Improving the Big Mac attack on Elliptic Curve Cryptography

Jean-Luc Danger[1,2], Sylvain Guilley[1,2], Philippe Hoogvorst[2], Cédric Murdica[1,2], and David Naccache[3]

[1] Secure-IC S.A.S., 80 avenue des Buttes de Coësmes, F-35700 Rennes, France
{jean-luc.danger, sylvain.guilley, cedric.murdica}@secure-ic.com
[2] Département COMELEC, Institut TELECOM, TELECOM ParisTech, CNRS LTCI, Paris, France
{jean-luc.danger, sylvain.guilley, philippe.hoogvorst, cedric.murdica}@telecom-paristech.fr
[3] École normale supérieure, Département d'informatique 45, rue d'Ulm, F-75230, Paris Cedex 05, France.
david.naccache@ens.fr

Abstract. At CHES 2001, Walter introduced the Big Mac attack against an implementation of RSA. It is an horizontal collision attack, based on the detection of common operands in two multiplications. The attack is very powerful since one single power trace of an exponentiation permits to recover all bits of the secret exponent. Moreover, the attack works with unknown or blinded input. The technique was later studied and improved by Clavier *et alii* and presented at INDOCRYPT 2012. At SAC 2013, Bauer *et alii* presented the first attack based on the Big Mac principle on implementations based on elliptic curves with simulation results.
In this work, we improve the attack presented by Bauer *et alii* to considerably increase the success rate. Instead of comparing only two multiplications, the targeted implementation permits to compare many multiplications. We give experiment results with traces taken from a real target to prove the soundness of our attack. In fact, the experimental results show that the original Big Mac technique given by Walter was better that the technique given by Clavier *et alii*. With our experiments on a real target, we show that the theoretical improvements are not necessarily the more suitable methods depending on the targeted implementations.

Keywords: Elliptic Curve Cryptography, Side-Channel Attack, Big Mac Attack, Side-Channel Atomicity

1 Introduction

RSA and Elliptic Curve Cryptography (ECC) are vulnerable to side-channel attacks. Walter introduced at CHES 2001 the Big Mac attack on RSA [14]. It consists in comparing the power trace of two multiplications, and detect if they share a common operand. The Big Mac attack as presented in [14] is not applicable on ECC because the manipulated integers are too small. The size of the integers is an important factor for the success of the attack [14,5,1]. The Big Mac was then improved at INDOCRYPT 2012 for RSA implementations in [5]. Finally, in their publication at SAC 2013, Bauer et

al. were able to perform an improved Big Mac attack on ECC [2]. They target a particular implementation on ECC. The implementation uses a side-channel countermeasure called *Side-Channel Atomicity* [8,10]. In [2], the authors noticed a vulnerability in the Side-Channel Atomicity. If an attacker is able to detect if two different multiplications share a common operand, she can recover the scalar. They illustrated the soundness of their attack with simulation results.

In this paper, we extend the work of [2]. If the Side-Channel Atomicity is used, the attacker is able to compare many multiplications (precisely fourteen pairs) instead of only two. Moreover, we present experimental results on a real target. With our experimentation, it turns out that the method presented in the first place by Walter [14] works better (*in practice*) than the improved ones (*from a theoretical standpoint*) presented in [14,2].

The rest of the paper is organized as follows. In Section 2, we give the backgrounds on ECC. In Section 2.3, we recall on the Side-Channel Atomicity countermeasure, which brings protection on ECC against the Simple Power Analysis. Section 3 describes the Big Mac attack of Walter [14] and the improved ones of [5,2]. Our attack is presented in Section 4. Finally, we conclude in Section 6.

2 Elliptic Curve Cryptography

An elliptic curve over a finite prime field $\mathbb{F}_p$ of characteristic $p > 3$ can be described by its reduced Weierstraß form:

$$E\colon y^2 = x^3 + ax + b \ . \tag{1}$$

We denote by $E(\mathbb{F}_p)$ the set of points $(x, y) \in \mathbb{F}_p^2$ satisfying equation (1), plus the point at infinity $\mathcal{O}$.

$E(\mathbb{F}_p)$ is an additive abelian group defined by the following addition law. Let $P = (x_1, y_1) \neq \mathcal{O}$ and $Q = (x_2, y_2) \notin \{\mathcal{O}, -P\}$ be two points on $E(\mathbb{F}_p)$. Point addition $R = (x_3, y_3) = P + Q$ is defined by the formula:

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \\ y_3 &= \lambda(x_1 - x_3) - y_1 \end{aligned} \qquad \text{where } \lambda = \begin{cases} \frac{y_1 - y_2}{x_1 - x_2} & \text{if } P \neq Q, \\ \frac{3x_1^2 + a}{2y_1} & \text{if } P = Q. \end{cases}$$

The inverse of point P is defined as $-P = (x_1, -y_1)$.

