

## Edwards Curve and BIG MAC Attack

Aubain Jose MAYEUKU<sup>1\*</sup>, Emmanuel FOUOTSA<sup>2,3</sup>, Celestin LELE<sup>1</sup>

<sup>1\*</sup> Departement Of Mathematics and Computer Sciences, University of Dschang, P.O.box 67, Dschang, Cameroon.

<sup>2</sup> Departement of Mathematics, Higher Teacher Training College, The University of Bamenda, P.O.Box 39, Bambili, Cameroon.

<sup>3</sup> Center for Cybersecurity and Mathematical Cryptology, The University of Bamenda., P.O. Box 39, Bambili, Cameroon.

\* **Correspondence:** Aubain Jose MAYEUKU

*The authors declare that no funding was received for this work.*



Received: 10-November-2025

Accepted: 12-December-2025

Published: 19-January-2026

**Copyright** © 2026, Authors retain copyright. Licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. <https://creativecommons.org/licenses/by/4.0/> (CC BY 4.0 deed)

This article is published in the **MSI Journal of Multidisciplinary Research (MSIJMR)** ISSN 3049-0669 (Online)

The journal is managed and published by MSI Publishers.

**Volume: 3, Issue: 1 (January-2026)**

**ABSTRACT:** In 2001, Walter introduced the BIG MAC attack on RSA systems. This horizontal collision attack exploits the detection of common operands between two multiplications. It is highly effective, as a single power consumption trace is sufficient to recover all the bits of the secret exponent. Initially, the BIG MAC attack was not applicable to cryptosystems based on elliptic curves. It was only in 2013 that Bauer et al. enhanced the attack and proposed a version adapted to elliptic curves. This new approach specifically targets the atomicity countermeasure and relies on identifying common operands between two multiplications within the addition and doubling algorithms for elliptic curve points. In 2016, Danger et al. further refined the attack by significantly improving its efficiency: instead of comparing only two multiplications, their method compares sixteen, thereby achieving much better results. In this work, we first analyze the atomicity countermeasure applied to Edwards curves, Twisted Edwards curves, and Edwards curves defined over binary fields. Then, we study the BIG MAC attack on these

curves, focusing on its effectiveness against atomicity-based countermeasures.

**Keywords:** *Elliptic Curve Cryptography, Big Mac Attack, Side Channel Atomicity, Edwards curves.*

## 1. Introduction

Point scalar multiplication is a fundamental component of elliptic curve cryptography. This cryptographic approach is increasingly popular and widely recommended due to its key advantage: the use of very small key sizes. However, cryptographic protocols based on elliptic curves are vulnerable to side-channel attacks. These attacks exploit information such as electromagnetic emissions, variations in current consumption, or power usage during the execution of scalar multiplication algorithms, aiming to extract details about the secret key.

Among side-channel attacks, we can notably mention SPA (Simple Power Analysis). This attack exploits the significant differences in power consumption between point addition and point doubling during the execution of scalar multiplication algorithms. By observing a single power consumption trace, the attacker can reconstruct the sequence of additions and doublings performed during scalar multiplication, which may allow them to recover the secret key. Several countermeasures have been proposed to mitigate this attack, including unified formulas and atomicity formulas [[7], [3]]. These approaches involve rewriting point addition and doubling operations as identical sequences of addition and multiplication operations over the field. This prevents distinguishing between addition and doubling operations solely by observing power consumption traces.

However, even with the use of atomicity, elliptic curves remain vulnerable to a Side-Channel Attack that exploits this countermeasure: the BIG MAC attack. Introduced in 2001 by Walter for RSA [8], this attack involves observing the power consumption traces of two multiplications and detecting whether these multiplications share a common operand. Initially, this attack was not applicable to elliptic curves. In 2013, Bauer et al. successfully enhanced the BIG MAC attack and applied it to elliptic curves [4]. They targeted the atomicity countermeasure and demonstrated that if an attacker can identify multiplications sharing common operands, they can recover the

secret key used in point scalar multiplication. In 2016, J.-L. Danger et al. further refined the attack, focusing on comparing 16 multiplications instead of just 2 [2].

Yusuke Takemura et al. proposed a new atomicity formula effective for binary Weierstrass elliptic curves and showed the vulnerability of this countermeasure to the BIG MAC attack [5].

In this work, we first analyze the atomicity countermeasure applied to Edwards curves, Twisted Edwards curves, and Edwards curves defined over binary fields. Then, we study the BIG MAC attack on these curves, focusing on its effectiveness against atomicity-based countermeasures. The rest of paper is organised as follows: In Section 2, we review the properties of Edwards curves, focusing specifically on the addition formulas in affine and projective coordinates. We propose rewriting this addition and pointing out doubling formulas as uniform sequences of scalar operations to implement the atomicity countermeasure. We also revisit the principle of the BIG MAC attack on elliptic curves. In Section 3, we analyze the BIG MAC attack on Edwards curves. In Section 4, we provide countermeasures against the BIG MAC attack and conclude with a summary.

