Correlated Extra-Reductions Defeat Blinded Regular Exponentiation

Extended Version

Margaux Dugardin[1,2], Sylvain Guilley[2,3], Jean-Luc Danger[2,3], Zakaria Najm[4], and Olivier Rioul[2,5]

[1] CESTI, Thales Communications & Security, 31000 Toulouse, France.
[2] LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75013 Paris, France.
firstname.lastname@telecom-paristech.fr
[3] Secure-IC SAS, 35510 Cesson-Sévigné, France.
firstname.lastname@secure-ic.com
[4] ST-Microelectronics, 13790 Rousset, France. zakaria.najm@st.com
[5] CMAP, Ecole Polytechnique, Université Paris-Saclay, 91128 Palaiseau, France.
olivier.rioul@polytechnique.edu

Abstract. Walter & Thomson (CT-RSA '01) and Schindler (PKC '02) have shown that extra-reductions allow to break RSA-CRT even with message blinding. Indeed, the extra-reduction probability depends on the type of operation (square, multiply, or multiply with a constant). Regular exponentiation schemes can be regarded as protections since the operation sequence does not depend on the secret.
In this article, we show that there exists a strong negative correlation between extra-reductions of two consecutive operations, provided that the first feeds the second. This allows to mount successful attacks even against blinded asymmetrical computations with a regular exponentiation algorithm, such as Square-and-Multiply Always or Montgomery Ladder. We investigate various attack strategies depending on the context—known or unknown modulus, known or unknown extra-reduction detection probability, etc.—and implement them on two devices: a single core ARM Cortex-M4 and a dual core ARM Cortex M0-M4.

Keywords: Side-channel analysis, Montgomery modular multiplication, Extra-reduction leakage, Message blinding, Regular exponentiation.

1 Introduction

State of the Art of Timing Attacks. Any cryptographic algorithm in an embedded system is vulnerable to side-channel attacks. Timing attacks on the RSA Straightforward Method (RSA-SFM) were pioneered by Kocher [14]. The attack consists in building "templates" whose distributions are compared to that of the response. It is required that the cryptographic parameters be known since the attack is profiled.

Schindler [20] extended timing attacks to RSA with Chinese Remainder Theorem (RSA-CRT) using chosen messages. This attack exploits a conditional extra-reduction at the end of modular multiplications. Schindler and co-authors carried out numerous improvements [1, 2, 21–24] in the case where the exponentiation uses windows or exponent randomization.

Walter and Thompson [25] remarked that even when data is blinded, the distribution of extra-reductions is different for a square and for a multiply. They assumed that side-channel measurements such as power or timing during exponentiation are sufficiently clean to detect the presence or absence of an extra-reduction at each individual operation. Schindler [21] improved this attack by also distinguishing multiplications by a constant from squarings and multiplications by non-fixed parameters.

Today's Solutions. In order to protect the implementation from the above attacks, a first solution consists in exponent randomization on top of message blinding. Such a protection, however, is sensitive to carry leakage [11] and amenable to other attacks like simple power analysis [9] (SPA). A second solution relies on regular exponentiation like Square-and-Multiply-Always (SMA, see Alg. 1.1) or Montgomery Ladder (ML, see Alg. 1.2). Both algorithms consist in a square and a multiply operation in each iteration i, yielding no leakage to SPA.

Algorithm 1.1 Square and Multiply Always Left-to-Right

Input: $m, k = (k_l k_{l-1} \dots k_0)_2, p \quad (k_l = 1)$
Output: $m^k \bmod p$
1: $R_0 \leftarrow 1$
2: $R_1 \leftarrow m$
3: **for** $i = l - 1$ **downto** 0 **do**
4: $\quad R_1 \leftarrow R_1 \times R_1 \bmod p \quad \triangleright S_i$
5: $\quad R_{k_i} \leftarrow R_1 \times m \bmod p \quad \triangleright M_i$
6: **end for**
7: **return** R_1

Algorithm 1.2 Montgomery Ladder Left-to-Right

Input: $m, k = (k_l k_{l-1} \dots k_0)_2, p \quad (k_l = 1)$
Output: $m^k \bmod p$
1: $R_0 \leftarrow m$
2: $R_1 \leftarrow R_0 \times R_0 \bmod p \quad \triangleright$ FS
3: **for** $i = l - 1$ **downto** 0 **do**
4: $\quad R_{\neg k_i} \leftarrow R_0 \times R_1 \bmod p \quad \triangleright M_i$
5: $\quad R_{k_i} \leftarrow R_{k_i} \times R_{k_i} \bmod p \quad \triangleright S_i$
6: **end for**
7: **return** R_0

Contributions of this Paper. We show that despite message blinding and regular exponentiation, it is still possible for an attacker to take advantage of extra-reductions: A new bias is found, namely a strong negative correlation between the extra-reduction of two consecutive operations. As shown in this paper, the bias can be easily leveraged to recover which registers are written to (at line 5 of Alg. 1.1 or at lines 4 and 5 of Alg. 1.2) which eventually leads to retrieve the secret key. The advantages of this method are the following:

- messages are unknown; this captures general situations such as RSA with OAEP or PSS padding and RSA input blinding [13, Sec. 10];
- RSA parameters can be unknown; hence RSA-CRT is also vulnerable;
- all binary exponentiation algorithms are vulnerable, even the regular ones like Square and Multiply Always, Montgomery Ladder, etc.;
- our attack can also be applied to Elliptic Curve Cryptography (ECC).

From a mathematical viewpoint, we also provide a comprehensive framework for studying the joint probabilities of extra-reductions in a sequence of multiplies and squares.

Related Works. The "horizontal/vertical" side-channel attacks against blinded exponentiation described in [8, 12, 28] also use the dependency between the input/output of operands in square and multiply algorithms. Such attacks exploit the *vertical* amplitude of the signal during the time duration. Our work is thus complementary to these ideas since it considers a novel *horizontal* exploitable bias.

Outline. The rest of the paper is organized as follows.Section 2 recalls known biases induced by extra-reductions in modular multiplication algorithms such as the Montgomery modular multiplication. Our contribution starts at Section 3, where the theoretical rationale for the strong negative correlation between extra-reductions of two chained operations is presented. Section 4 shows how this bias can be turned into a key recovery attack. Experimental validations for synthetic and practical traces are in Section 5. Section 6 concludes. Informative appendices contain auxiliary information: improvements and our attack and mitigation techniques are discussed in Sec. A. The proof of the two theorems of the paper is given in Appendix B; the maximum likelihood distinguisher is described in Appendix C. The listing of the source code we exploit is given in Appendix D. The Appendix E focuses on the dependency between operations in Montgomery Ladder exponentiation algorithm.

2 State of the Art of Extra-Reductions Probabilities

This section reviews known results about extra-reductions and their probability distributions. The results can be adapted easily to Barrett reduction or multiplication followed by reduction using the extended Euclid algorithm.

2.1 Montgomery Modular Multiplication: Definitions and Notations

Given two integers a and b, the classical modular multiplication $a \times b \bmod p$ computes the multiplication $a \times b$ followed by the modular reduction by p. Montgomery Modular Multiplication (MMM) transforms a and b into special representations known as their Montgomery forms.

Definition 1 (Montgomery Transformation [16]). *For any prime modulus p, the Montgomery form of $a \in \mathbb{F}_p$ is $\phi(a) = a \times R \bmod p$ for some constant R greater than and co-prime with p.*

In order to ease the computation, R is usually chosen as the smallest power of two greater than p, that is $R = 2^{\lceil \log_2(p) \rceil}$. Using the Montgomery form of integers, modular multiplications used in modular exponentiation algorithms (recall Alg. 1.1 & 1.2) can be carried out using the Montgomery Modular Multiplication (MMM):

Definition 2 (Montgomery Modular Multiplication [16]). *Let $\phi(a)$ and $\phi(b)$ two elements of $\mathbb{F}_p$ in Montgomery form. The MMM of $\phi(a)$ and $\phi(b)$ is $\phi(a) \times \phi(b) \times R^{-1} \bmod p$.*

Algorithm 2.1 below shows that the MMM can be implemented in two steps: (i) compute $D = \phi(a) \times \phi(b)$, then (ii) reduce D using Montgomery reduction which returns $\phi(c)$. In Alg. 2.1, the pair (R^{-1}, v) is such that $RR^{-1} - vp = 1$.

Algorithm 2.1 Montgomery Reduction (Alg. 14.32 of [15])

Input: $D = \phi(a) \times \phi(b)$
Output: $\phi(c) = \phi(a) \times \phi(b) \times R^{-1} \bmod p$
1: $m \leftarrow (D \bmod R) \times v \bmod R$
2: $U \leftarrow (D + m \times p) \div R$ ▷ Invariant: $0 \leq U < 2p$
3: **if** $U \geq p$ **then**
4: $\quad C \leftarrow U - p$ ▷ Extra-reduction
5: **else**
6: $\quad C \leftarrow U$
7: **end if**
8: **return** C

Definition 3 (Extra-Reduction). *In Alg. 2.1, when the intermediate value U is greater than p, a subtraction named eXtra-reduction occurs so as to have a result C of the Montgomery multiplication between 0 and $p - 1$. We set $X = 1$ in the presence of the eXtra-reduction, and $X = 0$ in its absence.*

Most software implementations of modular arithmetic for large numbers (such as OpenSSL and mbedTLS) use the MMM, where there is a final conditional extra-reduction. In mbedTLS, this extra-reduction is compensated. However, as shown below in Sec. 5.2, an attacker is still able in practice to detect using some side-channel which branch has been used (either line 4 or 6 of Alg. 2.1).

2.2 A Bias to Differentiate a Multiply from a Square

Proposition 1 (Probability of Extra-Reduction in a Multiply and a Square Operation [20, Lemma 1]). *Assuming uniform distribution of operands, the probabilities of an extra-reduction in a multiply X_{M_i} and in a square X_{S_i} at iteration i are*

$$\mathbb{P}(X_{M_i} = 1) = \mathbb{E}(X_{M_i}) = \frac{p}{4R} \qquad \textit{and} \qquad \mathbb{P}(X_{S_i} = 1) = \mathbb{E}(X_{S_i}) = \frac{p}{3R}. \quad (1)$$

We note that extra-reductions are 33% more likely when the operation is a square than when it is a multiply, irrespective of the ratio $\frac{p}{R} \in]\frac{1}{2}, 1[$. This allows one to break unprotected exponentiation algorithms.

We compare the theory (Prop. 1) and experimental occurrences of extra-reductions on one million multiplications and squares, using these moduli[1]:

1. `RSA-1024-p`: $p =$ `0xcd083568d2d46c44c40c1fa0101af2155e59c70b08423112af0c1202514bba5 \ 210765e29ff13036f56c7495894d80cf8c3baee2839bacbb0b86f6a2965f60db1`.
2. `RSA-1024-q`: $p =$ `0xca0eeea5e710e8e9811a6b846399420e3ae4a4c16647e426ddf8bbbcb11cd3f \ 35ce2e4b6bcad07ae2c0ec2ecbfcc601b207cdd77b5673e16382b1130bf465261`.
3. `RSA-1024-n`: $p =$ `RSA-1024-p` $\times$ `RSA-1024-q` (not a prime, but results apply).
4. `P-256` [17]: $p =$ `0xffffffff00000001000000000000000000000000ffffffffffffffffffffffff`.
5. `brainpoolP256r1` [7]: $p =$ `0xa9fb57dba1eea9bc3e660a909d838d726e3bf623d52620282013481d1f6e5377`.

Cryptographic parameters			$\mathbb{P}(X_{M_i} = 1)$		$\mathbb{P}(X_{S_i} = 1)$	
p	R	Ratio p/R	theory	experiment	theory	experiment
`RSA-1024-p`	2^{512}	0.800907	0.200227	0.200241	0.266969	0.266893
`RSA-1024-q`	2^{512}	0.789290	0.197323	0.198207	0.263097	0.263774
`RSA-1024-n`	2^{1024}	0.632147	0.158036	0.157875	0.210715	0.209865
`P-256`	2^{256}	1.000000	0.250000	0.250049	0.333333	0.333523
`brainpoolP256r1`	2^{256}	0.663991	0.165998	0.165846	0.221330	0.221134

Table 1. Extra-reduction probability for multiplications (M_i) and squares (S_i)

As shown in Tab. 1, experimental probabilities are very close to the theory. Theoretical results rely on some assumptions (cleanly stated in Appendix B.1), which justifies that this validation is useful.