ECC relies on the difficulty of the elliptic curve discrete logarithm problem (ECDLP, compute k given P and $Q = [k]P$) or on the hardness of related problems such as ECDH or ECDDH, which can be solved if ECDLP can be.

2.1 Jacobian Projective Arithmetic

To avoid costly divisions when using the formulæ previously described, projective or Jacobian are preferably used.

The equation of an elliptic curve in the Jacobian projective coordinates system in the reduced Weierstraß form is:

$$E^{\mathcal{J}}: Y^2 = X^3 + aXZ^4 + bZ^6 \ .$$

The projective point (X, Y, Z) corresponds to the affine point $(X/Z^2, Y/Z^3)$ and there is an equivalence relation between the points: the point (X, Y, Z) is equivalent to any point (r^2X, r^3Y, rZ) with $r \in \mathbb{F}_p^*$. The point at infinity is defined as $\mathcal{O} = (1, 1, 0)$ in Jacobian coordinates.

We give addition (ECADD) and doubling (ECDBL) formulas in the Jacobian projective coordinates system. Let $P_1 = (X_1, Y_1, Z_1)$ and $P_2 = (X_2, Y_2, Z_2)$ two points of $E^{\mathcal{J}}(\mathbb{K})$.

- **ECDBL**. $P_3 = (X_3, Y_3, Z_3) = 2P_1$ is computed as:
  $X_3 = T,\ Y_3 = -8Y_1^4 + M(S - T),\ Z_3 = 2Y_1Z_1,$
  $S = 4X_1Y_1^2,\ M = 3X_1^2 + aZ_1^4,\ T = -2S + M^2$
- **ECADD**. $P_3 = (X_3, Y_3, Z_3) = P_1 + P_2$ is computed as:
  $X_3 = -H^3 - 2U_1H^2 + R^2,\ Y_3 = -S_1H^3 + R(U_1H^2 - X_3),\ Z_3 = Z_1Z_2H,$
  $U_1 = X_1Z_2^2,\ U_2 = X_2Z_1^2,\ S_1 = Y_1Z_2^3,\ S_2 = Y_2Z_1^3,\ H = U_2 - U_1,\ R = S_2 - S_1$

For speeding up the doubling, Cohen et al. introduced the modified Jacobian coordinates [6]. A point P is represented by the coordinates (X, Y, Z, W) where X, Y, Z are the Jacobian coordinates of P and $W = aZ^4$. The doubling of the point $P_1 = (X_1, Y_1, Z_1, W_1)$ is given below.

- mod**ECDBL**. $P_3 = (X_3, Y_3, Z_3, W_3) = 2P_1$ is computed as:
  $X_3 = A^2 - 2C,\ Y_3 = A(C - X_3) - D,\ Z_3 = 2Y_1Z_1,\ W_3 = 2DW_1$
  $A = 3X_1^2 + W_1,\ B = 2Y_1^2,\ C = 2BX_1,\ D = 2B^2$

Remark 1. We summarize in this remark the conventional use of indices for field variables names in ECC operations. The inputs of **ECDBL** and **ECADD**, namely variables X, Y, Z in Jacobian coordinates (X, Y, Z, W in modified Jacobian coordinates), have indices 1 and 2. Of course, for **ECDBL**, indices 2 are not used. Index 3 is reserved for the **ECDBL**, **ECADD** and mod**ECDBL** outputs.

The indices used in the other (temporary) variables simply serve to uniquify them.

2.2 Elliptic Curve Scalar Multiplication

In ECC applications, one has to compute scalar multiplications (ECSMs), *i.e.* compute $[k]P$, given P and an integer k. Several methods exist to perform such a computation. This study focuses on the *Right-to-Left binary NAF mixed coordinates multiplication* [11]. Indeed, the countermeasure that we target was presented on this ECSM.

Algorithm 1 Right-to-Left binary NAF multiplication using mixed coordinates [11]

Input: $k, P = (X, Y, Z)$
Output: $[k]P$
  $(X_1, Y_1, Z_1) \leftarrow \mathcal{O}$
  $(T_1, T_2, T_3, T_4) \leftarrow (X, Y, Z, aZ^4)$
  while $k \geq 1$ **do**
    if $k_0 = 1$ **then**
      $u \leftarrow 2 - (k \bmod 4)$
      $k \leftarrow k - u$
      if $u = 1$ **then**
        $(X_1, Y_1, Z_1) \leftarrow \text{ECADD}((X_1, Y_1, Z_1), (T_1, T_2, T_3))$
      else
        $(X_1, Y_1, Z_1) \leftarrow \text{ECADD}((X_1, Y_1, Z_1), (T_1, -T_2, T_3))$
      end if
    end if
    $k \leftarrow k/2$
    $(T_1, T_2, T_3, T_4) \leftarrow \text{modECDBL}(T_1, T_2, T_3, T_4)$
  end while
  $(X_1, Y_1, Z_1) \leftarrow \text{ECADD}((X_1, Y_1, Z_1), (T_1, T_2, T_3))$
  return (X_1, Y_1, Z_1)

