer multiplication routine used to perform either $x \times x$ or $x \times y$, a set of N side-channel traces computing multiplications with random operands can be distinguished from a set of N traces

computing squarings, provided that N is sufficiently large to make the two distribution averages separable. This attack can thus target an atomic exponentiation such as Alg. 1 even in the case of message and modulus blinding. Regarding the exponent blinding, authors suggest that their attack should be extended to success on a single trace but do not give evidence of its feasibility. We thus study this point in the following of the paper.

Horizontal Correlation Analysis Correlation analysis on a single atomic exponentiation side-channel trace has been published in [5] where the message is known to the attacker but the exponent is blinded. This attack called *horizontal* correlation analysis requires only one exponentiation trace to recover the full RSA private exponent.

Instead of considering the whole k-th long-integer multiplication side-channel trace T^k as a block, the authors consider each inner side-channel trace segment corresponding to a single-precision multiplication on t-bit words. For instance, if the long-integer multiplication is performed using Alg. 2 on a device provided with a t-bit multiplier, then the trace T^k of the k-th long-integer multiplication $x \times y$ can be split into ℓ^2 trace segments $T^k_{i,j}$, $0 \leq i, j < \ell$, each of them representing a single-precision multiplication $x_i \times y_j$. More precisely, for each word y_j of the multiplicand y, the attacker obtains ℓ trace segments $T^k_{i,j}$, $0 \leq i, j < \ell$, corresponding to a multiplication by y_j. The slicing of T^k into trace segments $T^k_{i,j}$ is illustrated on Fig. 2.

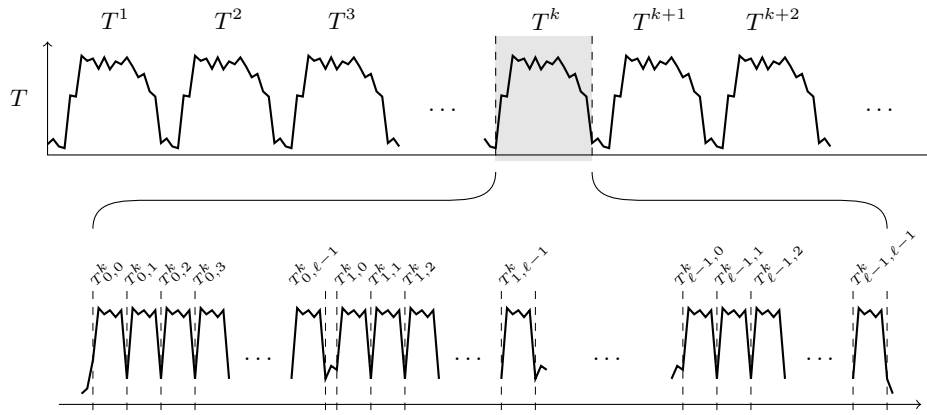

Fig. 2. Horizontal side-channel analysis on exponentiation

In the horizontal correlation analysis the attacker is able to identify whether the k-th long-integer operation T^k is a squaring or a multiplication by computing the correlation factor between the series of Hamming weights of each t-bit word m_j of the message m and the series of corresponding sets of ℓ trace segments $T^k_{i,j}$, $0 \leq i, j < \ell$. This correlation factor is expected to be much smaller when the

long-integer operation is a squaring (i.e. $R_0 \leftarrow R_0 \times R_0$ in Alg. 1) than when it is a multiplication by m (i.e. $R_0 \leftarrow R_0 \times R_1$). The correlation factor can be computed by using the Pearson correlation coefficient $\rho(H, T^k)$ where $H = (H_0, \ldots, H_{\ell-1})$, with $H_j = (\mathrm{HW}(m_j), \ldots, \mathrm{HW}(m_j))$, $\mathrm{HW}(m_j)$ standing for the Hamming weight of m_j and $T^k = (T_0^k, \ldots, T_{\ell-1}^k)$ with $T_j^k = (T_{0,j}^k, \ldots, T_{\ell-1,j}^k)$.