## 2 Elliptic Curves Cryptography

### 2.1 Edwards Curve

Let  $p > 3$  be a prime number and  $K = \mathbb{F}_p$  be a field. The Edwards form of an elliptic curve over a field  $\mathbb{F}_p$  is given by the equation  $E : x^2 + y^2 = c^2(1 + x^2y^2)$ , where  $c, \in \mathbb{F}_p$ . The set of points satisfying the equation of  $E$  form an abelian group, the neutral element of this is the point  $(0, c)$ . We will denote that group as  $E(K)$ . Let  $P_1 = (x_1, y_1)$  and  $P_2 = (x_2, y_2)$  be two points of  $E(K)$ ,  $P_1 + P_2$  is computed as follows:

$$P_1 + P_2 = \left( \frac{x_1y_2 + y_1x_2}{c(1 + dx_1x_2y_1y_2)}; \frac{y_1y_2 - x_1x_2}{c(1 - dx_1x_2y_1y_2)} \right)$$

The above addition formulae contain finite field inversions which are costly and can be avoided by changing the affine coordinates to other systems such as projective coordinates.

Addition In Projective Coordinates: Let  $P_1 = (x_1, y_1)$  and  $P_2 = (x_2, y_2)$  be two points of  $E(K)$ , in projective coordinates,  $P_1 = (X_1; Y_1; Z_1)$ ,  $P_2 = (X_2; Y_2; Z_2)$ , where  $X_1 = x_1Z_1$ ,  $Y_1 = y_1Z_1$ ,  $X_2 = x_2Z_2$ ,  $Y_2 = y_2Z_2$ ,  $Z_1, Z_2 \in K$ .  $(X_3; Y_3; Z_3) =$

$P_1 + P_2$  is computed as follow:

$$A = Z_1Z_2, B = X_2, C = X_1X_2, D = Y_1Y_2.$$

$$E = dCD, F = B - E, G = B + E,$$

$$X_3 = AF ((X_1 + Y_1)(X_2 + Y_2) - C - D)$$

$$Y_3 = AG(D - C)$$

$$, Z_3 = cFG$$

Doubling In Projective Coordinates: Let  $P_1 = (X_1; Y_1; Z_1)$  be a point in projective coordinates over the curve  $E$ .  $(X_3; Y_3; Z_3) = 2P_1$  is computed as follow:

$$B = (X_1 + Y_1)^2, C = X_1^2, D = Y_1^2$$

$$E = C + D, H = cZ_1^2, J = E - 2HX_3 = c(B - E)J$$

$$Y_3 = cE(C - D)$$

$$Z_3 = EJ$$

## 2.2 Twisted Edwards curve

The Twisted Edwards form of an elliptic curve over  $K$  is described as:

$$ET : ax^2 + y^2 = 1 + dx^2y^2 \quad (a, d \in K, ad(a - d) \neq 0)$$

The neutral element is the affine point  $(0, 1)$ . The inverse of a point  $(x_1, \sqrt{y_1})$  is the point  $(x_1, -\sqrt{y_1})$ . The point  $(0, -1)$  is a point of order 2 and the point  $(+1/a, 0)$  is a point of order 4.

The set of points  $P = (x, y)$  satisfy  $E_T$ , is denoted by  $E_T(K)$ , which forms an abelian group. Let  $P_1 = (x_1, y_1)$  and  $P_2 = (x_2, y_2)$  be two points on  $E_T(K)$ , the sum  $P_3 = P_1 + P_2 = (x_3, y_3)$  can be computed as follows:

$$(x_3, y_3) = \left( \frac{x_1 y_2 + x_2 y_1}{1 + dx_1 x_2 y_1 y_2}, \frac{y_1 y_2 - ax_1 x_2}{1 - dx_1 x_2 y_1 y_2} \right)$$

Addition point on Twisted Edwards curve is unified, that means we can use the same addition formulas for point doubling. On Twisted Edwards curve model, there are three different coordinates: projective coordinates, inverted coordinates, extended coordinates. In the next table, we present these different coordinates and give the number of different addition formulas and doubling in **EFD** [1]. **Addition In Projective Coordinates:**

$$\begin{aligned} \text{Given a pair of points } (X_1 : Y_1 : Z_1) \text{ and } (X_2 : Y_2 : Z_2), \text{ their sum } (X_3 : Y_3 : Z_3) \\ A = Z_1 \cdot Z_2, \quad B = A^2, \quad C = X_1 \cdot X_2, \quad D = Y_1 \cdot Y_2, \\ E = dC \cdot D, \quad F = B - E, \quad G = B + E, \\ X_3 = A \cdot F \cdot ((X_1 + Y_1) \cdot (X_2 + Y_2) - C - D) \\ , Y_3 = A \cdot G \cdot (D - aC), Z_3 = F \cdot G. \end{aligned} \quad (1)$$