2.3 Side-Channel Atomicity

Naive ECSM, such as the Right-to-Left binary NAF mixed coordinates multiplication (Algorithm 1), is vulnerable to the Simple Power Analysis [7]. Indeed, the field operations involved for a doubling or an addition are quite different. Using the power trace of the ECSM, an attacker can detect which operation (doubling or addition of points) is performed and therefore deduce the scalar with a single trace.

Chevallier-Mames, Ciet and Joye introduced the concept of *side-channel atomicity* [8]. The formulæ to perform a doubling and an addition are rewritten into sequences of identical *atomic patterns*.

It was later improved by Giraud and Verneuil for ECADD and modECDBL for the Right-to-Left binary NAF mixed coordinates multiplication [10]. Figure 1 describes the computation of $\text{ECADD}((X_2, Y_2, Z_2), (X_1, Y_1, Z_1))$ and $\text{modECDBL}(X_1, Y_1, Z_1, W_1)$ (see [10] for the details). Each column represents an atomic pattern. The addition is written with two patterns while the doubling is written with only one.

This implementation is not vulnerable to SPA anymore since the attacker cannot distinguish between the operations performed simply by regarding the power consumption trace during the execution of the scalar multiplication.

	ECADD - part 1 ($\mathcal{A}1$)	ECADD - part 2 ($\mathcal{A}2$)	modECDBL ($\mathcal{D}$)
1.	$T_1 \leftarrow Z_2^2$	$T_1 \leftarrow T_6^2$	$T_1 \leftarrow X_1^2$
2.	$\star$	$\star$	$T_2 \leftarrow Y_1 + Y_1$
3.	$T_2 \leftarrow Y_1 \times Z_2$	$T_4 \leftarrow T_5 \times T_1$	$Z_3 \leftarrow T_2 \times Z_1$
4.	$\star$	$\star$	$T_4 \leftarrow T_1 + T_1$
5.	$T_5 \leftarrow Y_2 \times Z_1$	$T_5 \leftarrow T_1 \times T_6$	$T_3 \leftarrow T_2 \times Y_1$
6.	$\star$	$\star$	$T_6 \leftarrow T_3 + T_3$
7.	$T_3 \leftarrow T_1 \times T_2$	$T_1 \leftarrow Z_1 \times T_6$	$T_2 \leftarrow T_6 \times T_3$
8.	$\star$	$\star$	$T_1 \leftarrow T_4 + T_1$
9.	$\star$	$\star$	$T_1 \leftarrow T_1 + W_1$
10.	$T_4 \leftarrow Z_1^2$	$T_6 \leftarrow T_2^2$	$T_3 \leftarrow T_1^2$
11.	$T_5 \leftarrow T_5 \times T_4$	$Z_3 \leftarrow T_1 \times Z_2$	$T_4 \leftarrow T_6 \times X_1$
12.	$\star$	$T_1 \leftarrow T_4 + T_4$	$T_5 \leftarrow W_1 + W_1$
13.	$T_2 \leftarrow T_2 - T_3$	$T_6 \leftarrow T_6 - T_1$	$T_3 \leftarrow T_3 - T_4$
14.	$T_5 \leftarrow T_1 \times X_1$	$T_1 \leftarrow T_5 \times T_3$	$W_3 \leftarrow T_2 \times T_5$
15.	$\star$	$X_3 \leftarrow T_6 - T_5$	$X_3 \leftarrow T_3 - T_4$
16.	$\star$	$T_4 \leftarrow T_4 - X_3$	$T_6 \leftarrow T_4 - X_3$
17.	$T_6 \leftarrow X_2 \times T_4$	$T_3 \leftarrow T_4 \times T_2$	$T_4 \leftarrow T_6 \times T_1$
18.	$T_6 \leftarrow T_6 - T_5$	$Y_3 \leftarrow T_3 - T_1$	$Y_3 \leftarrow T_4 - T_2$

Fig. 1. ECADD and modECDBL operations written with the same atomic pattern ($\star$ represents a dummy operation)

3 Big Mac Attack

3.1 Big Mac Attack on RSA

We present in this section the Big Mac Attack introduced by Walter against RSA implementations [14].

Long Integer Multiplication. We give in Alg. 2 the classical field multiplication. w is the word size (w is generally equal to $8, 16, 32$ or 64 in common architectures).