Big Mac Attack Walter's attack needs, as our technique, a single exponentiation side-channel trace to recover the secret exponent. For each long-integer multiplication, the Big Mac attack detects if the operation performed is either $R_0 \times R_0$ or $R_0 \times m$. The multiplications $x_i \times y_j$ — and corresponding trace segments $T_{i,j}^k$ — can be easily identified on the side-channel trace from their specific pattern which is repeated ℓ^2 times in the long-integer multiplication loop. A template side-channel trace is computed (either from the precomputations or from the first squaring operation) to characterize the manipulation of the message during the long-integer multiplication. The Euclidean distance between the template trace and each long-integer multiplication trace T^k is then computed. If it exceeds a threshold then the attack concludes that the operation is a squaring, or a multiplication by m otherwise.

Walter uses the Euclidean distance but we noticed that other distinguisher could be used. In the following section, we extend the Big Mac attack using a collision-correlation technique.

2.3 Big Mac Extension using Collision Correlation

A specific approach for SCA uses information leakages to detect collisions between data manipulated in algorithms. A side-channel collision attacks against a block cipher was first proposed by Schramm et al. in 2003 [22]. More recently Moradi et al. [18] proposed to use a correlation distinguisher to detect collisions in AES. The main advantage of this approach is that it is not necessary to define a leakage model as points of traces are directly correlated with other points of traces. Later, Clavier et al. [6] presented two collision-correlation techniques defeating different first order protected AES implementations. The same year, Witteman et al. [25] applied collision correlation to public key implementation. They describe an efficient attack on RSA using square-and-multiply-always exponentiation and message blinding. All these techniques require many side-channel execution traces. In this section, we extend Walter's Big Mac attack using the collision correlation as distinguisher instead of the Euclidean distance.

We consider a blinded exponentiation and use the fact that the second and third modular operations in an atomic exponentiation are respectively $1 * \tilde{m}$ and $\tilde{m} * \tilde{m}$, where $\tilde{m}$ is the blinded message. The trace of the second long-integer multiplication yields ℓ multiplication segments for each word $\tilde{m}_j$ of the blinded message. Considering the k-th long-integer multiplication, $k > 3$, we compute the correlation factor between the series of ℓ trace segments T_j^2 — each one being composed of the ℓ trace segments $T_{i,j}^2$ involved in the multiplication by $\tilde{m}_j$ — and the series of ℓ trace segments T_j^k. Since the blinded value of the message

does not change during the exponentiation, a high correlation occurs if the k-th long-integer operation is a multiplication, and a low correlation otherwise. Once the sequence of squarings and multiplications is found, the blinded exponent value is straightforwardly recovered. Notice that recovering the blinded value of the secret exponent is not an issue as it can be used to forge signature as well as its non-blinded value.

This attack also works if we use the trace segments T_j^3 of the third long-integer operation instead of the trace segments T_j^2. One can also combine the information provided by the second and third long-integer operations to improve the attack.

Remark As the original Big Mac, this attack also applies to the CRT RSA exponentiation since no information is required on either the message or the modulus. This is of the utmost importance since, to the best of our knowledge, this is the first practical attack on a CRT RSA fully blinded (message, modulus and exponent) atomic exponentiation.

3 ROSETTA: Recovery Of Secret Exponent by Triangular Trace Analysis

3.1 Attack Principle

The long-integer multiplication $\mathrm{LIM}(x, y)$ in base $b = 2^t$ is given by the classical schoolbook formula:

$$x \times y = \sum_{i=0}^{\ell-1} \sum_{j=0}^{\ell-1} x_i y_j b^{i+j}$$

and illustrated, with for instance $\ell = 4$ by the following matrix M:

$$M = \begin{pmatrix} x_0y_0 & x_0y_1 & x_0y_2 & x_0y_3 \\ x_1y_0 & x_1y_1 & x_1y_2 & x_1y_3 \\ x_2y_0 & x_2y_1 & x_2y_2 & x_2y_3 \\ x_3y_0 & x_3y_1 & x_3y_2 & x_3y_3 \end{pmatrix}$$