#### **Doubling In Projective Coordinates:**

Given a point  $(X_1 : Y_1 : Z_1)$ , its doubling point  $(X_3 : Y_3 : Z_3)$  is computed as:

$$\begin{aligned} B = (X_1 + Y_1)^2, \quad C = X_1^2, \quad D = Y_1^2, \\ E = aC, F = E + D, H = Z_1^2, \\ J = F - 2H, \\ X_3 = (B - C - D) \cdot J, \\ Y_3 = F \cdot (E - D), Z_3 = F \cdot J. \end{aligned} \quad (2)$$

### **2.3 Edwards curve over Binary Fields**

We recall that an Edwards elliptic curve over binary fields is defined as follows:

$$E : d_1(x + y) + d_2(x^2 + y^2) = xy + xy(x + y) + x^2 y^2$$

where  $d_1, d_2 \in K = \mathbb{F}_2^m$ , where  $m$  is a positive integer.

Let  $m$  be an integer and  $K = \mathbb{F}_{2^m}$  a binary field. Let  $d_1, d_2 \in K$  such that  $d_1 \neq 0$  and  $d_2 \neq d_1^2 + d_1$ . The binary Edwards curve with the coefficients  $d_1, d_2$  is the affine curve:

$$E : d_1(x + y) + d_2(x^2 + y^2) = xy + xy(x + y) + x^2y^2$$

If  $(x, y) \in E$  then  $(y, x) \in E$  and  $-(x, y) = (y, x)$ . The neutral element for the addition law is the point  $(0, 0)$ .

Let  $P_1 = (x_1, y_1)$  and  $P_2 = (x_2, y_2)$  be two points on the binary curve  $E$  and  $P_3 = (x_3, y_3) = P_1 + P_2$ .  $P_3$  is computed as follows:

$$x_3 = \frac{d_1(x_1 + x_2) + d_2(x_1 + y_1)(x_2 + y_2) + (x_1 + x_1^2)(x_2(y_1 + y_2 + 1) + y_1y_2)}{d_1 + (x_1 + x_1^2)(x_2 + y_2)}$$

$$y_3 = \frac{d_1(y_1 + y_2) + d_2(x_1 + y_1)(x_2 + y_2) + (y_1 + y_1^2)(y_2(x_1 + x_2 + 1) + x_1x_2)}{d_1 + (y_1 + y_1^2)(x_2 + y_2)}$$

If  $P_1 = P_2$  then  $P_3 = (x_3, y_3) = 2P_1$ .  $P_3$  is computed as follows:

$$x_3 = 1 + \frac{d_1(1 + x_1 + y_1)}{d_1 + x_1y_1 + x_1^2(1 + x_1 + y_1)}$$

$$y_3 = 1 + \frac{d_1(1 + x_1 + y_1)}{d_1 + x_1y_1 + y_1^2(1 + x_1 + y_1)}$$

In these different formulas we have inversions which are very expensive so it is preferable to use coordinate systems which will not need inversion such as projective coordinates.

Let  $P_1 = (x_1, y_1)$ , Let  $P_2 = (x_2, y_2)$  be points on the elliptic curve  $E$ , in projective coordinates  $P_1 = (Z_1x_1 : Z_1y_1 : Z_1)$ ,  $P_2 = (Z_2x_2 : Z_2y_2 : Z_2)$  for some  $Z_1, Z_2 \in \mathbb{F}_{2^m}$ . Let  $P_3 = (X_3 : Y_3 : Z_3) = P_1 + P_2$ .

---

**Algorithm:**dbl<sup>P</sup> over Edwards Binary Curves

**Input:**  $(P_1 \neq \mathcal{O})$

**Output:**  $P_3 = (X_3, Y_3, Z_3) = 2P_1$

---

Operation	
1	$A = X_1^2$
2	$B = A^2$
3	$C = Y_1^2$
4	$D = C^2$
5	$AE = Z_1^2$
6	$T_0 = E^2$
7	$F = d_1T_0$
8	$T_1 = B + D$
9	$G = \frac{d_2}{d_1}T_0$
10	$H = A \times E$
11	$I = C \times E$
12	$J = H + I$
13	$T_2 = d_2J$
14	$K = G + T_2$
15	$T_3 = F + J$
16	$Z_3 = T_3 + G$
17	$T_4 = K + H$
18	$X_3 = T_4 + D$
19	$T_5 = K + I$
20	$Y_3 = T_5 + B$

---

**Projective Addition:** Let  $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$  on a binary Edwards curve  $E$  with the coefficients  $d_1, d_2$ . In projective coordinates,  $P_1 = (X_1, Y_1, Z_1)$  where  $X_1 = x_1 Z_1, Y_1 = Y_1 Z_1$ , and  $P_2 = (X_2, Y_2, Z_2)$  where  $X_2 = x_2 Z_2, Y_2 = Y_2 Z_2$ .  $P_3 = (X_3, Y_3, Z_3) = P_1 + P_2$  is computed as follows:

---

**Algorithm:**  $\text{add}^P$  over Edwards Binary Curves

**Input:**  $(P_1 \neq \mathcal{O}, P_2 \neq \mathcal{O}, A, B)$

**Output:**  $P_3 = (X_3, Y_3, Z_3) = P_1 + P_2$

---

	Operation
1	$W_1 = X_1 + Y_1$ :
2	$W_2 = X_2 + Y_2$ :
3	$T_0 = X_1 + Z_1$ :
4	$A = X_1 \times T_0$ :
5	$T_1 = Y_1 + Z_1$ :
6	$B = Y_1 \times T_1$ :
7	$C = Z_1 \times Z_2$ :
8	$D = W_2 \times Z_2$ :
9	$T_2 = C \times C$ :
10	$E = d_1 \times T_2$ :
11	$T_3 = d_2 \times W_2$ :
12	$T_4 = d_1 \times Z_2$ :
13	$T_5 = T_4 + T_3$ :
14	$T_6 = W_1 \times C$ :
15	$H = T_5 \times T_6$ :
16	$T_7 = Z_1 \times C$ :
17	$I = d_1 \times T_7$ :
18	$T_8 = A \times D$ :
19	$U = E + T_8$ :
20	$T_9 = B \times D$ :
21	$V = E + T_9$ :
22	$S = U \times V$ :
23	$T_{10} = Y_2 + Z_2$ :
24	$T_{11} = A \times T_{10}$ :
25	$T_{12} = I + T_{11}$ :
26	$T_{13} = X_2 \times T_{12}$ :
27	$T_{14} = H + T_{13}$ :
28	$T_{15} = V \times Z_1$ :
29	$T_{16} = T_{14} \times T_{15}$ :
30	$T_{17} = S \times Y_1$ :
31	$X_3 = T_{16} + T_{17}$ :
32	$T_{18} = X_2 + Z_2$ :
33	$T_{19} = B \times T_{18}$ :
34	$T_{20} = I + T_{19}$ :
35	$T_{21} = Y_2 \times T_{20}$ :
36	$T_{22} = H + T_{21}$ :
37	$T_{23} = U \times Z_1$ :
38	$T_{24} = T_{22} \times T_{23}$ :
39	$T_{25} = S \times X_1$ :
40	$Y_3 = T_{24} + T_{25}$ :

## 2.4 Scalar Multiplication

Scalar multiplication consists of computing  $kP$  where  $P$  is a point on an elliptic curve and  $k$  is a scalar. For this we can use Right-to-Left binary NAF multiplication.

---

**Algorithm 1** Right-to-Left binary NAF multiplication using mixed [6]

---

**Input:**  $k, P = (X, Y, Z)$

**Output:**  $[k]P$

$(X_1, Y_1, Z_1) \leftarrow O$

$(T_1, T_2, T_3) \leftarrow (X, Y, Z)$

**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**  $(X_1, Y_1, Z_1) \leftarrow \text{ECADD}((X_1, Y_1, Z_1), (T_1, -T_2, T_3))$

$k \leftarrow k/2$

$(T_1, T_2, T_3, T_4) \leftarrow \text{ECDBL}(T_1, T_2, T_3)$

$(X_1, Y_1, Z_1) \leftarrow \text{ECADD}((X_1, Y_1, Z_1), (T_1, T_2, T_3))$

**Return:**  $(X_1, Y_1, Z_1)$

---

## 2.5 Side Channel Atomicity

The algorithm presented above is vulnerable to a Simple Power Analysis (SPA) attack. This vulnerability arises from the significant difference in power consumption between point addition and point doubling operations. It is clear that, in each iteration of the algorithm, a point addition is performed only if the corresponding bit of the key is 1. As a result, an attacker capable of experimentally distinguishing these variations in power consumption could effectively reconstruct the key by observing just a single power consumption trace.

A particularly effective countermeasure to protect elliptic curve-based protocols against SPA attacks is the atomicity countermeasure. This technique involves rewriting the addition and doubling formulas into an identical sequence of operations over the field. By doing so, it prevents an attacker from distinguishing between a point addition and a point doubling operation merely by analyzing the power consumption trace.

We propose several atomicity formulas specifically designed for Edwards curves, offering enhanced protection against SPA attacks.

Table 1 illustrates the computations for  $\text{ECADD}((X_2, Y_2, Z_2), (X_1, Y_1, Z_1))$  and  $\text{ECDBL}(X_1, Y_1, Z_1)$  over Edwards curves. Each column corresponds to an atomic pattern. The point addition is represented using two atomic patterns, while the point doubling utilizes a single pattern. This implementation effectively mitigates vulnerability to SPA attacks, as an attacker can no longer differentiate between the operations

performed based solely on the power consumption trace observed during the execution of the scalar multiplication.