Algorithm 2 Long Integer Multiplication

Input: $A = (a_{m-1}, \ldots, a_0)_{2^w}, B = (b_{m-1}, \ldots, b_0)_{2^w}$
Output: $C = (c_{2m-1}, \ldots, c_0)_{2^w} = A \times B$
1: $C \leftarrow 0$
2: **for** $i = 0$ **to** $m - 1$ **do**
3: $\quad u \leftarrow 0$
4: $\quad$ **for** $j = 0$ **to** $m - 1$ **do**
5: $\quad\quad (u, v)_{2^w} \leftarrow a_i \times b_j$
6: $\quad\quad (u, v)_{2^w} \leftarrow (u, v)_{2^w} + c_{i+j} + u$
7: $\quad\quad c_{i+j} \leftarrow v$
8: $\quad$ **end for**
9: $\quad c_{i+m} \leftarrow u$
10: **end for**
11: **return** S

Modular multiplication is performed either with the classical modular multiplication followed by a reduction, like the Montgomery [13] or the Barrett reduction [3], or with an interleaved modular multiplication. The important feature for the attack is the word-wise multiplication.

Goal of the attack. Denote T_1, T_2 the traces during the computation of respectively two multiplications $A \times B$, $C \times D$, with $A \neq C$. The attacker tries to assert if $B = D$ given T_1 and T_2.

Averaging. We suppose that the device leaks the Hamming Weight (denoted **HW**) of the manipulated values.

The power consumption during the computation of $a_i \times b_j$ (line 5 of Algorithm 2) can be expressed with $\mathbf{HW}(b_j)$, and other activities of the device (including $\mathbf{HW}(a_i)$, $\mathbf{HW}(a_i \times b_j)$) and the noise.

$$s_{i,j} = \mathbf{HW}(b_j) + r_{i,j} \ . \tag{2}$$

with $r_{i,j}$ corresponds to other activities and the noise. The sample points of the trace T_1, in which each $b_j, j \in [0, m[$ is manipulated, are averaged into one single value s_j.

$$s_j = \sum_{0}^{m-1} s_{i,j} \tag{3}$$

$$= \mathbf{HW}(b_j) + r_j \tag{4}$$

with r_j having a much smaller value compared with each $r_{i,j}$. The computation of s_0 is illustrated in Figure 2.

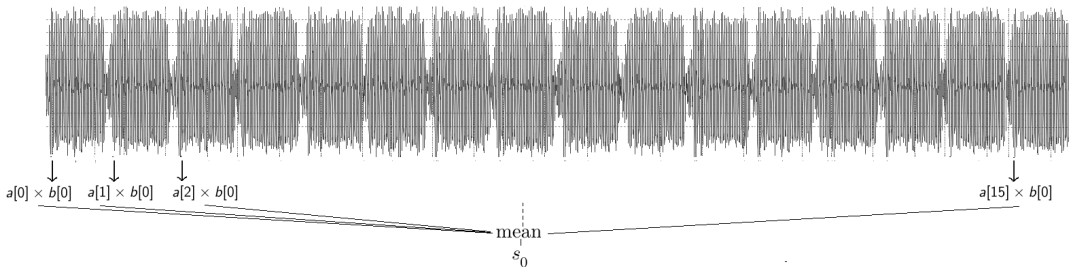

Fig. 2. Illustration of the computation of s_0 with a modular multiplication of integers of four words (256-bit integers in a 64-bit architecture)

Euclidean Distance. Denote $S_1 = s_0 || \ldots || s_{m-1}$ the concatenation of the s_j. The same is done with T_2 to obtain S_2.

If $B = D$, the Euclidean distance between S_1 and S_2 is small. In the case of $B \neq D$, the distance is high.

Big Mac CoCo. Instead of using a Euclidean Distance, the authors of [5] suggest to use the Pearson correlation instead of the Euclidean Distance. This refined attack is called *Big Mac CoCo* (CoCo for collision-correlation) in[5].

Comparison between Big Mac and Big Mac CoCo. They give simulation results to compare the Euclidean Distance with the Pearson correlation. The Big Mac CoCo gives much better results than the original Big Mac of Walter.

3.2 Big Mac Attack on ECC

The classical Big Mac of Walter is considered not applicable on ECC because the number of words is large compared to ECC[4].

However, Bauer et al. give simulation results of the Big Mac CoCo on elliptic curves size [2]. They target the Side-Channel Atomicity. Indeed, they notice that there are common operands regarding the side-channel atomicity formulæ. For instance, to distinguish an addition from a doubling, they suggest to compare the first multiplication (line 1) and the second multiplication (line 3) of Figure 1. If it is a doubling, the two multiplications share a common operand. They give the success rate on simulation results using a correlation which was high enough even for a 32 architecture.