In the case of a squaring, then $x = y$ and the inner multiplications become:

$$S = \begin{pmatrix} x_0x_0 & x_0x_1 & x_0x_2 & x_0x_3 \\ x_1x_0 & x_1x_1 & x_1x_2 & x_1x_3 \\ x_2x_0 & x_2x_1 & x_2x_2 & x_2x_3 \\ x_3x_0 & x_3x_1 & x_3x_2 & x_3x_3 \end{pmatrix}$$

We consider four observations to design our new attacks, assuming a large enough multiplier size $t \geq 16$:

(Ω_0) $\mathrm{LIM}(x, y)$ s.t. $x = y \Rightarrow \mathrm{Prob}(x_i \times y_i$ are squaring operations$) = 1 \quad \forall i$
(Ω_1) $\mathrm{LIM}(x, y)$ s.t. $x \neq y \Rightarrow \mathrm{Prob}(x_i \times y_i$ are squaring operations$) \approx 0 \quad \forall i$

(Ω_2) LIM(x, y) s.t. $x = y \Rightarrow \text{Prob}(x_i \times y_j = x_j \times y_i) = 1 \quad \forall i \neq j$.
(Ω_3) LIM(x, y) s.t. $x \neq y \Rightarrow \text{Prob}(x_i \times y_j = x_j \times y_i) \approx 0 \quad \forall i \neq j$.

From observations (Ω_0) and (Ω_1) one can apply the attack presented in [1] on a single trace as suggested by the authors. The main drawback is that only ℓ such operations are performed during a LIM which represents a small number of trace segments. It is likely to make the attack inefficient for small modulus lengths (with respect to the multiplier size t).

From observations (Ω_2) and (Ω_3) we notice that collisions between $x_i \times y_j$ and $x_j \times y_i$ for $i \neq j$ can be used to identify squarings from other multiplications. Moreover, LIM(x, y) provides $\ell^2 - \ell$ operations $x_i \times y_j$, $i \neq j$, thus $(\ell^2 - \ell)/2$ couples of potential collisions. This represents a fairly large number of trace segments. The principle of our new attack consists in detecting those internal collisions in a single long-integer operation to determine whether it is a squaring or not. Visually, we split the matrix M into an upper-right and a lower-left triangles of terms, thus we call this technique a *triangle trace analysis*.

We present in the following two techniques to identify these collisions on a single long-integer multiplication trace. The first analysis uses the Euclidean distance distinguisher and the second one relies on a collision-correlation technique.

3.2 Euclidean Distance Distinguisher

We use as distinguisher the Euclidean distance between two sets of points on a trace as Walter [24] in the Big Mac analysis. In order to exploit properties (Ω_2) and (Ω_3) we proceed as follows. For each LIM(x, y) operation we compute the following differential side-channel trace:

$$T_{\text{ED}} = \frac{2}{\ell^2 - \ell} \sum_{0 \leq i < j < \ell} \sqrt{(T_{i,j} - T_{j,i})^2}$$

If the operation performed is a squaring then the single-precision multiplications $x_i \times y_j$ and $x_j \times y_i$ store the same value in the result register (or in the memory) at the end of the operation. The side-channel leakage of the result storage of both operations should thus be similar. On the other hand, if $x \neq y$, products differ and the side-channel leakage should present less similarities. Assuming a side-channel leakage function linear in the Hamming weight of the data manipulated, a squaring should result in $E(T_{\text{ED}}) \approx 0$, whereas we should expect a significantly higher value (about $t/2$ for each of the product halves) in the case of a multiplication.

3.3 Collision-Correlation Distinguisher

We define the two following series of trace segments, where the ordering of couples (i, j) is the same for the two series:

$$\Theta_0 = \{T_{i,j} \text{ s.t. } 0 \leq i < j \leq \ell - 1\}$$

$$\Theta_1 = \{T_{j,i} \text{ s.t. } 0 \leq i < j \leq \ell - 1\}$$

Each set includes $N = (\ell^2 - \ell)/2$ trace segments of base b multiplications.