Step	ECADD - part 1 $A_1$	ECADD - part 2 $A_2$	ECDBL - $\mathcal{D}$
1	$M = X_2 + Y_2$	$N = X_1 + Y_1$	$R_4 \leftarrow X_1 + Y_1$
2	$A = Z_1 Z_2$	$K = A.G$	*
3	$B = A^2$	*	$R_1 \leftarrow X_1^2$
4	*	*	$R_{21} \leftarrow Y_1^2$
5	$C = X_1 X_2$	$CF$	*
6	$D = Y_1 Y_2$	*	*
7	*	*	$R_3 \leftarrow c \times Z_1$
8	*	*	$R_4 \leftarrow R_4^2$
9	$CD$	$Y_3 \leftarrow KJ$	*
10	$E = d \times CD$	*	*
11	*	*	$R_3 \leftarrow R_3 + R_3$
12	$G = B + E$	$O = C + D$	$R_5 \leftarrow R_1 + R_2$
13	$F = B - E$	*	$R_2 \leftarrow R_1 - R_2$
14	$J = D - C$	*	$R_4 \leftarrow R_4 - R_5$
15	$I = AF$	$L = M.N$	$R_3 \leftarrow R_2 \times R_5$
16	*	$H = L - O$	$R_3 \leftarrow R_3 - R_5$
17	*	$X_3 = IH$	$R_1 \leftarrow R_1 \times R_3$
18	*	*	$X_3 = c \times R_1$
19	*	*	$Y_3 = c \times R_2$
20	*	*	$Z_3 = R_3 \times R_5$

**Table 1:** ECADD and ECDBL operations written with the same atomic pattern (\* represents a dummy operation)

Table 2 illustrates the computations for ECADD( $(X_2, Y_2, Z_2), (X_1, Y_1, Z_1)$ ) and ECDBL( $X_1, Y_1, Z_1$ ) over Twisted Edwards curves. Each column corresponds to an atomic pattern. The point addition is represented using one atomic pattern, and the point doubling utilizes a single pattern. This implementation effectively mitigates vulnerability to SPA attacks, as an attacker can no longer differentiate between the operations performed based solely on the power consumption trace observed during the execution of the scalar multiplication

Step	ECADD $A_1$	ECDBL - $\mathcal{D}$
1	$X_1 \times X_2$	*
2	*	$C = X_1^2$
3	$aC$	$E = aC$
4	$dC$	*
5	$D = Y_1 Y_2$	*
6	*	$D = Y_1^2$
7	$A = Z_1 Z_2$	*
8	$B = A^2$	
9	$E = dCD$	*
10	$F = B - E$	$E - D$
11	$G = B + E$	$F = E + D$
12	$H = X_1 + Y_1$	$X_1 + Y_1$
13	*	$B = (X_1 + Y_1)^2$
14	*	$H + H$
15	$D - aC$	$J = F - 2H$
16	$Z_3 = F.G$	$Z_3 = F.J$
17	$I = X_2 + Y_2$	*
18	$J = IH$	$Y_3 = F.(E - D)$
19	$K = J - C$	$B - C$
20	$L = K - D$	$B - C - D$
21	$AF$	*
22	$X_3 = AFK$	$X_3 = (B - C - D)J$
23	$AG$	*
24	$Y_3 = AG(D - aC)$	*

**Table 2:** ECADD and ECDBL operations written with the same atomic pattern (\* represents a dummy operation)

Table 2 illustrates the computations for ECADD( $(X_2, Y_2, Z_2), (X_1, Y_1, Z_1)$ ) and ECDBL( $(X_1, Y_1, Z_1)$ ) over Edwards curves. Each column corresponds to an atomic pattern. The point addition is represented using two atomic patterns, while the point doubling utilizes a single pattern. This implementation effectively mitigates vulnerability to SPA attacks, as an attacker can no longer differentiate between the operations performed based solely on the power consumption trace observed during the execution of the scalar multiplication.