We experimentally tried both the Big Mac and the Big Mac CoCo on real measurements on a 64 bits architecture and we failed. In the next section, we present a significant improvement of the attack of [2]. We also present experimental results of our attack.

4 Improving the Big Mac Attack on the Side-Channel Atomicity

We describe in this section our attack. Instead of trying to differentiate between elliptic curves operations (addition or doubling) and only two patterns, we will analyses a sequence of several patterns depending of a bit of the scalar. The attack is recursive. For a better clarity, we will see how to recover the first bit of the scalar. The next bits are recover in the same way.

The core idea of the attack is to identify which operations are performed by analysing the possible repetitions of variables in the patterns.

4.1 Possibilities of the atomic patterns

For the first iteration of algorithm 1, the possible operations of the three first atomic patterns are:

1. $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$. In this case, $k_0 \neq 0$.

[4] For a 128 bits security, ECC must use 256-bit integers length, while RSA must use 3072-bit integers.

2. $\mathcal{D}; \mathcal{A}1; \mathcal{A}2$. In this case, $k_0 = 0$.
3. $\mathcal{D}; \mathcal{D}; \mathcal{A}1$. In this case, $k_0 = 0$.
4. $\mathcal{D}; \mathcal{D}; \mathcal{D}$. In this case, $k_0 = 0$.

More precisely, the four cases are for the three first bits of k equal to $1xx$, $01x$, 001 and 000 respectively, where x represents any value (0 or 1). We want to assert if the first three patterns correspond to $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$ ($k_0 \neq 0$).

4.2 Same values in the different patterns.

With Figure 1 and the different possibilities of the three first patterns, we label the operations with a common operand *only* if the operations are $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$; we neglect the multiplications sharing a common operand if they possibly occur in another sequence of patterns.

The common operands are illustrated in Figure 3. They are denoted with boxes with the same index. For example, the square at line 1 of the 1^{st} pattern and the multiplication at line 3 of the 1^{st} pattern share a common operand (Z_2) only if the sequence is $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$. Note that the multiplication at line 17 of the 1^{st} pattern and the multiplication at line 11 of the 3^{rd} pattern share a common operand (X_2 and X_1) only if $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$ is performed. The same holds for Z_2 in $\mathcal{A}1; \mathcal{A}2$ and Z_1 in $\mathcal{D}$. Indeed, the point (X_2, Y_2, Z_2) of $\mathcal{A}1; \mathcal{A}2$ and the point (X_1, Y_1, Z_1) of $\mathcal{D}$ both correspond to the point R or $-R$ in Algorithm 1.

The total number of pairs of multiplications or squares sharing a common operand is sixteen in the sequence $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$.

4.3 Assembling the pieces of the puzzle

We want to apply the method of the Big Mac attack to detect if the three first patterns indeed correspond to $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$. The low number of words is compensated by the large number of modular multiplications we compare. We can compare sixteen pairs (see Figure 3) instead of one, thanks to the atomicity countermeasure.

First, we split the trace of the three first patterns; we separate the field operations. We denote $s(\cdot)$ the method for constructing S_1 or S_2 as previously described for the Big Mac attack.

We then construct two sets U_1, U_2 as follows. U_1, U_2 are first set empty. We perform $s(\cdot)$ for the power traces of the multiplications that might share a common operand. One element of each pair is put in U_1, the other is put in U_2. The construction of U_1, U_2 is illustrated in Figure 4 for the first three pairs possibly sharing the same operand Z_2.

The Euclidean distance between U_1 and U_2 is low if each pair share a common operand. In this case the three patterns observed are actually $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$, and the attacker concludes that $k_0 \neq 0$. She then iterates the method with the next three patterns to target the digit k_1. The Euclidean distance between U_1 and U_2 is high if