In order to determine the operation performed by the LIM we compute the Pearson correlation factor between the two series Θ_0 and Θ_1 as described in [6]:

$$\begin{aligned}\hat{\rho}_{\Theta_0,\Theta_1}(t) &= \frac{\mathrm{Cov}(\Theta_0(t), \Theta_1(t))}{\sigma_{\Theta_0(t)}\sigma_{\Theta_1(t)}} \\ &= \frac{N\sum(T_{i,j}(t)T_{j,i}(t)) - \sum T_{i,j}(t)\sum T_{j,i}(t)}{\sqrt{N\sum(T_{i,j}(t))^2 - (\sum T_{i,j}(t))^2}\sqrt{N\sum(T_{j,i}(t))^2 - (\sum T_{j,i}(t))^2}}\end{aligned}$$

where summations are taken over all couples $0 \leq i < j \leq \ell - 1$.

In case of a squaring operation, a much higher correlation value $\hat{\rho}_{\Theta_0,\Theta_1}$ is expected than in case of a multiplication. Computing this correlation value for each LIM operation allows to determine its nature and to recover the sequence of exponent bits.

Remark Contrary to differential analysis on symmetric ciphers, each exponent bit requires to distinguish one hypothesis out of only two, instead of for instance 256 considering a differential attack on AES. Thus fixing a decision threshold is easier when dealing with the exponentiation. This has already been observed when applying DPA or CPA on RSA [2,15] compared to DES or AES.

4 Comparison of the Different Attacks

In order to validate these two techniques, we generated simulated side-channel traces for a classical 32×32-bit multiplier. As generally considered in the literature, we assume a side-channel leakage model linear in the Hamming weight of the manipulated data — here x_i, y_j, and $x_i \times y_j$ — and add a white Gaussian noise of mean $\mu = 0$ and standard deviation σ. We build simulated side-channel traces based on the Hamming weight of the data manipulated in the multiplication operation such that each processed single-precision multiplication generates four leakage points $\mathrm{HW}(x_i)$, $\mathrm{HW}(y_j)$, $\mathrm{HW}(x_i \times y_j \bmod b)$, and $\mathrm{HW}(x_i \times y_j \div b)$, where $\div$ stands for the Euclidean quotient.

Besides validating our two Rosetta variants — the Euclidean distance distinguisher (Rosetta ED) and the collision-correlation one (Rosetta CoCo) — we compare Rosetta with other techniques discussed previously, namely the classical Big Mac, the Big Mac using collision correlation (Big Mac CoCo), and the single trace variant of the Amiel et al. attack presented at SAC 2008.

We proceed in the following way: we randomly select two ℓ-bit integers x and y. Then we generate the side-channel traces of the multiplication $\mathrm{LIM}(x, y)$ and of the squaring $\mathrm{LIM}(x, x)$.

Each different attack is eventually applied and we keep trace of their success or failure to distinguish the squaring from the multiplication. Finally, we estimate

the success rate of each technique by running 1 000 such experiments. These tests are performed for three different noise standard deviation values[1]: from no noise ($\sigma = 0$) to a strong one ($\sigma = 7$).

Characterisation and Threshold A threshold for the attack must be selected for each technique to determine whether the targeted operation is a multiplication or a squaring. Using simulated side-channel traces, it was possible to determine the best threshold value for each technique. Without any knowledge on the component, it is more difficult to fix those threshold values. The attacks could be processed with guess on these thresholds, for instance selecting 0.5 for the collision correlation, but it could not reach optimal efficiency or fail. It is then preferable to determine the best threshold values through a characterization phase of the multiplier, either with an access to an open sample or using the public exponentiation calculation as suggested in [2].

Results We obtain the success rates given in tables 1 ($\sigma = 0$), 2 ($\sigma = 2$) and 3 ($\sigma = 7$) for different key lengths ranging from 512 bits to 2048 bits. Figures 3 and 4 present a graphic comparison of these results for $\sigma = 0$ and $\sigma = 7$.