3 A Bias to Test the Dependency of Operations

3.1 Principle of Correlated Extra-Reductions

In regular exponentiation algorithms, differentiating a multiply from a square does not allow SPA to distinguish the value of the exponent bits. Indeed, at every iteration i ($l - 1 \geq i > 0$ where i is decremented after each iteration),

[1] Notice that the moduli do not need to be prime numbers. Consider the case of RSA implemented without CRT (see e.g., `RSA-1024-n`). It is particularly relevant to our scenario. Indeed, when the factorization $n = pq$ is unknown, the exponent randomization $d \leftarrow d + r\varphi(n)$, where r is a small (e.g., 64 bit) uniformly distributed random number [13, section 10, page 9], is impossible since $\varphi(n) = (p-1)(q-1) = n + 1 - p - q$ cannot be computed without the knowledge of the two prime factors p and q. Thus, regular exponentiation is a very suitable protection, since more efficient than alternative approaches, such as signature verification or exponent splitting.

multiply and square operations are carried out unconditionally. However, the input value of each operation depends on the current exponent bit value k_i. Figure 1 illustrates the dependence or independence between the input/output values of multiplication M_i and the input value of the following square S_{i-1} as a function of the bit value k_i during the SMA algorithm (Alg. 1.1). Intuitively,

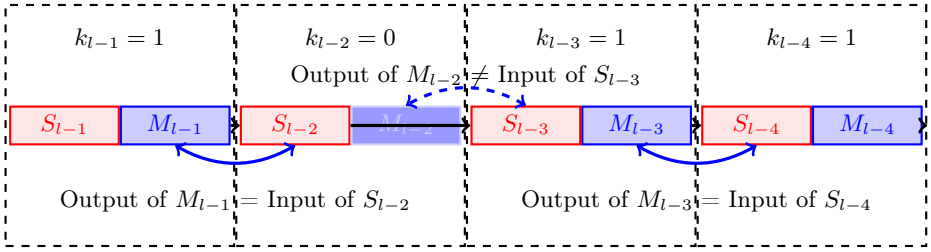


Fig. 1. Comparison between the output value of multiplication with the input of the following square in the Square-and-Multiply-Always exponentiation algorithm (Alg. 1.1).

when the output of M_i is equal to the input of S_{i-1}, we can expect that the extra-reductions in both operations are strongly correlated.

For the ML algorithm (Alg. 1.2), an illustration is provided in Fig. 11 from App. E. The M_i and S_{i-1} operations depends directly on the two consecutive key bit values k_i and k_{i-1}. If the bit value k_{i-1} and its previous bit value k_i are different then the output of multiplication M_i and the input of square S_{i-1} are equal and yield strongly correlated extra-reductions; in the opposite case they yield uncorrelated extra-reductions.

Definition 4 (Guess Notation). *Let $\mathcal{G}_i$ be the "guess' Boolean random variable defined to be* `True` *(T) if the output of an operation at iteration i is equal to the input of the next operation at iteration $i-1$, and* `False` *(F) otherwise.*

Also let X_{M_i} be a random variable corresponding to the eXtra-reduction of the MMM multiplication at iteration i and $X_{S_{i-1}}$ be a random variable corresponding to the eXtra-reduction during the MMM square at iteration $(i-1)$.

Then $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$ is their joint probability when the output value of the multiplication is equal to the input value of the square, and $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$ is their joint probability when the output value of the multiplication is not equal to the input value of the square.

The guess value $\mathcal{G}_i$ is linked to the key value depending on the regular exponentiation algorithm. For SMA and for a bit k_i, an attacker is able to estimate the probabilities $\hat{\mathbb{P}}(X_{M_i}, X_{S_{i-1}})$. This probability can be used to find the bit k_i as illustrated in Fig. 1 and explained in Section 4 below. For ML, $\mathcal{G}_i$ depends on two consecutive key bits as explained also in Section 4.

We have estimated the joint probabilities $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i)$ using 1.000.000 random values for both SMA and ML algorithms and the example `RSA-1024-p` defined in Sec. 2.2.The values of the obtained probabilities are shown in Tab. 2.

$(x_{M_i}, x_{S_{i-1}})$	(0,0)	(1,0)	(0,1)	(1,1)
$\mathbb{P}(x_{M_i}, x_{S_{i-1}}\|\mathcal{G}_i = T)$	0.541575	0.191615	0.258276	0.008532
$\mathbb{P}(x_{M_i}, x_{S_{i-1}}\|\mathcal{G}_i = F)$ for SMA	0.612756	0.120158	0.186803	0.080281
$\mathbb{P}(x_{M_i}, x_{S_{i-1}}\|\mathcal{G}_i = F)$ for ML	0.586105	0.147246	0.213521	0.053128

Table 2. Example of probabilities of eXtra-reduction X_{M_i} of multiply operation and $X_{S_{i-1}}$ of square operation knowing the Boolean value $\mathcal{G}_i$ for `RSA-1024-p`. The first line (correct guess) is applicable for both SMA and ML.

It is important to notice that for each $(x_{M_i}, x_{S_{i-1}}) \in \{0,1\}^2$, the conditional joint probabilities are distinct: $\mathbb{P}(X_{M_i} = x_{M_i}, X_{S_{i-1}} = x_{S_{i-1}}|\mathcal{G}_i = F) \neq \mathbb{P}(X_{M_i} = x_{M_i}, X_{S_{i-1}} = x_{S_{i-1}}|\mathcal{G}_i = T)$. Also for $\mathcal{G}_i = F$ in ML, it can be observed that $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i) = \frac{p}{4R} \times \frac{p}{3R} = \mathbb{P}(X_{M_i}) \times \mathbb{P}(X_{S_{i-1}})$, which is consistent with the fact the two operations X_{M_i} and $X_{S_{i-1}}$ should be independent since they are completely unrelated.

It should be emphasized that the leakage identified in Tab. 2 is fairly large, since the Pearson correlations ρ of the two randoms variables are[2]:

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) \approx -0.2535, \tag{2}$$

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F) \approx +0.1510 \text{ in SMA}, \tag{3}$$

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F) \approx -0.0017 \text{ in ML}. \tag{4}$$

To the best of our knowledge, such correlations have not been observed previously. A few observations are in order:

- when a square follows a multiply, and if there has been an extra-reduction in the multiplication, the result should be short, hence there is less chance for an extra-reduction to occur in the following square. This accounts for the negative correlation $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$;
- from Fig. 1 iteration $i = l - 2$ where $k_i = 0$, we can see that one input of the multiplication M_i equals the input of the following squaring S_{i-1}. Since a square and a multiplication share a common operand, provided it is sufficiently large, both operations are likely to have an extra-reduction at the same time, which accounts for the positive correlation $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$ for SMA;
- when a square and a multiply handle independent data, the extra-reductions are clearly also independent of each other, which explains the small value of $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$ for ML.

[2] $\rho(X_{M_i}, X_{S_{i-1}}) = \frac{\mathbb{C}ov(X_{M_i}, X_{S_{i-1}})}{\sigma_{X_{M_i}} \sigma_{X_{S_{i-1}}}} = \frac{\mathbb{P}(X_{M_i}=1, X_{S_{i-1}}=1) - (\mathbb{P}(X_{M_i}=1) \times \mathbb{P}(X_{S_{i-1}}=1))}{\sqrt{\mathbb{P}(X_{M_i}=1)(1-\mathbb{P}(X_{M_i}=1))}\sqrt{\mathbb{P}(X_{S_{i-1}}=1)(1-\mathbb{P}(X_{S_{i-1}}=1))}}$.

As explained next, when extra-reductions can be detected reliably[3], the data-flow can be analyzed accurately thereby defeating regular exponentiation protections.

3.2 Methodology to Analyze the Bias

In order to estimate the probability $\mathbb{P}(X_{M_i}, X_{S_{i-1}} | \mathcal{G}_i)$, we first determine the distribution of the output value after one MMM (following the method described by Sato et al. [19]) and then compute the joint probability for each case.

Let A, B be two independent random variables uniformly distributed in $[0, p[$ (represented in Montgomery form); let C be equal to the MMM product of A and B and U corresponds to the MMM product of A and B before eXtra-reduction (if any). Variables C and U coincide with that of Alg. 2.1. As a matter of fact, an attacker cannot observe values, only extra-reductions which occur during Montgomery reduction (at line 4 of Alg. 2.1). We use notations $\mathbb{P}$ for probabilities and f for probability density functions (p.d.f.'s).

Fig. 2 shows histograms for C and U obtained from one million simulations; the binning consists of 100 bins of the interval $[0, 2p[$. It can be observed that

- the p.d.f. of C is uniform on $[0, p[$;
- the p.d.f. of U is a piecewise continuous function composed of a strictly increasing part, a constant part and a strictly decreasing part;
- the two conditional p.d.f.'s of C knowing $X_{M_i} \in \{0, 1\}$ (resp. $X_{S_i} \in \{0, 1\}$) are not uniform;
- for $c \in [0, p[$, one has $f(C = c) = f(U = c) + f(U = c + p)$ by definition of U;
- the maximum value of U is $p + p^2/R$, which is strictly smaller than $2p$.

Recall that we use the Montgomery reduction described in Alg. 2.1, where the reduction modulo p is carried out after every multiplication. This is also the case in [20, 21], but *not* in [24, 25] where the multiplicands lie in $[0, R[$. To complement those works, we now derive a closed-form expression of the output distribution of the Montgomery multiplication product and square (not found in [20, 21]).

3.3 Mathematical Derivations

This subsection provides a mathematical justification of the biases observed in Tab. 2. In particular, it shows that such biases hold for all values of p and $R = 2^{\lceil \log_2(p) \rceil}$. Our closed-form expressions are derived as limits in distribution when $p \to +\infty$ that we shall write as approximations.

[3] In particular, the global timing of the algorithm is insufficient for the attack to succeed, because the attacker must be able to relate the extra-reductions to a target operation.

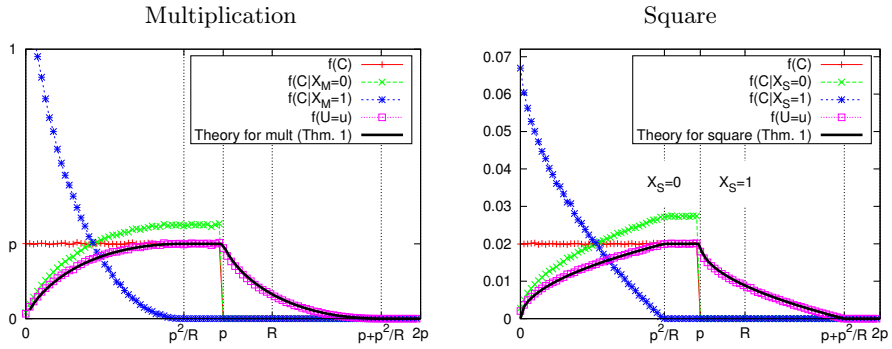


Fig. 2. Distribution of the output value of Montgomery multiplication (*left*) and square (*right*) for `RSA-1024-p`.

Theorem 1 (P.d.f. of MMM Before Extra-Reduction). *Asymptotically when modulus p is large, the result of a Montgomery multiplication before the final extra-reduction (at line 2 of Alg. 2.1) have piecewise p.d.f. given by*

$$f_U(u) = \begin{cases} \frac{Ru}{p^3}\left(1 - \ln(\frac{Ru}{p^2})\right) & \text{if } 0 \le u \le \frac{p^2}{R}; \\ \frac{1}{p} & \text{if } \frac{p^2}{R} \le u \le p; \\ \frac{1}{p} - \frac{R(u-p)}{p^3}\left(1 - \ln(\frac{R(u-p)}{p^2})\right) & \text{if } p \le u \le p + \frac{p^2}{R}; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The corresponding p.d.f. for the square is also in four pieces with the same intervals for u, and differs only from the multiplication in that it is equal to $\sqrt{Ru}/p^2$ when $0 \le u \le \frac{p^2}{R}$, and $1/p - \sqrt{R(u-p)}/p^2$ when $p \le u \le p + \frac{p^2}{R}$.

Proof. See proof in Appendix B.2. □

The theoretical values of Theorem 1 nicely superimpose with experimentally estimated p.d.f.'s as shown in Fig. 2.