	ECADD - part 1 ($\mathcal{A}1$)	ECADD - part 2 ($\mathcal{A}2$)	modECDBL ($\mathcal{D}$)
1.	$T_1 \leftarrow \boxed{Z_2}^2_{1,2,14}$	$T_1 \leftarrow \boxed{T_6}^2_{9,10}$	$T_1 \leftarrow \boxed{X_1}^2_{12}$
2.	$\star$	$\star$	$T_2 \leftarrow Y_1 + Y_1$
3.	$T_2 \leftarrow Y_1 \times \boxed{Z_2}_{1,3,15}$	$T_4 \leftarrow T_5 \times T_1$	$Z_3 \leftarrow T_2 \times \boxed{Z_1}_{14,15,16}$
4.	$\star$	$\star$	$T_4 \leftarrow T_1 + T_1$
5.	$T_5 \leftarrow Y_2 \times \boxed{Z_1}_{4,5}$	$T_5 \leftarrow T_1 \times \boxed{T_6}_{9,11}$	$T_3 \leftarrow T_2 \times Y_1$
6.	$\star$	$\star$	$T_6 \leftarrow T_3 + T_3$
7.	$T_3 \leftarrow \boxed{T_1}_7 \times T_2$	$T_1 \leftarrow \boxed{Z_1}_{5,6} \times \boxed{T_6}_{10,11}$	$T_2 \leftarrow T_6 \times T_3$
8.	$\star$	$\star$	$T_1 \leftarrow T_4 + T_1$
9.	$\star$	$\star$	$T_1 \leftarrow T_1 + W_1$
10.	$T_4 \leftarrow \boxed{Z_1}^2_{4,6}$	$T_6 \leftarrow T_2^2$	$T_3 \leftarrow T_1^2$
11.	$T_5 \leftarrow T_5 \times \boxed{T_4}_8$	$Z_3 \leftarrow T_1 \times \boxed{Z_2}_{2,3,16}$	$T_4 \leftarrow T_6 \times \boxed{X_1}_{13}$
12.	$\star$	$T_1 \leftarrow T_4 + T_4$	$T_5 \leftarrow W_1 + W_1$
13.	$T_2 \leftarrow T_2 - T_3$	$T_6 \leftarrow T_6 - T_1$	$T_3 \leftarrow T_3 - T_4$
14.	$T_5 \leftarrow \boxed{T_1}_7 \times X_1$	$T_1 \leftarrow T_5 \times T_3$	$W_3 \leftarrow T_2 \times T_5$
15.	$\star$	$X_3 \leftarrow T_6 - T_5$	$X_3 \leftarrow T_3 - T_4$
16.	$\star$	$T_4 \leftarrow T_4 - X_3$	$T_6 \leftarrow T_4 - X_3$
17.	$T_6 \leftarrow \boxed{X_2}_{12,13} \times \boxed{T_4}_8$	$T_3 \leftarrow T_4 \times T_2$	$T_4 \leftarrow T_6 \times T_1$
18.	$T_6 \leftarrow T_6 - T_5$	$Y_3 \leftarrow T_3 - T_1$	$Y_3 \leftarrow T_4 - T_2$

Fig. 3. Common operands in the atomic patterns

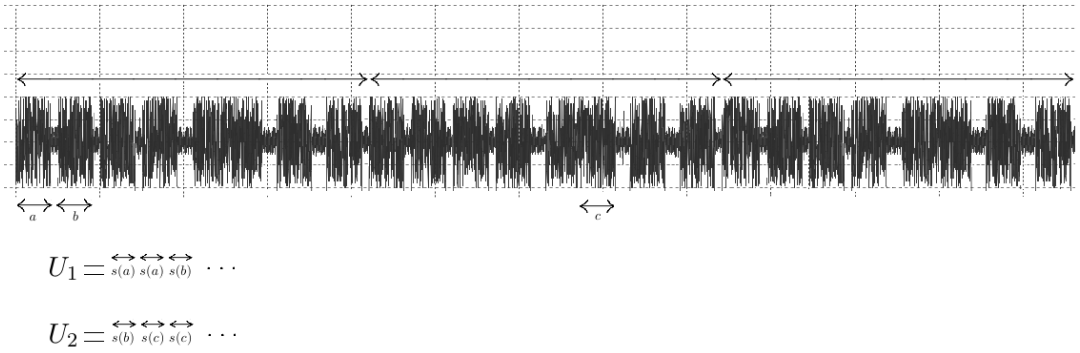

Fig. 4. Assembling the pieces of the puzzle of three atomic patterns

no multiplication among all multiplications shares a common operand. In this case, the three patterns observed are not $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$, and the attacker concludes that $k_0 = 0$. She starts again with the two last patterns of the three, added together with the fourth pattern of the ECSM to target k_1.

4.4 Experimental Results

We implemented a modular multiplication on a 64-bit architecture in the Side-channel Attack Standard Evaluation Board SASEBO-GII [SASEBO]. We mounted the attack with 384-bit integers (six words of 64 bits).

Characterization. The first step of the attack is the characterisation of the arithmetic module. We constructed U_1, U_2 as previously described with fourteen pairs of multiplications sharing a common operand[5] 1000 times. The average Euclidean distance was 2.165. The same was done with fourteen pairs of multiplication with random operands. The average Euclidean distance was 3.198. We established that a distance lower than the mean 2.682 correspond to $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$.

Attack on real operations. We then assembled the pieces of the puzzle as previously described with a trace of $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$ 100 times. Only three distances were higher than 2.682. We conclude that the attacker can detect $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$ with a success of 97%. The same was done with $\mathcal{D}; \mathcal{D}; \mathcal{A}1$ 100 times. Only four distances were lower than 2.682. We conclude that the attacker wrongly detects a patterns triplet as $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$ with probability 4%.