Technique	512 bits	768 bits	1024 bits	1536 bits	2048 bits
Big Mac [24]	0.986	0.990	0.993	0.994	0.995
SAC 2008 [1]	0.533	0.618	0.734	0.858	0.897
Big Mac CoCo (§2.3)	0.999	1.00	1.00	1.00	1.00
Rosetta ED (§3.2)	1.00	1.00	1.00	1.00	1.00
Rosetta CoCo (§3.3)	1.00	1.00	1.00	1.00	1.00

Table 1. Success rate with a null noise, $\sigma = 0$

Technique	512 bits	768 bits	1024 bits	1536 bits	2048 bits
Big Mac [24]	0.767	0.775	0.807	0.816	0.818
SAC 2008 [1]	0.546	0.629	0.717	0.805	0.855
Big Mac CoCo (§2.3)	0.981	0.998	0.999	1.00	1.00
Rosetta ED (§3.2)	1.00	1.00	1.00	1.00	1.00
Rosetta CoCo (§3.3)	1.00	1.00	1.00	1.00	1.00

Table 2. Success rate with a moderate noise, $\sigma = 2$

[1] Regarding the standard deviation of the noise, a unit corresponds to the side-channel difference related to a one bit difference in the Hamming weight.

Technique	512 bits	768 bits	1024 bits	1536 bits	2048 bits
Big Mac [24]	0.557	0.577	0.621	0.614	0.632
SAC 2008 [1]	0.551	0.577	0.623	0.662	0.702
Big Mac CoCo (§2.3)	0.737	0.855	0.909	0.963	0.981
Rosetta ED (§3.2)	0.711	0.821	0.878	0.953	0.992
Rosetta CoCo (§3.3)	0.685	0.816	0.906	0.992	0.997

Table 3. Success rate with a strong noise, $\sigma = 7$

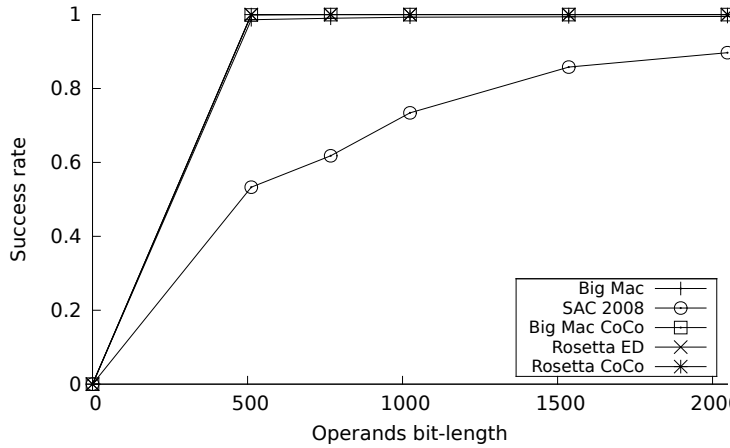

Fig. 3. Success rate of the different attacks with no noise.

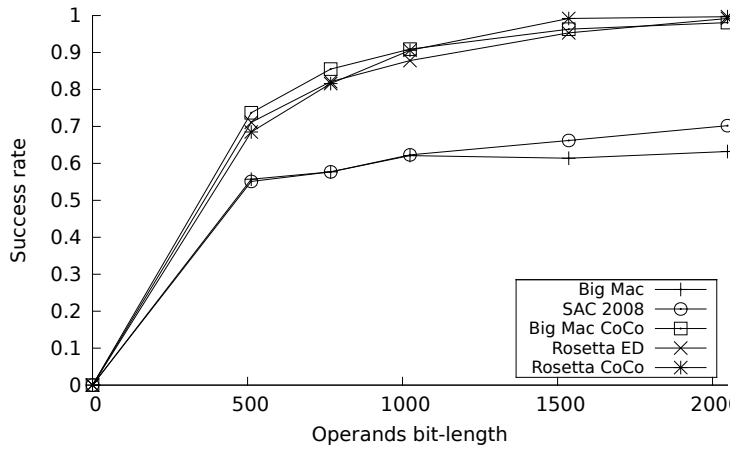

Fig. 4. Success rate of the different attacks with a strong noise, $\sigma = 7$.