Theorem 2 (Joint Probability of Extra-Reduction in Multiplication Followed by a Square). *The following joint probabilities do not depend on the iteration index i, where $l - 1 \ge i > 0$.*

When $\mathcal{G}_i = T$:

$\mathbb{P}(x_{M_i}, x_{S_{i-1}})$	$x_{S_{i-1}} = 0$	$x_{S_{i-1}} = 1$
$x_{M_i} = 0$	$1 - \frac{7}{12}\frac{p}{R} + \frac{1}{48}\left(\frac{p}{R}\right)^4$	$\frac{p}{3R} - \frac{1}{48}\left(\frac{p}{R}\right)^4$
$x_{M_i} = 1$	$\frac{p}{4R} - \frac{1}{48}\left(\frac{p}{R}\right)^4$	$\frac{1}{48}\left(\frac{p}{R}\right)^4$

When $\mathcal{G}_i = F$ in SMA:

$\mathbb{P}(x_{M_i}, x_{S_{i-1}})$	$x_{S_{i-1}} = 0$	$x_{S_{i-1}} = 1$
$x_{M_i} = 0$	$1 - \frac{7}{12}\frac{p}{R} + \frac{1}{8}\left(\frac{p}{R}\right)^2$	$\frac{p}{3R} - \frac{1}{8}\left(\frac{p}{R}\right)^2$
$x_{M_i} = 1$	$\frac{p}{4R} - \frac{1}{8}\left(\frac{p}{R}\right)^2$	$\frac{1}{8}\left(\frac{p}{R}\right)^2$

When $\mathcal{G}_i = F$ in ML:

$\mathbb{P}(x_{M_i}, x_{S_{i-1}})$	$x_{S_{i-1}} = 0$	$x_{S_{i-1}} = 1$
$x_{M_i} = 0$	$1 - \frac{7}{12}\frac{p}{R} + \frac{1}{12}\left(\frac{p}{R}\right)^2$	$\frac{p}{3R} - \frac{1}{12}\left(\frac{p}{R}\right)^2$
$x_{M_i} = 1$	$\frac{p}{4R} - \frac{1}{12}\left(\frac{p}{R}\right)^2$	$\frac{1}{12}\left(\frac{p}{R}\right)^2$

Proof. See proof in Appendix B.3. □

It can be easily checked that Theorem 2 accurately matches experimental probability estimations given in Tab. 2.

Corollary 1. *The corresponding correlation coefficients are*

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) = \frac{\frac{p^4}{48R^4} - \frac{p^2}{12R^2}}{\sqrt{\frac{p}{4R}\left(1 - \frac{p}{4R}\right)\frac{p}{3R}\left(1 - \frac{p}{3R}\right)}},$$

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F) = \frac{\frac{p^2}{24R^2}}{\sqrt{\frac{p}{4R}\left(1 - \frac{p}{4R}\right)\frac{p}{3R}\left(1 - \frac{p}{3R}\right)}} \text{ in SMA,}$$

$$\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F) = 0 \text{ in ML.}$$

Proof. Apply Pearson's correlation definition on the results of Theorem 2. □

When the guess is correct, $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$ is negative and increasingly negative as p/R increases, where

$$-\frac{3}{16}\sqrt{\frac{5}{7}} \approx -0.158 \quad \leq \quad \rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) \quad \leq \quad -\frac{3}{4\sqrt{6}} \approx -0.306.$$

When the guess is incorrect, either the correlation is null (in the case of ML), or it is positive and increasing with p/R, where for $1/2 \leq p/R \leq 1$,

$$\frac{1}{2\sqrt{5\times 7}} \approx 0.085 \quad \leq \quad \rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F) \quad \leq \quad \frac{1}{2\sqrt{6}} \approx 0.204.$$

The variations of the correlation coefficients between X_{M_i} and $X_{S_{i-1}}$ in the three scenarios of Corollary 1 are plotted in Fig. 3.

Fig. 3 shows that the correlation difference between guesses `True/False` is greater for the SMA algorithm than for the ML algorithm. Thus our attack on SMA should outperform that on ML. Also notice that the larger the ratio p/R, the larger the correlation difference; hence, we expect `P-256` to be easier to break than `brainpoolP256r1` with our attack.

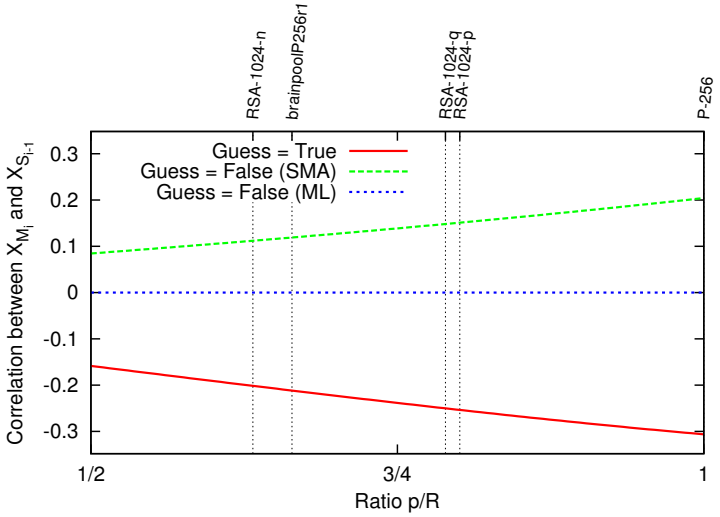


Fig. 3. Pearson's correlation between X_{M_i} and $X_{S_{i-1}}$.

4 Exploiting the Bias Using our Attack

The difference between the two Pearson correlations according to the guess value $\mathcal{G}_i$ (Corollary 1) allows us to test whether some data produced by an operation is fed into the next operation. The bit value k_i can be estimated using the Pearson correlation ρ as a distinguisher, a threshold $\mathcal{T}$ depending of the knowledge of the attacker and a decision function denoted by $\mathcal{F}_{\mathcal{ALG}}$ which depends of the regular exponentiation algorithm and the used distinguisher.

Attacker's Method. An attacker calls Q times the cryptographic operation with a static key k and measures the corresponding side-channel trace. For each trace $q \in \{1, \ldots, Q\}$, $(l-1)$ pairs of extra-reductions $(x^q_{M_i}, x^q_{S_{i-1}})_{l-1 \geq i > 0}$ are captured. The complete acquisition campaign is denoted $(\underline{x_{M_i}}, \underline{x_{S_{i-1}}})$, and is a matrix of size $Q \times (l-1)$ pairs of bits. Notice that neither the input nor the output of the cryptographic algorithm is required. For all $i \in \{l-1, \ldots, 1\}$ and $q \in \{1, \ldots, Q\}$, $x^q_{M_i}$ is equal to 1 (resp. 0) if the eXtra-reduction is present (resp. missing) during the multiplication M_i for query q. Similarly, $x^q_{S_{i-1}}$ is equal to 1 (resp. 0) if the eXtra-reduction is present (resp. missing) during the square S_{i-1} for query q. For each pair of random variable $(x_{M_i}, x_{S_{i-1}}) \in \{0,1\}^2$, the attacker first computes the estimated probability $\hat{\mathbb{P}}(X_{M_i} = x_{M_i}, X_{S_{i-1}} = x_{S_{i-1}})$, using:

$$\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}}) = \frac{1}{Q} \sum_{q=1}^{Q} \mathbb{1}_{(x^q_{M_i} = x_{M_i}) \wedge (x^q_{S_{i-1}} = x_{S_{i-1}})}. \tag{6}$$

The attacker then computes the Pearson correlation[4] $\hat{\rho}(X_{M_i}, X_{S_{i-1}})$ using the estimated probability $\hat{\mathbb{P}}(X_{M_i}, X_{S_{i-1}})$. Finally, she estimates the exponent bit k_i with her knowledge corresponding to threshold $\mathcal{T}$ and decision function $\mathcal{F}_{\mathcal{ALG}}$.

Attacker's Knowledge. In public key cryptography, the attacker wants to recover the private exponent in RSA or the private scalar in ECC. In our attacks, we assume these secret values are static, as for instance in RSA-CRT decryption or static Diffie-Hellman key agreement protocol.

- In RSA-SFM and ECC, the attacker knows the parameters p and R defined in Sec. 2.1. In RSA-SFM, p is equal to the public modulus n_{RSA}. In ECC, p equals the characteristic of the finite field over which the elliptic curve is defined. The attacker can compute the Pearson correlations $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$ and $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$ using corollary 1. The threshold for the successful attack is defined by:

$$\mathcal{T} = \frac{\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) + \rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)}{2}. \tag{7}$$

- In RSA-CRT, the attacker does not know the parameters p and R defined in Sec. 2.1, because the prime factors p_{RSA} and q_{RSA} are secret parameters. Hence the determination of the probabilities by theory or simulation are impossible[5]. However, using the Q measurements $(\underline{x_{M_i}}, \underline{x_{S_{i-1}}})$, the attacker is able to determine the mean estimated probability $\hat{\mathbb{E}}_i\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}})$ for all $((x_{M_i}, x_{S_{i-1}}) \in \{0,1\}^2$ by[6]:

$$\hat{\mathbb{E}}_i\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}}) = \frac{\sum_{i=1}^{l-1} \hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}})}{l-1}. \tag{8}$$

  The attacker then computes the mean estimated Pearson correlations using the mean estimated probability (8), and the threshold for the successful attack is defined by:

$$\mathcal{T} = \frac{\hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{M_i} = 1, X_{S_{i-1}} = 1) - \hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{M_i} = 1) \times \hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{S_{i-1}} = 1)}{\sqrt{\hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{M_i} = 1)\hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{M_i} = 0)}\sqrt{\hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{S_{i-1}} = 1)\hat{\mathbb{E}}_i\hat{\mathbb{P}}(X_{S_{i-1}} = 0)}}. \tag{9}$$

In fact, the threshold value $\mathcal{T}$ computed in (7) or (9) does not depend on i. The indication of index i was kept as a reminder that the multiplication M_i is done in the iteration which precedes that of the square S_{i-1}.

---

[4] $\hat{\rho}(X_{M_i}, X_{S_{i-1}}) = \frac{\hat{\mathbb{C}ov}(X_{M_i}, X_{S_{i-1}})}{\hat{\sigma}_{X_{M_i}}\hat{\sigma}_{X_{S_{i-1}}}} = \frac{\hat{\mathbb{P}}(X_{M_i}=1, X_{S_{i-1}}=1) - (\hat{\mathbb{P}}(X_{M_i}=1) \times \hat{\mathbb{P}}(X_{S_{i-1}}=1))}{\sqrt{\hat{\mathbb{P}}(X_{M_i}=1)(1-\hat{\mathbb{P}}(X_{M_i}=1))}\sqrt{\hat{\mathbb{P}}(X_{S_{i-1}}=1)(1-\hat{\mathbb{P}}(X_{S_{i-1}}=1))}}$.

[5] To be exact, as underlined by Werner Schindler in [21, Sec. 10, page 277], the ratio p/R can be estimated using the empirical probability for an extra reduction in a squaring, which is equal to $p/3R$ (recall Proposition 1).

[6] Notice that in some cases, e.g. if the key bits happen not to be balanced, $\hat{\mathbb{E}}_i\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}})$ can be estimated in a less biased way using $\max_{i=1}^{l-1}\{\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}})\} - \min_{i=1}^{l-1}\{\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}})\}$.

Decision Function. The decision function depending of the regular algorithm and the used distinguisher ρ is denoted as $\mathcal{F}_{\mathcal{ALG}}$. We detail this function for the SMA (Alg. 1.1) and ML (Alg. 1.2) algorithms.

- In the SMA algorithm, the scalar bit k_i decides whether the output of M_i is the input of S_{i-1} or not (see Fig. 1). If the bit value k_i equals 1, then the square S_{i-1} depends on M_i ($\mathcal{G}_i = T$), otherwise the output value of M_i is different from the input value of S_{i-1} ($\mathcal{G}_i = F$). Using the Sec. 3, we see that $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) < \rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$, so the decision function $\mathcal{F}_{SMA}$ is defined by:

$$\hat{k}_i = \mathcal{F}_{SMA}(\rho, \mathcal{T}) = \begin{cases} 0 & \text{if } \hat{\rho}(X_{M_i}, X_{S_{i-1}}) \geq \mathcal{T}, \\ 1 & \text{otherwise.} \end{cases} \tag{10}$$

- For the Montgomery Ladder (ML) algorithm, the M_i and S_{i-1} operations do not depend directly on the key bit value k_i. The dependence comes from the bit value k_{i-1} and the previous bit value k_i. If the two bits value k_{i-1} and k_i are different then the output of multiplication M_i and the input of square S_{i-1} are equal ($\mathcal{G}_i = T$), otherwise these output/input are different ($\mathcal{G}_i = F$). Using Sec. 3, we see that $\rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T) < \rho(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$, so the decision function $\mathcal{F}_{ML}$ using the previously estimated bit $\hat{k}_{i-1}$ is defined for each i ($l - 1 > i \geq 1$) by:

$$\hat{k}_i = \mathcal{F}_{ML}(\hat{k}_{i-1}, \rho, \mathcal{T}) = \begin{cases} \hat{k}_{i-1} & \text{if } \hat{\rho}(X_{M_i}, X_{S_{i-1}}) \geq \mathcal{T}, \\ \neg\hat{k}_{i-1} & \text{otherwise.} \end{cases} \tag{11}$$

  Regarding the second most significant bit k_{l-1} of the exponent, either both values $k_{l-1} = 0$ and $k_{l-1} = 1$ are tested to recover the full secret key, or our attack can be applied between the first square FS (defined at line 2 of Alg. 1.2) and the square S_{l-1} (line 5 of Alg. 1.2).