We performed the experiment with 256-bit integers (four words) as well. We obtained a probability of 96% to correctly detect $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$, and a probability of 16% that $\mathcal{D}; \mathcal{D}; \mathcal{A}1$ was detected as $\mathcal{A}1; \mathcal{A}2; \mathcal{D}$, which is still acceptable to perform the attack.

We believe that the success probability is higher on a 32-bit architecture because of the larger number of words.

Big Mac CoCo. We also tried using the Pearson correlation as in [5,2]. Surprisingly, the coefficient was high (around 0.9) each time, even if the guess was incorrect (i.e. even if there are no common operand for all multiplications).

The reason is that there are similarities in long integer multiplications even if the values are different such as the variation of the word numbers manipulated. Our experiment shows that in certain cases, the Euclidean Distance is better than the correlation.

5 Countermeasures

In this section, we discuss on the classical countermeasures on ECC that thwart or not our attack.

[5] We use fourteen pairs instead of sixteen as shown in Figure 3 because we avoid the pairs where the possibly same operand is not in the same side: boxes 5 and 13.

5.1 Ineffective Countermeasures

Scalar Randomization [7, §5.1]. If k is the scalar and $P \in E$ the base point, Coron suggests to randomize the scalar as $k' = k + r\#E$ with $\#E$ the number of points in the curve and r a random integer. This prevents from the Differential Power Analysis [7]. When applying our attack, the attacker recovers k' and trivially recovers the original previous secret $k = k' \bmod \#E$.

Scalar Splitting [9, §4.2]. Clavier and Joye proposed a method to randomize the scalar. Instead of computing $[k]P$, one can compute $Q = [k-r]P+[r]P$ with a random r. If the two scalar multiplications are performed successively, the attack presented in this paper can trivially be applied for both ECSMs and recover the initial scalar k. On the other hand, if the two ECSMs are performed in parallel, it is quite more difficult. Indeed, when attacking the scalar of one ECSM, the power consumption or the electromagnetic radiation coming from the second ECSM is necessarily considered as noise. To conclude, the attack can still be applied in theory but the success rate should be decreased considerably.

Point Binding [7, §5.2]. The countermeasure, by Coron, consists in computing $Q = [k](P + R)$ instead of $[k]P$, with R a pseudo-random point. The chip returns $Q - [k]R$. Our attack does not need the knowledge of the base point and therefore the countermeasure is ineffective. We focus on possible collisions of values that will happen even if the base point is randomized this way.

Random Projective Coordinates [7, §5.3]. A point $P = (X, Y, Z)$ in Jacobian coordinates is equivalent to any point (r^2X, r^3Y, rZ), with $r \in \mathbb{F}_p^*$. Coron suggests to randomize the base point at the beginning of the ECSM by choosing a random nonzero r. The previous analysis on the point blinding stands here.

Random Curve Isomorphism [12]. Elliptic curves $E\colon y^2 = x^3 + ax + b$ and $E'\colon y^2 = x^3 + a'x + b'$ are isomorphic if and only if there exists $u \in \mathbb{F}_p^*$ such that $u^4a' = a$ and $u^6b' = b$. The isomorphism φ is defined as:

$$\varphi\colon E \xrightarrow{\sim} E', \begin{cases} \mathcal{O} \to \mathcal{O} \\ (x, y) \to (u^{-2}x, u^{-3}y) \end{cases}$$

The countermeasure, introduced by Joye and Tymen, consists in computing the ECSM on a random curve E' instead of E. The previous analysis on the point blinding stands here.

5.2 Effective Countermeasures

Multiplication with Random Permutation [CFG+10,1]. Clavier, Feix, Gagnerot, Rousselet and Verneuil introduced the Multiplication with Random Permutation countermeasure [CFG+10]. It consists in randomizing the order of the manipulation of the words during a long multiplication. For example, in Algorithm 2, it consists

in randomizing the order of both loops (lines 2 and 4) with two random permutations in $[0, m[$ (m being the word number of the manipulated integers). The construction of $s_0, 0 \leq j < m$ is no longer possible for the Big Mac attack. Another method for randomizing the loops was proposed in [1].

No Same Values Algorithm. We suggest to implement elliptic curves without the possible repetitions of values depending on the scalar. The side-channel atomicity brings too much multiplications that we can compare. That makes our attack possible and practicable.

Other countermeasures exist to prevent the Simple Power Analysis aside the Side-Channel Atomicity. Regularizing the ECSM, *i.e.* perform the same elliptic curve operations at each iteration of the ECSM prevents the Simple Power Analysis without bringing many multiplications that possibly share common operands.