Results Interpretation We observe that with no noise (cf. Table 1) all techniques are efficient when applied to large modulus bit lengths (1536 bits or more). For smaller modulus lengths, the SAC 2008 technique is inefficient (probability of success close to 0.5) as expected since the number of useful operations in that case is too small.

In case of a noisy component, we observe that the original Big Mac and the attack from SAC 2008 are not efficient, their probability of success is about 0.5–0.7. Big Mac analysis using collision correlation, and both Rosetta techniques start to be efficient from 1024-bit operands and are very efficient for 1536-bit and 2048-bit operands.

Our study demonstrates that these three last techniques are the most efficient ones and represent a more serious threat for blinded exponentiation than the original Big Mac.

From Partial to Full Exponent Recovery Depending on the component, on the leakage and noise level of the chip, we observe that the success rate of the attack varies and may reveal too few information to recover the whole exponent value. In the case where uncertainty remains on some exponent bits, the attack from Schindler and Itoh [21] may help to reveal them. If necessary, Rosetta analysis can thus be advantageously combined with this technique to completely recover the exponent.

5 Countermeasures

As for the other attacks considered in this paper, both Rosetta techniques we introduced present the following interesting properties: (i) they make use of a single side-channel trace and, (ii) they do not require the knowledge of the message nor of the modulus. As a consequence they are applicable even when the classical set of blinding countermeasures (message, modulus, exponent) is implemented and whatever the size of the random values used.

A first idea to prevent these attacks is to improve the message blinding by randomizing it before each long-integer multiplication, for instance by adding the modulus n or a multiple thereof to the message. At this point, it is worth noticing a specific difference between both Rosetta and other attacks. Rosetta can distinguish a squaring from a multiplication without using any template or previous leakage. This is not the case with the other techniques — except for the single trace variant of the SAC 2008 attack which we demonstrate not to be efficient in the previous section. The consequence is that Rosetta is still applicable even when this improved blinding is implemented.

We recall hereafter three existing countermeasures that we believe to withstand all the techniques presented in this paper.

Shuffled Long-Integer Multiplication In [5], a long integer multiplication algorithm with internal single-precision multiplications randomly permuted is presented. More details are given in [23, Sec. 2.7]. This countermeasure makes

Rosetta analysis virtually infeasible as indices i, j of multiplication $x_i \times y_j$ are not known anymore.

Always True Multiplication This solution consists in ensuring that multiplication operands are always different (or different with high probability). To achieve this objective, before each multiplication $\mathrm{LIM}(x, y)$, both operands x and y are randomized by $x^\star = x + r_1.n$ and $y^\star = y + r_2.n$. If $r_1 \neq r_2$, two equal operands x and y are traded for $x^\star$ and $y^\star$ with $x^\star \neq y^\star$ and the operation $\mathrm{LIM}(x^\star, y^\star)$ is not a squaring.

Square-Always algorithm The square-always algorithm presented in [7] processes any multiplication using two squarings. As for the solution of using multiplications of different terms only, Rosetta does not apply. Regular atomic square always algorithms can be used to prevent SSCA. Exponent blinding countermeasure must be associated with this solution.

6 Conclusion

We present in this study new side-channel methods — the Big Mac using collision correlation and the two Rosetta techniques — allowing to distinguish a squaring from a multiplication when the same long-integer multiplication algorithm is used for both operations. They can be used to recover an RSA secret exponent — both in standard or CRT mode — with a single execution side-channel trace. We compare our new techniques with other single trace side-channel analyses and demonstrate that they are more efficient than previous ones, especially on noisy measurements. We show that classical combination of message, modulus and exponent blindings is not sufficient to counteract our analysis and we suggest more advanced countermeasures. As a conclusion, we quote Colin Walter to recall the very interesting property of these attacks: "The longer the key length, the easier the attacks.".