Summary of the Attack. To estimate the exponent k by $\hat{k}$, we define two attacks:

- The attack named "ρ-attack-Hard", knowing the values of $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$ and $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$, using the threshold $\mathcal{T}$ computed by (7).
- The attack named "ρ-attack-Soft", when the theoretical value $\mathbb{P}(X_{M_i}, X_{S_{i-1}}|\mathcal{G}_i)$ is unknown. It uses the estimated probability $\hat{\mathbb{P}}(X_{M_i}, X_{S_{i-1}})$ to compute the threshold $\mathcal{T}$ by (9).

Algorithm 4.1 describes the attack to recover a full key. Lines 1-12 correspond to the computation of the estimated probabilities for each bit k_i defined by (6). Line 13 is the computation of the threshold: if the attack is ρ-attack-Hard the attacker uses (7), otherwise the attack is ρ-attack-Soft and she uses (9). The lines 14-16 compute the full estimated key using the decision function $\mathcal{F}_{\mathcal{ALG}}$, defined by the equations (10) or (11).

Algorithm 4.1 ρ-attack

Input: $(\underline{x_{M_i}}, \underline{x_{S_{i-1}}})$, a set of Q pairs of $(l-1)$ bits

Output: An estimation $\hat{k} \in \{0,1\}^{l-1}$ of the secret exponent

for $i = l-1$ **downto** 1 **do**
  for $(x_{M_i}, x_{S_{i-1}}) \in \{0,1\}^2$ **do**
    $\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}}) \leftarrow 0$
  end for
  for $q = 1$ **to** Q **do**
    $\hat{\mathbb{P}}(X_{M_i} = x^q_{M_i}, X_{S_{i-1}} = x^q_{S_{i-1}}) \leftarrow \hat{\mathbb{P}}(X_{M_i} = x^q_{M_i}, X_{S_{i-1}} = x^q_{S_{i-1}}) + 1$
  end for
  for $(x_{M_i}, x_{S_{i-1}}) \in \{0,1\}^2$ **do**
    $\hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}}) \leftarrow \hat{\mathbb{P}}(x_{M_i}, x_{S_{i-1}}) \,/\, Q$ ▷ Normalization
  end for
  Compute $\hat{\rho}(X_{M_i}, X_{S_{i-1}})$ using $\hat{\mathbb{P}}(X_{M_i}, X_{S_{i-1}})$
end for
Compute $\mathcal{T}$ depending on the attacker's knowledge
for $i = l-1$ **downto** 1 **do**
  $\hat{k}_i \leftarrow \mathcal{F}_{\mathcal{ALG}}\left(\hat{\rho}(X_{M_i}, X_{S_{i-1}}), \mathcal{T}\right)$ ▷ Threshold
end for
return $\hat{k}$

5 Experimental Results

In the first part of this section, we detail a simulated attack which exploits the bias (explained in Corollary 1) to determine the number of queries necessary for the success of the attack. Then, we detail the side-channel part (*local timing analysis* using power consumption and electromagnetic analysis to distinguish *functional vs dummy subtractions*) in order to detect whether an eXtra-reduction is performed ($X = 1$) or not ($X = 0$) during the Montgomery reduction (Alg. 2.1).

5.1 Simulations

Let `RSA-1024-p` defined at Sec.2.2the modulus p used in the SMA algorithm (Alg. 1.1). We generated one thousand random queries and saved for all MMM the information whether an extra-reduction is done or not. The length of static key k is 512 bits. As detailed in the ρ-attack (Alg. 4.1) we computed the estimated probabilities $\hat{\mathbb{P}}(X_{M_i}, X_{S_{i-1}})$ and the estimated Pearson correlation $\hat{\rho}(X_{M_i}, X_{S_{i-1}})$ to retrieve each k_i. The estimated threshold $\mathcal{T}$ computed by (9) in our simulation is equal to -0.06076, which is an excellent approximation of the theoretical threshold (7). To retrieve each bit if the exponent, we used the decision function $\mathcal{F}_{SMA}$ described for ρ-attack in SMA by (10).

Fig. 4 shows the estimated Pearson correlation values $\hat{\rho}(X_{M_i}, X_{S_{i-1}})$ for the first iterations. It can be easily seen that the estimated key value by this sequence corresponds to `0x1`$\ldots$`011`$\ldots$ = `0x11f5dd3`$\ldots$ Our ρ-attack retrieves the 511 bits of the exponent using 1000 randoms queries with success rate 100%.

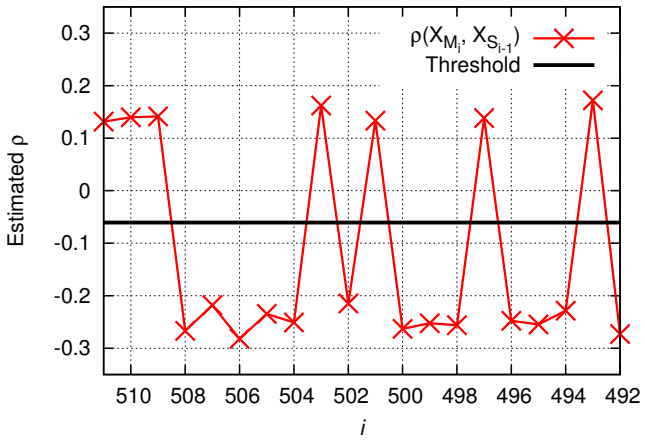


Fig. 4. Estimated Pearson correlations using 1000 randoms queries for `RSA-1024-p` for the first 20 iterations.

Success Rate Curves. We implemented ρ-attack-Hard and ρ-attack-Soft in the ideal case, i.e., without noise. The success rate to recover one bit of the exponent is represented in Fig. 5, for both SMA and ML cases. Interestingly, ρ-attack-Hard and ρ-attack-Soft yield the same success rate, which happens to be (very close to) the optimal value. This optimal value is that obtained with the maximum likelihood distinguisher discussed in Appendix C.

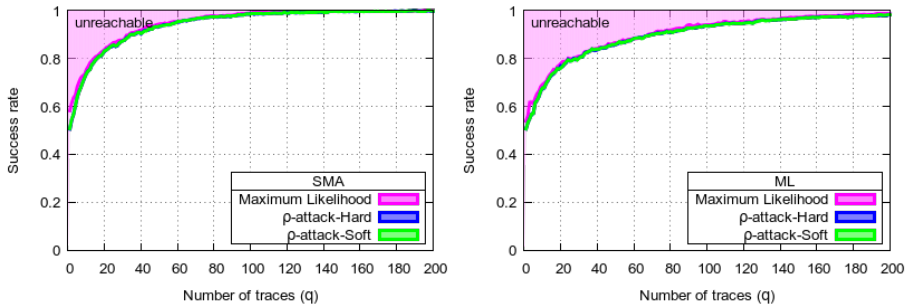


Fig. 5. Evolution of the success rate for the ρ-attack-Soft and the ρ-attack-Hard as a function of the number Q of queries (upper bound is the maximum likelihood), for `RSA-1024-p`.

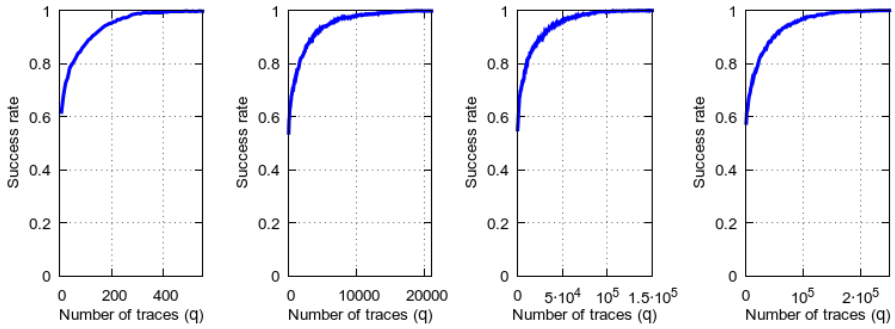
The reason for the hard and soft attacks to have similar success probability is that the online estimation of the threshold is very good. Indeed, in the example

of Fig. 5, the threshold $\mathcal{T}$ (Eq. (9)) is estimated based on $512Q$ traces, which is huge (one needs only to estimate 4 probabilities to get the estimation of $\mathcal{T}$). So, in the rest of this section, we make no difference between the hard and soft versions of the attacks from a success rate point of view.

The ρ-attacks are very close to the Maximum Likelihood attack for a similar reason. Estimating the difference between two random variables of very little dimensionality (recall that $(X_{M_i}, X_{S_{i-1}})$ lives in $\{0,1\}^2$) can be done almost equivalently in the proportional scale [27] (Pearson correlation) as in the context of information theoretic attacks (maximum likelihood attack) App. C.

We may also notice that as the *distinguisher margin* [26] is larger for SMA than for ML (recall Fig. 3), the former attack requires less traces to reach a given success rate.

In practical cases, detecting an extra-reduction using only one acquisition can lead to errors. The probability to have an error is denoted by $\mathbb{P}_{\text{noise}}$. We show in Fig. 6 that the attack continues to be successful (albeit with more traces) over a large range of $\mathbb{P}_{\text{noise}}$ values. Evidently when $\mathbb{P}_{\text{noise}} = 50\%$ the attack becomes infeasible.



(a) $\mathbb{P}_{\text{noise}} = 10\%$ (b) $\mathbb{P}_{\text{noise}} = 20\%$ (c) $\mathbb{P}_{\text{noise}} = 30\%$ (d) $\mathbb{P}_{\text{noise}} = 40\%$

Fig. 6. Evolution of the success rate for the ρ-attack in function of queries Q using $p =$ `RSA-1024-p` for four increasing noise values.

5.2 Experimental Detection of Extra-Reductions

Two Montgomery reduction implementations will be analyzed in this section. We raise the following questions.

1. How to exploit the *local timing* to distinguish the eXtra-reduction using power consumption measurements, on OpenSSL v1.0.1k-3 ([7])?

[7] Latest stable version at the time of submission.

2. How to exploit the difference between a *real* and a *dummy* final subtraction using electromagnetic (EM) emanations, on mbedTLS v 2.2.0 ([8])?

1a) Experiment Setup in Power. The target is a dual core LPC43S37 micro-controller fabricated in CMOS 90 nm Ultra Low Leakage process soldered on an LPCXpresso4337 board, and running at its maximum frequency (208 MHz). The side-channel traces were obtained measuring the instantaneous power consumption with a PICOSCOPE 6402C featuring 256 MB of memory, 500 MHz bandwidth and 5 GS/s sampling rate. We executed the private function of RSA in OpenSSL with the private primes parameters defined by `RSA-1024-p` and `RSA-1024-q` defined in Sec. 2.2.The private modular exponentiation is RSA-CRT with a regular algorithm.

1b) OpenSSL Experiment. In OpenSSL (see Listing 1.1 in Appendix D), the final subtraction is made when U is greater than p like described in Alg. 2.1. A simple power analysis using the delay (referred to as "`SPA-Timing`") between two MMM operations found whether the extra-reduction is present ($X = 1$) or not ($X = 0$). On the Cortex M4 core, the delay between the M_i and S_{i-1} when $X_{M_i} = 1$ is 41.4952 μs, whereas the delay when $X_{M_i} = 0$ is 41.1875 μs. For the square operation S_{i-1}, the delay is 41.5637 μs when $X_{S_{i-1}} = 1$ and it is 41.2471 μs when $X_{S_{i-1}} = 0$. All in one, the observable timing differences are respectively 308 ns and 317 ns. When OpenSSL is offloaded on the Cortex M0 core of the LPC43S37, the timing difference is respectively 399 ns and 411 ns. The success rate of this detection attack is 100%, hence $\mathbb{P}_{\text{noise}} = 0$.

2a) Experiment Setup in EM. The target device is an STM32F4 micro-controller, which contains an ARM Cortex-M4 processor running at its maximum frequency (168 MHz). For the acquisition, we used a Tektronix oscilloscope and a Langer near field probe. The sampling frequency is 1 GSa/s with 50 MHz hardware input low-pass filter enabled. The position of the probe was determined to maximize the signal related to the activity of the hardware 32×32 processor. We executed the private function of RSA in mbedTLS, with the private primes parameters defined by `RSA-1024-p` and `RSA-1024-q` in 2.2.The private modular exponentiation is RSA-CRT with a regular algorithm.

2b) mbedTLS Experiment. In order to achieve constant-time MMM, mbedTLS library implements a countermeasure using a dummy subtraction (see Listing 1.2 in Appendix D). In order to test the efficiency of the countermeasure, the duration of the real and dummy subtraction were compared as shown in Fig. 7. The durations are the same. Therefore, the `SPA-Timing` attack is not practical anymore.

In a view to differentiate the two patterns, we use a horizontal side-channel analysis [3], namely Pearson correlation (`max-corr`) [4] or the sum of the absolute differences (`min-abs-diff`). We build two reference patterns of the real

[8] Latest version at the time of submission.