Step	ECADD - part 1 $\mathcal{A}_1$	ECADD - part 2 $\mathcal{A}_2$	ECDBL - $\mathcal{D}$
1	*	*	$A = X_1^2$
2	*	*	$B = A^2$
3	*	*	$C = Y_1^2$
4	*	*	$D = C^2$
5	$W_1 = X_1 + Y_1$	$T_{10} = Y_2 + Z_2$	$T_1 = B + D$
6	$C = Z_1 Z_2$	$T_{11} = A.T_{10}$	*
7	$T_2 = C^2$	*	$E = Z_1^2$
8	*	*	$T_0 = E^2$
9	$T_6 = W_1.C$	$U.Z_1$	$H = A.E$
10	$E = d_1.T_2$	$I = d_1.T_7$	$F = d_1 T_0$
11	$T_7 = Z_1.C$	$T_{17} = S.Y_1$	$I = C.E$
12	$W_2 = X_2 + Y_2$	$T_{12} = I + T_{11}$	$J = H + I$
13	$T_3 = d_2 W_2$	$T_{25} = S.X_1$	$T_2 = d_2.J$
14	$T_0 = X_1 + Z_1$	$T_{18} = X_2 + Z_2$	$T_3 = F + J$
15	$A = X_1.T_{10}$	$T_{19} = B.T_{18}$	$G = d_2/d_1.T_0$
16	$T_1 = Y_1 + Z_1$	$T_{20} = I + T_{19}$	$Z_3 = T_3 + J$
17	$B = Y_1.T_1$	$T_{13} = X_2.T_{12}$	*
18	$D = W_2.Z_2$	$T_{21} = Y_2.T_{20}$	*
19	$T_8 = A.D$	$Y_3 = S.Z_1$	*
20	$T_4 = d_1.Z_2$	$T_{15} = V.Z_1$	*
21	$H = T_1.T_4$	$T_{23} = U.Z_1$	*
22	$T_5 = T_3 + T_4$	$T_{14} = H + T_{13}$	$K = G + T_2$
23	$U = E + T_8$	$T_{22} = H + T_{21}$	$T_4 = K + H$
24	$T_9 = B.D$	$T_{16} = T_{14}.T_{15}$	*
25	$V = E + T_9$	$X_3 = T_{16} + T_{17}$	$X_3 = T_4 + D$
26	$S = U.V$	$T_{24} = T_{22}.T_{23}$	*
27	*	$Y_3 = T_{24} + T_{25}$	$T_5 = K + I$
28	*	*	$Y_3 = T_5 + B$

**Table 3:** ECADD and ECDBL operations written with the same atomic pattern (\* represents a dummy operation)

## 2.6 Review of Big MAC Attack On Wierstrasss Elliptic Curve

The atomicity countermeasure in the implementation of scalar multiplication on elliptic curves was initially introduced by Chevalier and later improved more effectively by Giraud. This countermeasure was subsequently targeted for enhancement by Danger et al. in the context of the BIG MAC attack on elliptic curves.

The principle of the attack is as follows: Danger et al. focus on the atomicity countermeasure with the aim of identifying multiplications that share common operands. The attack is specifically designed to exploit this countermeasure, which relies on rewriting addition and doubling operations within the framework of scalar multiplication using the NAF representation.

---

The atomicity countermeasure involves reformulating point addition and doubling operations into a uniform sequence of operations. In this approach, addition is represented using two distinct atomic patterns (A1 and A2), while doubling is represented with a single atomic pattern (D). The authors of the BIG MAC attack observed that when the operation sequence follows the order  $A1, A2, D$ , exactly sixteen multiplications share common operands, creating an exploitable vulnerability.

The attack is recursive in nature. By observing the execution of the scalar multiplication algorithm, the attacker focuses on detecting the initial operations that occur during the implementation. This allows the attacker to derive the following information: if the first three operations follow the order  $A1, A2, D$ , then  $k_0 = 1$ ; otherwise,  $k_0 = 0$ . This approach enables the attacker to progressively reconstruct the bits of the secret key by exploiting this structural weakness.

two sets  $U_1$  and  $U_2$  are constructed as follow: initially, both  $U_1$  and  $U_2$  are empty. A particular function is applied to the power traces of the multiplications that might share a common operand. For each pair, one element is added to  $U_1$ , and the other to  $U_2$ . If the Euclidean distance between  $U_1$  and  $U_2$  is low, this indicates that the pairs indeed share a common operand. In this case, the three observed patterns are  $A1, A2, D$ , and the attacker concludes that  $k_0 \neq 0$ . The method is then repeated with the next three patterns to determine the value of  $k_1$ . Conversely, if the Euclidean distance between  $U_1$  and  $U_2$  is high. Unlike the initial BIG MAC attack on elliptic curves, which was limited to comparing two multiplications sharing a common operand, the improved version proposed by Danger et al. extends the analysis to sixteen traces. This approach has experimentally demonstrated that their improvement significantly increases the effectiveness of the attack, thereby enhancing its ability to exploit the vulnerabilities of the targeted implementations. We will apply this principle of comparing multiple multiplications in the next section to analyze the attack on Edwards curves.

### 3 BIG MAC Attack On Edwards Curves

Let us recall that the countermeasures we are targeting here are those presented in the previous section. We begin by considering Edwards curves defined over fields with odd characteristic. Specifically, we focus on the scalar multiplication algorithm applied to these curves

#### 3.1 BIG MAC Attack On Edwards Curves Over Fields of Odd Characteristic

By observing the scalar multiplication algorithm, the point  $(X_2, Y_2, Z_2)$  of  $A_1, A_2$  and the point  $(X_l, Y_l, Z_l)$  of  $D$  correspond to the same point  $R$  or  $-R$ . We summarize in Table 4 the multiplications and squares that share a common operand. These common operands are highlighted with a box, and identical indices are assigned to common operands appearing in two multiplications.