References

1. F. Amiel, B. Feix, M. Tunstall, C. Whelan, and W. P. Marnane. Distinguishing Multiplications from Squaring Operations. In R. Avanzi, L. Keliher, and F. Sica, editors, *Selected Areas in Cryptography – SAC 2008*, Lecture Notes in Computer Science. Springer, 2008.
2. F. Amiel, B. Feix, and K. Villegas. Power Analysis for Secret Recovering and Reverse Engineering of Public Key Algorithms. In C. M. Adams, A. Miri, and M. J. Wiener, editors, *Selected Areas in Cryptography – SAC 2007*, Lecture Notes in Computer Science, pages 110–125. Springer, 2007.
3. E. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.

4. B. Chevallier-Mames, M. Ciet, and M. Joye. Low-cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity. *IEEE Transactions on Computers*, 53(6):760–768, 2004.
5. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Horizontal Correlation Analysis on Exponentiation. In M. Soriano, S. Qing, and J. López, editors, *Information and Communications Security – ICICS 2010*, volume 6476 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2010.
6. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Improved Collision-Correlation Power Analysis on First Order Protected AES. In B. Preneel and T. Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*. Springer, 2011.
7. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Square always exponentiation. In D. J. Bernstein and S. Chatterjee, editors, *INDOCRYPT 2011 - 12th International Conference on Cryptology in India*, volume 7107 of *Lecture Notes in Computer Science*, pages 40–57. Springer, 2011.
8. J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Ç. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES '99*, volume 1717 of *Lecture Notes in Computer Science*, pages 292–302. Springer, 1999.
9. J.-F. Dhem. *Design of an efficient public-key cryptographic library for RISC-based smart cards.* PhD thesis, Université catholique de Louvain, Louvain, 1998.
10. K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic Analysis: Concrete Results. In Ç. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer, 2001.
11. H. Garner. The Residue Number System. *IRE Transactions on Electronic Computers*, 8(6):140–147, June 1959.
12. M. Joye. Highly regular m-ary powering ladders. In M. J. Jacobson, V. Rijmen, and R. Safavi-Naini, editors, *Selected Areas in Cryptography – SAC 2009*, Lecture Notes in Computer Science, pages 135–147. Springer, 2009.
13. M. Joye and S.-M. Yen. The Montgomery Powering Ladder. In B. Kaliski Jr., Ç. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 291–302. Springer, 2002.
14. P. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.
15. T. Messerges. Using Second-order Power Analysis to Attack DPA Resistant Software. In Ç. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 238–251. Springer, 2000.
16. T. Messerges, E. Dabbish, and R. Sloan. Investigations of Power Analysis Attacks on Smartcards. In *the USENIX Workshop on Smartcard Technology (Smartcard '99)*, pages 151–161, 1999.
17. P. Montgomery. Modular multiplication without trial division. *Math. Comp.*, 44(170):519–521, Apr. 1985.
18. A. Moradi, O. Mischke, and T. Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In S. Mangard and F.-X. Standaert, editors, *Cryptographic Hardware and Embedded Systems – CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 125–139. Springer, 2010.

19. PKCS #1. *RSA Cryptography Specifications Version 2.1*. RSA Laboratories, 2003.
20. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
21. W. Schindler and K. Itoh. Exponent blinding does not always lift (partial) spa resistance to higher-level security. In J. Lopez and G. Tsudik, editors, *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011*, volume 6715 of *Lecture Notes in Computer Science*, pages 73–90, 2011.
22. K. Schramm, T. Wollinger, and C. Paar. A New Class of Collision Attacks and its Application to DES. In T. Johansson, editor, *Fast Software Encryption – FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 2003.
23. V. Verneuil. *Elliptic Curve Cryptography and Security of Embedded Devices.* PhD thesis, Université de Bordeaux, Bordeaux, 2012.
24. C. D. Walter. Sliding Windows Succumbs to Big Mac Attack. In Ç. K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 286–299. Springer, 2001.
25. M. Witteman, J. van Woudenberg, and F. Menarini. Defeating RSA Multiply-Always and Message Blinding Countermeasures. In A. Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2011.