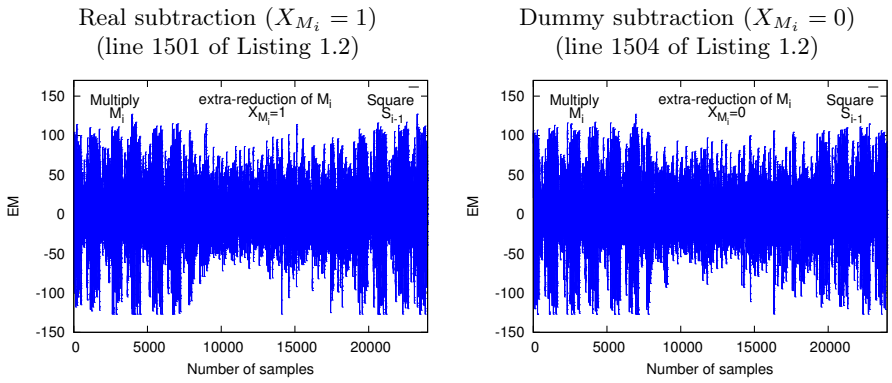


Fig. 7. Electromagnetic acquisition focus on one real subtraction (*left*) and pattern of one dummy subtraction (*right*) between two consecutive MMM operations.

subtraction $RP(X = 1)$ and dummy subtraction $RP(X = 0)$, and compare these patterns with one acquisition.

For this experiment, we use 500 acquisitions to build template $RP(X = 1)$ and again 500 acquisitions to make $RP(X = 0)$. The detection attack using one acquisition $\mathcal{A}_x$ where the extra-reduction $X = x$ is considered successful:

- when $\rho(\mathcal{A}_x, RP(X = x)) > \rho(\mathcal{A}_x, RP(X = \neg x))$ for `max-corr`, and
- when $\mathbb{E}(|\mathcal{A}_x - RP(X = x)|) < \mathbb{E}(|\mathcal{A}_x - RP(X = x)|)$ for `min-abs-diff`.

The success rate of the extra-reduction detection using 30000 acquisitions is 82.50% for `max-corr` and 83.47% for `min-abs-diff`, hence $\mathbb{P}_{\text{noise}} < 20\%$.

5.3 Conclusions on Experiments

By combining the detection of extra-reductions using side-channel analysis (Section 5.2) and the theoretical attack to decide whether or not there is a dependency between various MMMs (Section 4), we deduce the number of queries Q needed to recover the secret exponent k. Table 3 summaries the results.

6 Conclusion

This paper has presented a new theoretical and practical attack against asymmetrical computation with regular exponentiation using extra-reductions as a side-channel. The working factor is the existence of a strong bias between the extra-reductions during the Montgomery Modular Multiplication of two consecutive operations. This new bias can be exploited in each regular binary algorithm, because each iteration consists in a square and a multiply whose inputs depend on the outputs of an operation from the previous iteration.

Type of attack side-channel for detection	`SPA-Timing`	`max-corr`	`min-abs-diff`
Detection probability for one query $= 1 - \mathbb{P}_{\text{noise}}$	100%	82.50%	83.47%
Number of queries (SMA)	≈ 200	≈ 10000	≈ 10000
Number of queries (ML)	≈ 400	≈ 20000	≈ 20000

Table 3. Summary of the number of queries (see Fig. 6(b)) to retrieve all key bits of a secret exponent, as a function of side-channel detection method and regular exponentiation algorithm.

The new attacks have been detailed on RSA but are also applicable to ECC with appropriate customizations for various ECC implementations. As an example [5] for addition `madd-2004-hmv`, the Z-coordinate in output of addition is computed by a multiplication $Z3 = Z1 \times T1$ and for doubling `dbl-2007-bl`, the Z-coordinate in input of doubling is a square $ZZ = Z1 \times Z1$.

Acknowledgements

The authors would like to thank the anonymous reviewers for their useful comments that improved the quality of the paper. The first author would also like to thank François Dassance, Jean-Christophe Courrège and her colleagues for the suggestion of the main idea of this paper and their valuable insights.

References

1. Onur Aciiçmez and Werner Schindler. A vulnerability in RSA implementations due to instruction cache analysis and its demonstration on openssl. In Tal Malkin, editor, *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings*, volume 4964 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2008.
2. Onur Aciiçmez, Werner Schindler, and Çetin Kaya Koç. Improving Brumley and Boneh timing attack on unprotected SSL implementations. In Vijay Atluri, Catherine Meadows, and Ari Juels, editors, *Proceedings of the 12th ACM Conference on Computer and Communications Security, CCS 2005, Alexandria, VA, USA, November 7-11, 2005*, pages 139–146. ACM, 2005.
3. Aurélie Bauer, Éliane Jaulmes, Emmanuel Prouff, Jean-René Reinhard, and Justine Wild. Horizontal collision correlation attack on elliptic curves – Extended Version. *Cryptography and Communications*, 7(1):91–119, 2015.
4. Pierre Belgarric, Shivam Bhasin, Nicolas Bruneau, Jean-Luc Danger, Nicolas Debande, Sylvain Guilley, Annelie Heuser, Zakaria Najm, and Olivier Rioul. Time-Frequency Analysis for Second-Order Attacks. In Aurélien Francillon and Pankaj Rohatgi, editors, *CARDIS*, volume 8419 of *LNCS*, pages 108–122. Springer, 2013.
5. Daniel J. Bernstein and Tanja Lange. Explicit formulas database. `http://www.hyperelliptic.org/EFD/`.

6. Alexandre Berzati, Cécile Canovas-Dumas, and Louis Goubin. Public key perturbation of randomized RSA implementations. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 306–319. Springer, 2010.
7. BSI. RFC 5639 - Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation. Technical report, Bundesamt für Sicherheit in der Informationstechnik (BSI), 2010.
8. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Horizontal correlation analysis on exponentiation. In Miguel Soriano, Sihan Qing, and Javier Lopez, editors, *Information and Communications Security*, volume 6476 of *Lecture Notes in Computer Science*, pages 46–61. Springer-Verlag, 2010.
9. Jean-Christophe Courrège, Benoit Feix, and Mylène Roussellet. Simple power analysis on exponentiation revisited. In Dieter Gollmann, Jean-Louis Lanet, and Julien Iguchi-Cartigny, editors, *Smart Card Research and Advanced Application, 9th IFIP WG 8.8/11.2 International Conference, CARDIS 2010, Passau, Germany, April 14-16, 2010. Proceedings*, volume 6035 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2010.
10. Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, July 18 2006. ISBN-10: ISBN-10: 0471241954, ISBN-13: 978-0471241959, 2nd edition.
11. Pierre-Alain Fouque, Denis Réal, Frédéric Valette, and M'hamed Drissi. The carry leakage on the randomized exponent countermeasure. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, volume 5154 of *Lecture Notes in Computer Science*, pages 198–213. Springer, 2008.
12. Neil Hanley, HeeSeok Kim, and Michael Tunstall. Exploiting collisions in addition chain-based exponentiation algorithms using a single trace. In Kaisa Nyberg, editor, *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, volume 9048 of *Lecture Notes in Computer Science*, pages 431–448. Springer, 2015.
13. Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.
14. Paul C. Kocher. On certificate revocation and validation. In Rafael Hirschfeld, editor, *Financial Cryptography, Second International Conference, FC'98, Anguilla, British West Indies, February 23-25, 1998, Proceedings*, volume 1465 of *Lecture Notes in Computer Science*, pages 172–177. Springer, 1998.
15. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, October 1996. `http://www.cacr.math.uwaterloo.ca/hac/`.
16. Peter L. Montgomery. Modular multiplication without trial division. *Math. Comput.*, 44(170):519–521, April 1985.
17. NIST. FIPS publication 186-4 - Digital Signature standard (DSS). Technical report, National Institute of Standards and Technology (NIST), July 2013.

18. Siddika Berna Örs, Lejla Batina, Bart Preneel, and Joos Vandewalle. Hardware implementation of an elliptic curve processor over GF(p) with Montgomery modular multiplier. *IJES*, 3(4):229–240, 2008.
19. Hisayoshi Sato, Daniel Schepers, and Tsuyoshi Takagi. Exact analysis of montgomery multiplication. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings*, volume 3348 of *Lecture Notes in Computer Science*, pages 290–304. Springer, 2004.
20. Werner Schindler. A timing attack against RSA with the chinese remainder theorem. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings*, volume 1965 of *Lecture Notes in Computer Science*, pages 109–124. Springer, 2000.
21. Werner Schindler. A combined timing and power attack. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*, volume 2274 of *Lecture Notes in Computer Science*, pages 263–279. Springer, 2002.
22. Werner Schindler. Exclusive exponent blinding may not suffice to prevent timing attacks on RSA. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *Lecture Notes in Computer Science*, pages 229–247. Springer, 2015.
23. Werner Schindler, François Koeune, and Jean-Jacques Quisquater. Improving divide and conquer attacks against cryptosystems by better error detection / correction strategies. In Bahram Honary, editor, *Cryptography and Coding, 8th IMA International Conference, Cirencester, UK, December 17-19, 2001, Proceedings*, volume 2260 of *Lecture Notes in Computer Science*, pages 245–267. Springer, 2001.
24. Werner Schindler and Colin D. Walter. More detail for a combined timing and power attack against implementations of RSA. In Kenneth G. Paterson, editor, *Cryptography and Coding, 9th IMA International Conference, Cirencester, UK, December 16-18, 2003, Proceedings*, volume 2898 of *Lecture Notes in Computer Science*, pages 245–263. Springer, 2003.
25. Colin D. Walter and Susan Thompson. Distinguishing exponent digits by observing modular subtractions. In David Naccache, editor, *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, volume 2020 of *Lecture Notes in Computer Science*, pages 192–207. Springer, 2001.
26. Carolyn Whitnall and Elisabeth Oswald. A Comprehensive Evaluation of Mutual Information Analysis Using a Fair Evaluation Framework. In Phillip Rogaway, editor, *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2011.
27. Carolyn Whitnall, Elisabeth Oswald, and François-Xavier Standaert. The Myth of Generic DPA ... and the Magic of Learning. In Josh Benaloh, editor, *CT-RSA*, volume 8366 of *Lecture Notes in Computer Science*, pages 183–205. Springer, 2014.
28. Marc F. Witteman, Jasper G. J. van Woudenberg, and Federico Menarini. Defeating RSA multiply-always and message blinding countermeasures. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Pro-*

ceedings, volume 6558 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2011.

A Discussion

A.1 Attack using consecutive square operations

The attack works too considering the input/output dependencies between the eXtra-reduction of two consecutive square. To find the bit value k_i of the key, we compute the theoretical probabilities $\mathbb{P}(X_{S_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$ and $\mathbb{P}(X_{S_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$, and make the same kind of attack like described in Sec. 4. In Fig. 1, we can see when there is a dependency between the output of S_i and the input of S_{i-1} according to the bit value k_i during the modular exponentiation SMA. We can clearly see that the bit estimation $\hat{k}_i$ in all the attack algorithms is the opposite of the estimation described by (10). Indeed, when the bit value is 0, the input of square S_{i+1} is equal to the output of the square S_i ($\mathcal{G}_i = T$), and when the bit value is 1, the input of square S_{i-1} is different from the output of the square S_i ($\mathcal{G}_i = F$). We have the same relation between $\rho(X_{S_i}, X_{S_{i-1}}|\mathcal{G}_i = T)$ and $\rho(X_{S_i}, X_{S_{i-1}}|\mathcal{G}_i = F)$. So, the new decision function $\mathcal{F}_{SMA}$ is defined by:

$$\hat{k}_i = \mathcal{F}_{SMA}(\rho, \mathcal{T}) = \begin{cases} 0 & \text{if } \rho(X_{S_i}, X_{S_{i-1}}) \leq \mathcal{T}, \\ 1 & \text{otherwise.} \end{cases} \tag{12}$$

The main advantage of using only the squares is that the time to retrieve the eXtra-reduction is divided by two. When the length of the exponent is 1024 bits, we need to detect only 1024 eXtra-reductions during the square operations. Note that the best strategy is to use both methods to increase the success rate, and reduce the number of required queries.

Remark: For the ML algorithm, we can choose the two consecutive squares S_i and S_{i-1}. If the two bits value k_i and k_{i-1} are equal, then the Output/Input of two squares are equals ($\mathcal{G}_i = T$), else these operations are independent ($\mathcal{G}_i = F$).

A.2 Other exponentiation implementations

This work limits the attack to binary regular algorithms, but some other algorithms can also be attacked by exploiting correlations between eXtra-reductions.

Multiply-always. The Multiply-Always algorithm applies when the square implementation is the same as the multiply operation. In ECC, it is equivalent to the Add-Always algorithm, when the doubling and adding operations are unified. These kinds of algorithms can be attacked by the classical Square-and-Multiply proposed by Schindler in [20] or his improvements [1, 2, 23] using the result of Prop. 1.