Step	ECADD - part 1 $\mathcal{A}_1$	ECADD - part 2 $\mathcal{A}_2$	ECDBL - $\mathcal{D}$
1	$M = X_2 + Y_2$	$N = X_1 + Y_1$	$R_4 \leftarrow X_1 + Y_1$
2	$A = Z_1 \times \boxed{Z_2}_1$	$K = \boxed{A}_{6,7} \times \boxed{G}_{11}$	*
3	$B = \boxed{A}_{4,6}^2$	*	$R_1 \leftarrow \boxed{X_1}_2^2$
4	*	*	$R_2 \leftarrow \boxed{Y_1}_3^2$
5	$C = X_1 \times \boxed{X_2}_2$	$\boxed{C}_8 \times \boxed{F}_{5,10}$	*
6	$D = Y_1 \times \boxed{Y_2}_3$	*	*
7	*	*	$R_3 \leftarrow c \times \boxed{Z_1}_1$
8	*	*	$R_4 \leftarrow R_4^2$
9	$\boxed{C}_8 \times D$	$Y_3 \leftarrow K \times J$	*
10	$E = d \times CD$	*	*
11	*	*	$R_3 \leftarrow R_3 + R_3$
12	$G = B + E$	$O = C + D$	$R_5 \leftarrow R_1 + R_2$
13	$F = B - E$	*	$R_2 \leftarrow R_1 - R_2$
14	$J = D - C$	*	$R_4 \leftarrow R_4 - R_5$
15	$I = \boxed{A}_{4,7} \times \boxed{F}_{5,9}$	$L = M \times N$	$R_3 \leftarrow R_2 \times R_5$
16	*	$H = L - O$	$R_3 \leftarrow R_3 - R_5$
17	$\boxed{F}_{9,10} \times \boxed{G}_{11}$	$X_3 = I \times H$	$R_1 \leftarrow R_1 \times R_3$
18	$Z_3 = c \times FG$	*	$X_3 = c \times R_1$
19	*	*	$Y_3 = c \times R_2$
20	*	*	$Z_3 = R_3 \times R_5$

Table 4: operations written with the same operand )

We identify in Table 5 the common values shared between two multiplications, specify which partners share them, and indicate the steps where these common values appear

numerotation	common operand	atomic partner sharing common operand
1	$Z_2$	$\mathcal{A}_1$ line 2 and $\mathcal{D}$ line 7
2	$X_2$	$\mathcal{A}_1$ line 5 and $\mathcal{D}$ line 3
3	$Y_2$	$\mathcal{A}_1$ line 6 and $\mathcal{D}$ line 4
4	$A$	$\mathcal{A}_1$ line 3 and $\mathcal{A}_1$ line 15
5	$F$	$\mathcal{A}_1$ line 15 and $\mathcal{A}_2$ line 5
6	$A$	$\mathcal{A}_1$ line 3 and $\mathcal{A}_2$ line 2
7	$A$	$\mathcal{A}_1$ line 15 and $\mathcal{A}_2$ line 2
8	$C$	$\mathcal{A}_1$ line 9 and $\mathcal{A}_2$ line 5
9	$F$	$\mathcal{A}_1$ line 15 and $\mathcal{A}_2$ line 17
10	$F$	$\mathcal{A}_1$ line 17 and $\mathcal{A}_2$ line 5
11	$G$	$\mathcal{A}_1$ line 17 and $\mathcal{A}_2$ line 2

Table 5: operations written with the same operand )

By examining the scalar multiplication algorithm and considering the atomicity countermeasure presented in this section, the possible operations of the three atomic components during the first iteration of the scalar multiplication algorithm could be  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ ,  $(\mathcal{D}, \mathcal{A}_1, \mathcal{A}_2)$ ,  $(\mathcal{D}, \mathcal{D}, \mathcal{D})$  or  $(\mathcal{D}, \mathcal{D}, \mathcal{A}_1)$ . The the first case appear if the first bit of the secret key  $k_0 = 1$  and the other case appear if  $k_0 = 0$ .

To recover the secret key bit by bit, the principle of the BIG MAC attack is applied. Since the attack is recursive, we first explain how the initial bit is obtained, and subsequent bits are recovered in a similar manner. Recall that if the first three components are  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ , then the bit is 1.

The process unfolds as follows:

1. **Trace Recovery:** We first extract the power consumption traces corresponding to the three initial components.
2. **Operation Decomposition:** Once these components are identified, we subdivide the operations (additions, multiplications, squarings, and subtractions) appearing in each component.

The BIG MAC attack assumes that the Hamming weight of a multiplication can be deduced from the power consumption trace. Two sets,  $U_1$  and  $U_2$ , are constructed as follows:

- **Initialization:** Both sets are initially empty.
- **Trace Assignment:** We identify the operations' traces that are assumed to involve common operands, given the components  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ .
- **Hamming Weight Assignment:** Using the BIG MAC principle, the Hamming weights of these multiplications are recovered. The weight of one operand is placed in  $U_1$ , and the other in  $U_2$ .