6 Conclusion

A practical horizontal attack on ECC is presented against the Side-Channel Atomicity countermeasure, based on the Big Mac principle [14]. It is an extension of the attack presented in [2]. The difference is that we compare many multiplications instead of only one. The Side-Channel Atomicity permits to compare many multiplications.

This attack is powerful since it permits to recover the entire scalar with a single trace. The secret scalar can thus be recovered with a single execution of the ECSM and we can target protocol such as ECDSA where the scalar is randomly chosen for each new signature. Also, scalar randomization techniques are ineffective.

Moreover, the base point does not matter for the attack. Therefore, countermeasures which consist in randomizing the inputs are ineffective.

To prove the soundness of our attack, we give experimental results. We emphasis in the fact that the correlation used as a distinguisher is not the optimal solution in our case (in fact we failed) as presented in [5,2]. The Euclidean Distance, as presented in the original Big Mac attack [14].

We target a particular countermeasure which has the particularity to bring multiple possible common operands during the elliptic curve operations (addition and doubling). However, we believe that the method might be adapted on other implementations, where many modular multiplications can be compare. This is not the case for classical implementations of ECC (with classical addition and doubling formulæ) but might the case for other specific implementations.

References

1. A. Bauer, É. Jaulmes, E. Prouff and J. Wild, *Horizontal and Vertical Side-Channel Attacks against Secure* RSA *Implementations*. Proceedings of CT-RSA'13, LNCS vol. 7779, Springer-Verlag, 2013, pp. 1-17.
2. A. Bauer, É. Jaulmes, E. Prouff and J. Wild, *Horizontal Collision Correlation Attack on Elliptic Curves*. To appear in SAC'13.
3. P. Barrett, *Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor*. Proceedings of CRYPTO'86, LNCS vol. 263, Springer-Verlag, 1987, pp. 311-323.
4. M. Ciet and M. Joye, *(Virtually) Free Randomization Techniques for Elliptic Curve Cryptography*. Proceedings of ICIS'03, LNCS vol. 2836, Springer, 2003, pp. 348-359.
5. C. Clavier, B. Feix, G. Gagnerot, C. Giraud, M. Roussellet and V. Verneuil, *ROSETTA for Single Trace Analysis*. Proceedings of INDOCRYPT'12, LNCS vol. 7668, Springer-Verlag, 2012, pp. 140-155.

[CFG+10] C. Clavier, B. Feix, G. Gagnerot, M. Roussellet and V. Verneuil, *Horizontal Correlation Analysis on Exponentiation*. Proceedings of ICICS'10, LNCS vol. 6476, Springer, 2010, pp. 46-61.

6. H. Cohen, A. Miyaji and T. Ono, *Efficient Elliptic Curve Exponentiation Using Mixed Coordinates*. Proceedings of ASIACRYPT'98, LNCS vol. 1514, Springer-Verlag, 1998, pp. 51-65.
7. J.-S. Coron, *Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems*. Proceedings of CHES'99, LNCS vol. 1717, Springer-Verlag, 1999, pp. 292-302.
8. B. Chevallier-Mames, M. Ciet and M. Joye, *Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity*. Journal of IEEE Transactions on Computers'04, IEEE TRANS. COMPUTERS vol. 53, num. 6, 2004, pp. 460-468.
9. C. Clavier and M. Joye, *Universal Exponentiation Algorithm*. Proceedings of CHES'01, LNCS vol. 2162, Springer, 2001, pp. 300-308.
10. C. Giraud and V. Verneuil, *Atomicity Improvement for Elliptic Curve Scalar Multiplication*. Proceedings of CARDIS'10, LNCS vol. 6035, Springer-Verlag, 2010, pp. 80-101.
11. M. Joye, *Fast Point Multiplication on Elliptic Curves without Precomputation*. Proceedings of WAIFI'08, LNCS vol. 5130, Springer-Verlag, 2008, pp. 36-46.
12. M. Joye and C. Tymen, *Protections against Differential Analysis for Elliptic Curve Cryptography*. Proceedings of CHES'01, LNCS vol. 2162, Springer, 2001, pp. 377-390.
13. P. L. Montgomery, *Modular Multiplication without Trial Division*. Journal of Mathematics of Computation'85, J. MATHEMATICS OF COMPUTATION vol. 44, num. 170, 1985, pp 519-521.

[SASEBO] *Side-channel Attack Standard Evaluation Board (SASEBO)*. `http://www.rcis.aist.go.jp/special/SASEBO/`

14. C. D. Walter, *Sliding Windows Succumbs to Big Mac Attack*. Proceedings of CHES'01, LNCS vol. 2162, Springer-Verlag, 2001, pp. 286-299.