Window algorithm. The window exponentiation algorithm is used to provide efficient computation. The timing attacks proposed by Schindler in [21] allows to find the exponent during the window algorithm, also with the help of extra-reductions.

A.3 Efficient countermeasures

To avoid our attack, one of the following countermeasures described here is sufficient.

Use Exponent Blinding. For RSA (resp. ECC), a classic countermeasures is the exponent blinding (resp. scalar blinding). This countermeasure is efficient to avoid the attack, because to compute the estimated probabilities for the Q queries, the exponent bit must be fixed. Although Berzati and al. in [6] show that the exponent blinding is partially ineffective on some bits depending on the chosen modulo, the bias seems not easily exploitable.

Use another Montgomery Reduction. In [18, Alg. 2], Örs and al. describe an MMM algorithm without final subtraction. Thus, there are no eXtra-reductions to exploit in our attack.

B Proof of theorems 1 and 2

B.1 Technical lemmas

Before proving theorems, we need the following technical lemmas 1, 2, 3, 4 and 5.

Let A and B two independent discrete random variables uniformly distributed on $\{0, \ldots, p-1\}$. In the sequel, we are interested in asymptotic convergence in distribution when $p \to +\infty$. In general, it is known that a series of discrete random variables X_p converges in distribution to X (continuous random variable, having a density function), if and only if there is a convergence of cumulative density functions (c.d.f.):

$$\mathbb{P}(X_p \leq x) \to \mathbb{P}(X \leq x) \qquad \text{when } p \to +\infty.$$

This is why we will work on c.d.f. Besides, it is well known that if A is uniformly distributed on $[0, p-1]$, then A/p converges in distribution to $\mathcal{U}([0,1])$ when $p \to +\infty$. Indeed, for all $x \in]0,1[$,

$$\mathbb{P}\big(\frac{A}{p} \leq x\big) = \sum_{0 \leq a < px} \frac{1}{p} = \frac{\lfloor px \rfloor}{p} \to x = \int_0^x \mathrm{d}u.$$

Now, it is incorrect to write that for $0 \leq a < p$, the limit of $\mathbb{P}(A \leq a)$ is $\frac{a}{p}$ (since $p \to \infty$). However, we will use this abuse of notation in the sequel.

Lemma 1. *Let A, B two random variables uniformly distributed over $[0, p[$. Let x, y two values in $[0, p^2[$. Then, in the limit $p \to +\infty$, we have:*

$$f(x) = \frac{1}{p^2} \ln \frac{p^2}{x} \qquad \text{where } f \text{ is the density function of } AB, \quad (13)$$

$$f(y) = \frac{1}{2p\sqrt{y}} \qquad \text{where } f \text{ is the density function of } A^2, \quad (14)$$

$$f(x,y) = \frac{1}{2p^2 y} \mathbb{1}_{[0, \frac{\sqrt{y}}{p}]}\big(\frac{x}{p^2}\big) \qquad \text{where } f \text{ is the density function of } (AB, A^2). \quad (15)$$

Proof. Case of the product. As A and B are independent, (A, B) has a uniform law on $\{0, \dots, p-1\}^2$. Hence the convergence in distribution of the pair $(A/p, B/p)$ to a uniform law on $[0,1]^2$. Thus, for all $x \in]0, 1[$,

$$\mathbb{P}\big(\frac{AB}{p^2} \le x\big) \to \iint_{[0,1]^2} \mathbb{1}_{uv<x} \, \mathrm{d}u \, \mathrm{d}v \qquad \text{when } p \to +\infty.$$

The convergence is illustrated in Fig. 8 as the proportion of points below the hyperbola curve represented in red.

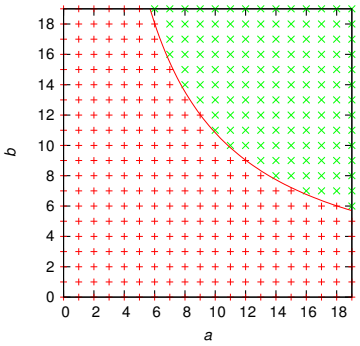


Fig. 8. Illustration of $\{(a,b) \in \{0, p-1\}^2 \mid ab < p^2 x\}$ for prime $p = 19$ and $x = 0.3$

As $uv < x \iff v < x/u$ and $x/u < 1 \iff u > x$, we have

$$\iint_{[0,1]^2} \mathbb{1}_{uv<x} \, \mathrm{d}u \, \mathrm{d}v = \int_0^1 \min(\frac{x}{u}, 1) \, \mathrm{d}u = \int_0^x \mathrm{d}u + \int_x^1 x \frac{\mathrm{d}u}{u} = x(1 - \ln x).$$

Thus, the limit distribution of AB/p^2 has for density the derivative:

$$f(x) = \ln \frac{1}{x}.$$

As $AB = p^2(AB/p^2)$, a variable change yields (13).

Case of the square. For all $y \in]0,1[$,

$$\mathbb{P}\big(\frac{A^2}{p^2} < y\big) = \mathbb{P}\big(\frac{A}{p} < \sqrt{y}\big) \to \sqrt{y} \qquad \text{when } p \to +\infty.$$

The limit distribution has for density the derivative:

$$f(y) = \frac{1}{2\sqrt{y}},$$

hence (14) by change of variable $A^2 = p^2(A^2/p^2)$.

Case of the pair multiplication and square. We have, for all $x, y \in]0,1[$:

$$\mathbb{P}\big(\frac{AB}{p^2} < x \;\Big|\; \frac{A^2}{p^2} = y\big) = \mathbb{P}\big(\frac{B}{p} < \frac{x}{\sqrt{y}} \;\Big|\; \frac{A^2}{p^2} = y\big) = \mathbb{P}\big(\frac{B}{p} < \frac{x}{\sqrt{y}}\big) \to \min(\frac{x}{\sqrt{y}}, 1)$$

because A and B are independent and because B is uniform. The conditional distribution of $(AB/p^2 \mid A^2/p^2 = y)$ has in the limit the (uniform) density:

$$f(x) = \frac{1}{\sqrt{y}} \mathbb{1}_{[0,\sqrt{y}]}(x)$$

and so, the joint probability $(AB/p^2, A^2/p^2)$ in $(x, y) \in [0,1]^2$ has the following limit

$$\mathbb{P}\big(AB/p^2 = x, A^2/p^2 = y\big) \to \frac{1}{\sqrt{y}} \mathbb{1}_{[0,\sqrt{y}]}(x) \frac{1}{2\sqrt{y}} = \frac{1}{2y} \mathbb{1}_{[0,\sqrt{y}]}(x) \quad \text{(by (14))}.$$

Again, a variable change yields (15).

□

Lemma 2. *Let A a random variable defined on $[0,p]$, with density f. Then the probability density function of A^2 in $z \in [0,p^2]$ is equal to $f(\sqrt{z})\frac{1}{2\sqrt{z}}$.*

Proof. Use (14) in lemma 1 in a variable change. □

Lemma 3. *Let u an integer such that $0 \le u < p$. The set $\mathcal{C}_u = \{z, 0 \le z < p^2,$ s.t. $z + (zv \bmod R)p = Ru\}$ is equal to $\mathcal{C}_u = \{(Ru \bmod p) + ip,$ where $0 \le i \le \min\Big(p, \lfloor \frac{Ru}{p} \rfloor\Big)\}$.*

Proof. Let z such that $z + (zv \bmod R)p = Ru$. Clearly, we have $(z \bmod p) = (Ru \bmod p)$, hence $\mathcal{C}_u \subseteq \{((Ru \bmod p) + ip,$ where $i \in \mathbb{N}\}$. But given the bounds on z, we have $0 \le i < p$. Let us precise which values of i make $(Ru \bmod p) + ip$ belong to $\mathcal{C}_u$.

We have $(Ru \bmod p) + ip + (((Ru \bmod p)v + ipv) \bmod R)p = Ru$, hence, as $Ru - (Ru \bmod p) = \lfloor \frac{Ru}{p} \rfloor$, $i + (((Ru \bmod p)v - i) \bmod R) = \lfloor \frac{Ru}{p} \rfloor$. In this expression, $(Ru \bmod p) = Ru - \lfloor \frac{Ru}{p} \rfloor p$. Let us denote $1 + vp = \ell R$, where $0 < \ell < p$. We have

$$(((Ru \bmod p)v - i) \bmod R) = (Ruv - \lfloor \frac{Ru}{p} \rfloor(\ell R - 1) - i \bmod R)$$

$$= (\lfloor \frac{Ru}{p} \rfloor - i \bmod R)$$
$$= \begin{cases} \lfloor \frac{Ru}{p} \rfloor - i & \text{if } 0 \le i \le \lfloor \frac{Ru}{p} \rfloor, \\ R + \lfloor \frac{Ru}{p} \rfloor - i & \text{if } i > \lfloor \frac{Ru}{p} \rfloor. \end{cases}$$

Consequently, the condition is met if and only if $0 \le i \le \lfloor \frac{Ru}{p} \rfloor$. Hence the proof of the lemma, as i is upper bounded both by p and $\lfloor \frac{Ru}{p} \rfloor$. □

Lemma 4 (Approximation of finite summations). *Let I a large number (comparable to p). When $I \to +\infty$, we have,*

$$\sum_{i=0}^{I} i^\alpha \to \frac{1}{1+\alpha} I^{1+\alpha} \text{ for } \alpha \in \mathbb{R} \setminus \{-1\}, \tag{16}$$
$$\sum_{i=1}^{I} 1/i \to \ln(I), \tag{17}$$
$$\sum_{i=0}^{I} \ln(i) \to I \ln(I) - I. \tag{18}$$

Proof. When $I \to +\infty$, $\sum_{i=0}^{I} i^\alpha \to \int_0^I x^\alpha \,\mathrm{d}x = \frac{1}{1+\alpha} I^{1+\alpha}$. Similarly, $\sum_{i=1}^{I} 1/i \to \int_1^I 1/x \,\mathrm{d}x = \ln(I)$. Eventually, by Stirling formula, we have that $\ln I! = I \ln I - I + \mathcal{O}(\ln I) \to I \ln I - I$. □

Lemma 5 (Miscellaneous approximations). *When $p \to +\infty$, we have*

$$\left\lfloor \frac{Ru}{p} \right\rfloor \to \frac{Ru}{p} \quad \text{for } 0 \le u \le p, \tag{19}$$
$$\frac{(Ru \bmod p) + ip}{p} \to i \quad \text{for } i \gg 1. \tag{20}$$

Proof. The first equation arises from $\lim_{p \to +\infty} \lfloor p \rfloor / p \to 1^-$, whereas the second one holds all the more for large values of i, since $(Ru \bmod p) < p \ll ip$ when $i \gg 1$. □

B.2 Proof of Theorem 1

Proof (of Theorem 1). As explained in Sec. B.1, $\mathcal{P}(U = u)$ tends to a density $f(U = u)$ when $p \to +\infty$. Case $0 \le u \le p$: let U be the result of MMM before final reduction (definition at line 2 of Alg. 2.1). We have, in the limit $p \to +\infty$:

$$\mathbb{P}(U = u) = \sum_{a=0}^{p-1} \sum_{b=0}^{p-1} \mathbb{P}(A = a, B = b) \; \mathbb{1}_{ab + (abv \bmod R)p = Ru}$$

$$= \frac{1}{p^2}\sum_{z=1}^{p^2} \ln\left(\frac{p^2}{z}\right) \mathbb{1}_{z+(zv \bmod R)p=Ru} \qquad \begin{array}{l}(\text{denote } z=ab \text{ and}\\ \text{apply (13) of lemma 1)}\end{array}$$

$$= \frac{1}{p^2}\sum_{i=1}^{\min\left(p,\lfloor\frac{Ru}{p}\rfloor\right)} \ln\left(\frac{p^2}{(Ru \bmod p)+ip}\right) \quad \text{(by lemma 3)}$$

$$\approx \frac{1}{p^2}\sum_{i=1}^{\min\left(p,\frac{Ru}{p}\right)} \ln\frac{p}{i} \quad \text{(by lemma 5)}$$

$$= \frac{\min\left(p,\frac{Ru}{p}\right)}{p^2}\left(\ln(p) - \ln(\min\left(p,\frac{Ru}{p}\right)) + 1\right) \quad \text{(by (18) in lemma 4)}$$

$$= \begin{cases} \frac{uR}{p^3}\left(1-\ln\frac{uR}{p^2}\right) & \text{if } p \le \frac{Ru}{p}, \text{ i.e., } u \ge \frac{p^2}{R},\\ \frac{1}{p} & \text{if } p > \frac{Ru}{p}, \text{ i.e., } u < \frac{p^2}{R}. \end{cases}$$

Case $p < u < 2p$: By the definition of U and C, for each $c \in [0,p[$ we have

$$\mathbb{P}(C=c) = \mathbb{P}(U=c) + \mathbb{P}(U=c+p).$$

$C = ABR^{-1} \bmod p$, so C is a random variable uniformly distributed over $[0,p[$.

Then, $p < u < 2p$, we have, by the definition of C (at line 4 or 6 of Alg. 2.1):

$$\mathbb{P}(U=u) = \mathbb{P}(C=u-p) - \mathbb{P}(U=u-p) = 1/p - \mathbb{P}(U=u-p).$$

In the case where $A = B$, the demonstration is similar. For $0 \le u \le p$, we have:

$$\mathbb{P}(U=u) = \sum_{z=0}^{p^2} \frac{1}{2p\sqrt{z}} \mathbb{1}_{z+(zv \bmod R)p=Ru} \qquad \begin{array}{l}(\text{denote } z=a^2 \text{ and}\\ \text{apply (14) of lemma 1)}\end{array}$$

$$= \frac{1}{2p}\sum_{i=1}^{\min\left(p,\lfloor\frac{Ru}{p}\rfloor\right)} ((Ru \bmod p)+ip)^{-1/2} \quad \text{(by lemma 3)}$$

$$\approx \frac{1}{2p}\sum_{i=1}^{\min\left(p,\frac{Ru}{p}\right)} (ip)^{-1/2} \quad \text{(by lemma 5)}$$

$$= \frac{1}{p^{3/2}}\sqrt{\min\left(p,\frac{Ru}{p}\right)} \quad \text{(by (16) for } \alpha = -1/2 \text{ in lemma 4)}$$

$$= \begin{cases} \frac{\sqrt{uR}}{p^2} & \text{if } p \le \frac{Ru}{p}, \text{ i.e., } u \ge \frac{p^2}{R},\\ \frac{1}{p} & \text{if } p > \frac{Ru}{p}, \text{ i.e., } u < \frac{p^2}{R}. \end{cases}$$

□

B.3 Proof of Theorem 2

Here is the proof of Theorem 2:

Proof (of Theorem 2). Let A a random variable resulting from an MMM with an extra-reduction, and denote by f its density function. Then we have $f(a) = 1/p - aR/p^3(1 - \ln(aR/p^2))$ if $0 \le a \le p^2/R$, and 0 if $p^2/R \le a \le p$. We denote by U the value of A^2 before final reduction.

First case, where $0 < u < p$ (Lemma 3 applies): We have:

$$\begin{aligned}
\mathbb{P}(U = u) &= \sum_{z=0}^{p^2} f(z) \frac{1}{2p\sqrt{z}} \mathbb{1}_{z+(zv \bmod R)p = Ru} \quad \text{(use lemma 2)} \\
&= \sum_{i=1}^{I} f(\sqrt{(Ru \bmod p) + ip}) \frac{1}{2\sqrt{(Ru \bmod p) + ip}} \quad \text{(by lemma 3)} \\
&\approx \sum_{i=1}^{I} f(\sqrt{ip}) \frac{1}{2\sqrt{ip}}, \qquad \text{(by approximation (20) of lemma 5)} \quad (21)
\end{aligned}$$

where the upper bound I is $\min(p, \frac{Ru}{p}, \frac{1}{p}\left(\frac{p^2}{R}\right)^2)$. As $\frac{1}{p}\left(\frac{p^2}{R}\right)^2 = p(p/R)^2 < p$, we have:

$$I = \begin{cases} \frac{p^3}{R^2} & \text{if } \frac{Ru}{p} > \frac{p^3}{R^2}, \text{ i.e., } u > \frac{p^4}{R^3} = p(p/R)^3 \text{ (but with constraint } p > u), \\ \frac{Ru}{p} & \text{otherwise.} \end{cases}$$

We can rewrite (21) (where the dependency in u is via I) as

$$\begin{aligned}
\mathbb{P}(U = u) &= \frac{1}{2\sqrt{p}} \sum_{i=1}^{I} \frac{1}{\sqrt{i}} \left(\frac{1}{p} - \sqrt{i} \frac{R}{p^{5/2}} (1 - \ln(\frac{R}{p^{3/2}}) - \frac{1}{2}\ln(i)) \right) \\
&= \frac{1}{2p^{3/2}} \sum_{i=1}^{I} \frac{1}{\sqrt{i}} - \frac{R}{2p^3} \sum_{i=1}^{I} \left(1 - \ln(\frac{R}{p^{3/2}}) - \frac{1}{2}\ln(i) \right) \\
&= \frac{\sqrt{I}}{p^{3/2}} - \frac{IR}{2p^3} \left(1 - \ln(\frac{R}{p^{3/2}}) \right) + \frac{R}{4p^3} \sum_{i=1}^{I} \ln(i) \qquad \text{(Using (16) of lemma 4 for } \alpha = 1/2) \\
&= \frac{\sqrt{I}}{p^{3/2}} - \frac{IR}{2p^3} \left(1 - \ln(\frac{R}{p^{3/2}}) \right) + \frac{RI}{4p^3} (\ln(I) - 1) \quad \text{(Using (18) of lemma 4)} \\
&= \frac{\sqrt{I}}{p^{3/2}} + \frac{IR}{4p^3} \left(-2 + 2\ln(\frac{R}{p^{3/2}}) + \ln(I) - 1 \right) \\
&= \frac{\sqrt{I}}{p^{3/2}} + \frac{IR}{4p^3} \left(\ln(\frac{R^2 I}{p^3}) - 3 \right)
\end{aligned}$$

So, when $p > u > p(p/R)^3$, we have:

$$\mathbb{P}(U = u) = \frac{1}{R} + \frac{1}{4R}(-3) = \frac{1}{4R}.$$

And when $u < p(p/R)^3$, we have:

$$\mathbb{P}(U = u) = \frac{\sqrt{Ru}}{p^2} + \frac{R^2 u}{4p^4}\left(\ln(\frac{R^3 u}{p^4}) - 3\right).$$

Second case, where $p < u < 2p$): Like in Theorem 1, we have $\mathbb{P}(U = u) = \frac{1}{p} - \mathbb{P}(U = u - p)$.

Now, we have that (first case situation only):

$$\begin{aligned}
&\mathbb{P}(X_{M_i} = 1, X_{S_{i-1}} = 0 \mid \mathcal{G}_i = T)\\
&= \int_0^p \mathbb{P}(U = u)\,\mathrm{d}u\\
&= \int_0^{p^4/R^3} \frac{\sqrt{Ru}}{p^2} + \frac{R^2 u}{4p^4}\left(\ln(\frac{R^3 u}{p^4} - 3\right)\,\mathrm{d}u + \frac{p}{4R}(1 - \frac{p^3}{R^3})\\
&= \frac{p^4}{R^4}\int_0^1 \sqrt{u'}\,\mathrm{d}u' + \frac{p^4}{4R^4}\int_0^1 u'(\ln u' - 3)\,\mathrm{d}u' + \frac{p}{4R}(1 - \frac{p^3}{R^3}) \quad (u' = uR^3/p^4)\\
&= \frac{p^4}{R^4}\left[\frac{2}{3}u'^{3/2}\right]_0^1 + \frac{p^4}{4R^4}\left[\frac{1}{2}u'^2\ln(u') - \frac{1}{4}u'^2\right]_0^1 - \frac{3p^4}{4R^4}\left[\frac{u'^2}{2}\right]_0^1 + \frac{p}{4R}(1 - \frac{p^3}{R^3})\\
&= \frac{p}{4R} + \frac{p^4}{R^4}\left(\frac{2}{3} - \frac{1}{16} - \frac{3}{8} - \frac{1}{4}\right) = \frac{p}{4R} - \frac{p^4}{48R^4}.
\end{aligned}$$

Other entries of the table of Theorem 2 corresponding to $\mathcal{G}_i = T$ can be deduced from the partial probabilities given in Proposition 1 namely

$$\begin{aligned}
&\mathbb{P}(X_{M_i}=0, X_{S_{i-1}}=0|\mathcal{G}_i = T) + \mathbb{P}(X_{M_i}=1, X_{S_{i-1}}=0|\mathcal{G}_i = T) = \mathbb{P}(X_{S_{i-1}}=0) = 1 - \frac{p}{3R},\\
&\mathbb{P}(X_{M_i}=0, X_{S_{i-1}}=1|\mathcal{G}_i = T) + \mathbb{P}(X_{M_i}=1, X_{S_{i-1}}=1|\mathcal{G}_i = T) = \mathbb{P}(X_{S_{i-1}}=1) = \frac{p}{3R},\\
&\mathbb{P}(X_{M_i}=0, X_{S_{i-1}}=0|\mathcal{G}_i = T) + \mathbb{P}(X_{M_i}=0, X_{S_{i-1}}=1|\mathcal{G}_i = T) = \mathbb{P}(X_{M_i}=0) = 1 - \frac{p}{4R},\\
&\mathbb{P}(X_{M_i}=1, X_{S_{i-1}}=0|\mathcal{G}_i = T) + \mathbb{P}(X_{M_i}=1, X_{S_{i-1}}=1|\mathcal{G}_i = T) = \mathbb{P}(X_{M_i}=1) = \frac{p}{4R}.
\end{aligned}$$

Regarding the case $\mathcal{G}_i = F$ for SMA, we shall compute the probability density that a multiplication before extra-reduction is equal to u_1 and that a square is equal to u_2, whereby one operand of the multiplication is the input of the square and the second operand is uniformly distributed over $[0, p]$. Let us assume $0 \leq u_1, u_2 < p$. This density is equal to:

$$\begin{aligned}
&f(U_1 = u_1, U_2 = u_2)\\
&= \sum_{x=0}^{p}\sum_{y=0}^{p} f(AB = x, A^2 = y)\ \mathbb{1}_{x+(xv \bmod R)p = u_1 R}\ \mathbb{1}_{y+(yv \bmod R)p = u_2 R}\\
&= \sum_{x=0}^{p}\sum_{y=0}^{p} \frac{1}{2p^2 y}\mathbb{1}_{[0,\frac{\sqrt{y}}{p}]}\left(\frac{x}{p^2}\right)\ \mathbb{1}_{x+(xv \bmod R)p = u_1 R}\ \mathbb{1}_{y+(yv \bmod R)p = u_2 R} \quad \text{(by (15) of lemma 1)}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i_1=0}^{\min(\lfloor Ru_1/p \rfloor, p)} \sum_{i_2=0}^{\min(\lfloor Ru_2/p \rfloor, p)} \frac{\mathbb{1}_{\left[0, \frac{\sqrt{(Ru_2 \bmod p)+i_2 p}}{p}\right]} \left(\frac{(Ru_1 \bmod p)+i_1 p}{p^2}\right)}{2p^2((Ru_2 \bmod p)+i_2 p)} \quad \text{(by lemma 3)} \\
&\approx \frac{1}{2p^3} \sum_{i_2=0}^{\min(\lfloor Ru_2/p \rfloor, p)} \frac{1}{i_2} \sum_{i_1=0}^{\min(\lfloor Ru_1/p \rfloor, p)} \mathbb{1}_{[0, \sqrt{i_2/p}]} \left(\frac{i_1}{p}\right) \quad \text{(by (20) in lemma 5, twice)} \\
&= \frac{1}{2p^3} \sum_{i_2=0}^{\min(\lfloor Ru_2/p \rfloor, p)} \min\left(\sqrt{\frac{p}{i_2}}, \frac{\min(Ru_1/p, p)}{i_2}\right) \quad \text{(by (19) in lemma 5, twice).}
\end{aligned}
\tag{22}
$$

The figure 9 illustrates that the general formula (22) takes five different expressions depending on the regions where (u_1, u_2) live.

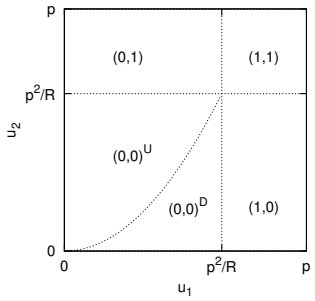


Fig. 9. Study of values $p(U_1 = u_1, U_2 = u_2 \mid \mathcal{G}_i = F)$ for the case SMA

They are detailed below:

– $(0,0)^D$: when $u_1 \leq p^2/R$, $u_2 \leq p^2/R$, and $u_2 \leq \frac{R}{p^2} u_1^2$:

$$f(U_1 = u_1, U_2 = u_2) = \frac{\sqrt{Ru_2}}{p^3}.$$

– $(0,0)^U$: when $u_1 \leq p^2/R$, $u_2 \leq p^2/R$, and $u_2 > \frac{R}{p^2} u_1^2$:

$$f(U_1 = u_1, U_2 = u_2) = \frac{u_1 R}{p^4} + \frac{u_1 R}{2p^4} \ln \frac{p^2 u_2}{u_1^2 R}.$$

– $(1,0)$: when $p^2/R \leq u_1 < p$ and $u_2 \leq p^2/R$:

$$f(U_1 = u_1, U_2 = u_2) = \frac{\sqrt{Ru_2}}{p^3} \qquad \text{(same expression as in neighboring region } (0,0)^D\text{)}.$$

– $(0,1)$: when $u_1 \leq p^2/R$ and $p^2/R \leq u_2 < p$:

$$f(U_1 = u_1, U_2 = u_2) = \frac{u_1 R}{p}\left(1 - \frac{u_1 R}{p^2}\right).$$

– $(1,1)$: when $p^2/R \leq u_1 < p$ and $p^2/R \leq u_2 < p$:

$$f(U_1 = u_1, U_2 = u_2) = \frac{1}{p^2}.$$

We have that, when $\mathcal{G}_i = F$ in SMA,

$$\begin{aligned}\mathbb{P}(X_{M_i} = 0, X_{S_{i-1}} = 0) &= \iint_{[0,p]^2} f(U_1 = u_1, U_2 = u_2)\,\mathrm{d}u_1\,\mathrm{d}u_2 \\ &= 1 - \frac{7}{12}\frac{p}{R} + \frac{1}{8}\left(\frac{p}{R}\right)^2.\end{aligned}$$

Again, partial probabilities given in Proposition 1 allow to derive the three other probabilities of the table of Theorem 2 corresponding to $\mathcal{G}_i = F$ in SMA.

Eventually, when $\mathcal{G}_i = F$ in ML, we have independent multiplication and square, hence the factorization $\mathbb{P}(X_{M_i} = 0, X_{S_{i-1}} = 1) = \mathbb{P}(X_{M_i} = 0)\mathbb{P}(X_{S_{i-1}} = 1)$. □

C Comparison of attacks in various adversarial contexts

In this appendix, we first model the attack setup and then derive the optimal attack (Maximum Likelihood) in terms of success probability. This attack can be performed in practice only provided that the noise distribution and the modulus p are known. Being optimal, it allows to bound the success rate of other attacks suited to other less favorable scenarios, like the hard and soft correlation attacks. For the sake of clarity, we focus on SMA regular exponentiation, but adaptation to the ML is straightforward. In SMA, the guess function $\mathcal{G}_i$ is directly k_i.

C.1 Leakage model

The attacker gets access to side-channel information about each bit k_i ($l - 1 \geq i > 0$) of the exponent k through the noised distribution of the pair of extra-reductions $(X_{M_i}, X_{S_{i-1}})$. The noise consists in two binary random variables $(N_{M_i}, N_{S_{i-1}})$. Additionally, the random variables N_{M_i} and $N_{S_{i-1}}$ are assumed independent and identically distributed (i.i.d.), as is usually the case of measurement noise of different operations in a side-channel trace. Namely, we denote by $\mathbb{P}_{\text{noise}}$ the probability

$$\mathbb{P}_{\text{noise}} = \mathbb{P}(N_{M_i} = 1) = \mathbb{P}(N_{S_{i-1}} = 1) \qquad \text{for all } i.$$

Thus, as depicted in Fig. 10, the attacker garners an i.i.d. sequence $(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}) = (y^q_{M_i}, y^q_{S_{i-1}})_{q=1,\ldots,Q}$, where for each query q and exponent index i, $y^q_{M_i} = x^q_{M_i} \oplus$

$n^q_{M_i}$ and $y^q_{S_{i-1}} = x^q_{S_{i-1}} \oplus n^q_{S_{i-1}}$. This means that X_{M_i} and Y_{M_i} are respectively the input and the output of a binary symmetric channel (BSC) of parameter $\mathbb{P}_{\text{noise}}$. Similarly, $X_{S_{i-1}}$ and $Y_{S_{i-1}}$ are also input and output of an independent identical BSC parallel to the first one.

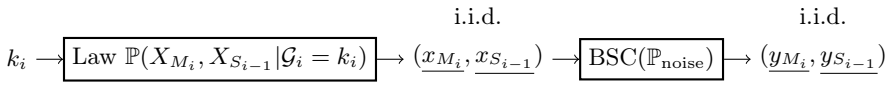


Fig. 10. Observable leakage corresponding to exponent bit k_i

C.2 Maximum likelihood attack

An attack consists in estimating $\hat{k}_i$ based on the observation of a series $(\underline{y_{M_i}}, \underline{y_{S_{i-1}}})$. Let us denote as $\mathbb{P}(\hat{k}_i = k_i)$ the probability of success in recovering k_i. The attacker wants to maximize it. We have

$$\mathbb{P}(\hat{k}_i = k_i) = \sum_{\underline{y_{M_i}}, \underline{y_{S_{i-1}}}} \mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}) \mathbb{P}(\hat{k}_i = k_i | \underline{y_{M_i}}, \underline{y_{S_{i-1}}}).$$

The term $\mathbb{P}(\hat{k}_i = k_i | \underline{y_{M_i}}, \underline{y_{S_{i-1}}})$ is thus to be maximized, which is called the MAP (Maximum A Posteriori).

We also have

$$\mathbb{P}(\hat{k}_i = k_i | \underline{y_{M_i}}, \underline{y_{S_{i-1}}}) = \frac{\mathbb{P}(k_i)\mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|k_i)}{\mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}})},$$

where the denominator does not depend on the key. If the key bit k_i is uniformly distributed, then one shall maximize $\mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|k_i)$, which is called the Maximum Likelihood.

So, we have:

$$\hat{k}_i = \underset{k_i \in \{0,1\}}{\operatorname{argmax}} \mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|k_i),$$

where, since we have a Markov chain (recall Fig. 10),

$$\begin{aligned}\mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|k_i) &= \sum_{\underline{x_{M_i}}, \underline{x_{S_{i-1}}}} \mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|\underline{x_{M_i}}, \underline{x_{S_{i-1}}}, k_i) \\ &= \sum_{\underline{x_{M_i}}, \underline{x_{S_{i-1}}}} \mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}}|\underline{x_{M_i}}, \underline{x_{S_{i-1}}}).\end{aligned}$$

As the BSC is a parallel composition of two independent BSC (which is called a channel extension, as defined in [10, p. 193]), we have $\mathbb{P}(\underline{y_{M_i}}, \underline{y_{S_{i-1}}} | \underline{x_{M_i}}, \underline{x_{S_{i-1}}}) = \mathbb{P}(\underline{y_{M_i}} | \underline{x_{M_i}}) \mathbb{P}(\underline{y_{S_{i-1}}} | \underline{x_{S_{i-1}}})$, where

$$\mathbb{P}(\underline{y_{M_i}} | \underline{x_{M_i}}) = (1 - \mathbb{P}_{\text{noise}})^{n - d_H(\underline{x_{M_i}}, \underline{y_{M_i}})} \mathbb{P}_{\text{noise}}^{d_H(\underline{x_{M_i}}, \underline{y_{M_i}})} \propto \left(\frac{\mathbb{P}_{\text{noise}}}{1 - \mathbb{P}_{\text{noise}}} \right)^{d_H(\underline{x_{M_i}}, \underline{y_{M_i}})}.$$

Hence

$$\hat{k}_i = \underset{k_i}{\text{argmax}} \sum_{\underline{x_{M_i}}, \underline{x_{S_{i-1}}}} \left(\frac{\mathbb{P}_{\text{noise}}}{1 - \mathbb{P}_{\text{noise}}} \right)^{d_H(\underline{x_{M_i}}, \underline{y_{M_i}}) + d_H(\underline{x_{S_{i-1}}}, \underline{y_{S_{i-1}}})} \mathbb{P}(\underline{x_{M_i}}, \underline{x_{S_{i-1}}} | k_i) \tag{23}$$

$$= \underset{k_i}{\text{argmax}} \ \mathbb{E}_{k_i} \left(\left(\frac{\mathbb{P}_{\text{noise}}}{1 - \mathbb{P}_{\text{noise}}} \right)^{d_H(\underline{X_{M_i}}, \underline{y_{M_i}}) + d_H(\underline{X_{S_{i-1}}}, \underline{y_{S_{i-1}}})} \right). \tag{24}$$

Equation (24) is a rewriting of (23), where the expectation $\mathbb{E}_{k_i}$ is taken over distribution $\mathbb{P}(X_{S_i}, X_{M_{i-1}} | \mathcal{G}_i = k_i)$ (given in Theorem 2).

Let us define $\lambda = \frac{\mathbb{P}_{\text{noise}}}{1 - \mathbb{P}_{\text{noise}}}$. By developing the expression (24) over the Q observations, we derive the explicit maximum likelihood distinguisher:

$$\begin{aligned}
\hat{k}_i &= \underset{k_i \in \{0,1\}}{\text{argmax}} \prod_{q=1}^{Q} \mathbb{E}_{k_i} \left(\lambda^{\delta_{X^q_{M_i} \neq y^q_{M_i}} + \delta_{X^q_{S_{i-1}} \neq y^q_{S_{i-1}}}} \right) \\
&= \underset{k_i \in \{0,1\}}{\text{argmax}} \sum_{q=1}^{Q} \ln \big(\lambda^2 \mathbb{P}(X_{M_i} = \neg y^q_{M_i}, X_{S_{i-1}} = \neg y^q_{S_{i-1}} | \mathcal{G}_i = k_i) \\
&\qquad + \lambda \mathbb{P}(X_{M_i} = \neg y^q_{M_i}, X_{S_{i-1}} = y^q_{S_{i-1}} | \mathcal{G}_i = k_i) \\
&\qquad + \lambda \mathbb{P}(X_{M_i} = y^q_{M_i}, X_{S_{i-1}} = \neg y^q_{S_{i-1}} | \mathcal{G}_i = k_i) \\
&\qquad + \mathbb{P}(X_{M_i} = y^q_{M_i}, X_{S_{i-1}} = y^q_{S_{i-1}} | \mathcal{G}_i = k_i) \big).
\end{aligned} \tag{25}$$

C.3 Summary

There are four kinds of attacker, depending whether p/R is known and depending whether $\mathbb{P}_{\text{noise}}$ is known. Notice that we can expect an attacker to profile $\mathbb{P}_{\text{noise}}$ using a public exponent. The suitable attacks are summarized in Tab. 4.

D Analysis of extra-reduction in OpenSSL and mbedTLS source code

The extra-reduction is explicit in the source code of OpenSSL, as shown in Listing 1.1.

		$\mathbb{P}(X_{M_i}, X_{S_{i-1}})$	
		known RSA-SFM, ECC	unknown RSA-CRT
$\mathbb{P}_{\text{noise}}$	unknown online	ρ-attack-Hard (Alg. 4.1 with (7)) (ρ-attack-Soft)	ρ-attack-Soft (Alg. 4.1 with (9))
	known offline	Maximum likelihood (25) (ρ-attack-Hard) (ρ-attack-Soft)	

Table 4. Most suitable attack for each scenario

Listing 1.1. Extra-reduction in OpenSSL code. File `crypto/bn/bn_mont.c`, function `BN_from_montgomery`

```
309 if (BN_ucmp(ret, &(mont->N)) >= 0)
310 {
311         if (!BN_usub(ret,ret,&(mont->N))) goto err;
312 }
```

The big number library of mbedTLS implements a protection against timing attacks. A subtraction is also carried out: it is either functional or dummy, as shown in Listing 1.2.

Listing 1.2. Extra-reduction in mbedTLS code. File `library/bignum.c`, function `mpi_montmul`

```
1500 if( mpi_cmp_abs( A, N ) >= 0 )
1501         mpi_sub_hlp( n, N->p, A->p );
1502 else
1503         /* prevent timing attacks */
1504         mpi_sub_hlp( n, A->p, T->p );
```

E Dependency between the operations of two consecutive iterations in Montgomery Ladder exponentiation

For the Montgomery Ladder (ML) algorithm, the M_i and S_{i-1} operations do not depend directly on the key bit value k_i. As we can see on the Fig. 11, the dependence comes from the bit value k_i and the previous bit value k_{i-1}. If the two bits value k_{i-1} and k_i are different then the output of multiplication M_i and the input of square S_{i-1} are equal ($\mathcal{G}_i = T$), otherwise these output/input are different ($\mathcal{G}_i = F$).

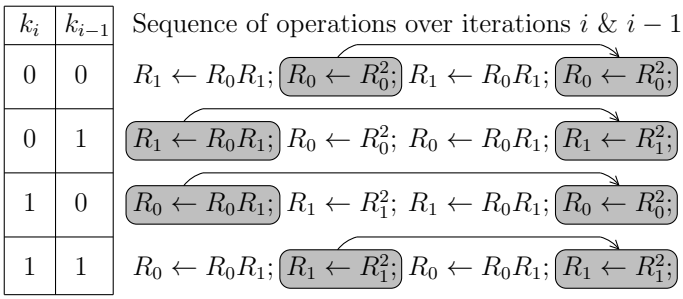


Fig. 11. Dependency between the consecutive operation and the consecutive key bit value in the Montgomery Ladder exponentiation algorithm (Alg. 1.2)