After retrieving the Hamming weights of the eleven multiplications that are expected to share common operands, the Euclidean distance between  $U_1$  and  $U_2$  is calculated:

- **Small Distance:** If the distance is very small, it confirms the presence of common operands among the eleven multiplications. Thus, the sequence corresponds to  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ , and  $k_0 = 1$ .
- **Large Distance:** Otherwise,  $k_0 = 0$ .

### 3.2 BIG MAC Attack On Twisted Edwards Curve

By observing the scalar multiplication algorithm, the point  $(X_2, Y_2, Z_2)$  of  $\mathcal{A}_1$  and the point  $(X_1, Y_1, Z_1)$  of  $\mathcal{D}$  correspond to the same point  $R$  or  $-R$ . We summarize in Table 6 the multiplications and squares that share a common operand. These common operands are highlighted with a box, and identical indices are assigned to common operands appearing in two multiplications.

Step	ECADD $\mathcal{A}_1$	ECDBL - $\mathcal{D}$
1	$C = X_1 \times \boxed{X_2}_1$	*
2	*	$C = \boxed{X_1}_1^2$
3	$a \times \boxed{C}_{11}$	$E = aC$
4	$d \times \boxed{C}_{11}$	*
5	$D = Y_1 \times \boxed{Y_2}_2$	*
6	*	$D = \boxed{Y_1}_2^2$
7	$A = Z_1 \times \boxed{Z_2}_3$	*
8	$B = \boxed{A}_{4,6}^2$	$H = \boxed{Z_1}_3^2$
9	$E = dC \times D$	*
10	$F = B - E$	$E - D$
11	$G = B + E$	$F = E + D$
12	$H = X_1 + Y_1$	$X_1 + Y_1$
13	*	$B = (X_1 + Y_1)^2$
14	*	$H + H$
15	$D - aC$	$J = F - 2H$
16	$Z_3 = \boxed{F}_9 \times \boxed{G}_8$	$Z_3 = \boxed{F}_7 \times \boxed{J}_{10}$
17	$I = X_2 + Y_2$	*
18	$J = I \times H$	$Y_3 = \boxed{F}_7 \times (E - D)$
19	$K = J - C$	$L = B - C$
20	$L = K - D$	$K = L - D$
21	$\boxed{A}_{4,5} \times \boxed{F}_9$	*
22	$X_3 = AF \times K$	$X_3 = K \times \boxed{J}_{10}$
23	$\boxed{A}_{5,6} \times \boxed{G}_8$	*
24	$Y_3 = AG \times (D - aC)$	*

**Table 6:** ECADD and ECDBL operations written with the same atomic pattern (\* represents a dummy operation)

We identify in Table 7 the common values shared between two multiplications, specify which partners share them, and indicate the steps where these common values appear

numeration	common operand	atomic partner sharing common operand
1	$X_2$	$\mathcal{A}_1$ line 1 and $\mathcal{D}$ line 2
2	$Y_2$	$\mathcal{A}_1$ line 5 and $\mathcal{D}$ line 6
3	$Z_2$	$\mathcal{A}_1$ line 7 and $\mathcal{D}$ line 8
4	$A$	$\mathcal{A}_1$ line 8 and $\mathcal{A}_1$ line 21
5	$A$	$\mathcal{A}_1$ line 21 and $\mathcal{A}_1$ line 23
6	$A$	$\mathcal{A}_1$ line 8 and $\mathcal{A}_1$ line 23
7	$F$	$\mathcal{D}$ line 16 and $\mathcal{D}$ line 18
8	$G$	$\mathcal{A}_1$ line 16 and $\mathcal{A}_1$ line 23
9	$F$	$\mathcal{A}_1$ line 16 and $\mathcal{A}_1$ line 21
10	$J$	$\mathcal{D}$ line 16 and $\mathcal{A}$ line 22
11	$C$	$\mathcal{A}_1$ line 3 and $\mathcal{A}_1$ line 4

**Table 7:** operations written with the same operand )

By examining the scalar multiplication algorithm and considering the atomicity countermeasure presented in this section, the possible operations of the two atomic components during the first iteration of the scalar multiplication algorithm could be  $(\mathcal{A}_1, \mathcal{D})$ ,  $(\mathcal{D}, \mathcal{A}_1)$ ,  $(\mathcal{D}, \mathcal{D})$  or  $\cdot$ . The first case appear if the first bit of the secret key  $k_0 = 1$  and the other case appear if  $k_0 = 0$ .

To recover the secret key bit by bit, the principle of the BIG MAC attack is applied. Since the attack is recursive, we first explain how the initial bit is obtained, and subsequent bits are recovered in a similar manner. Recall that if the first three components are  $(\mathcal{A}_1, \mathcal{D})$ , then the bit is 1.

The process unfolds as follows:

1. **Trace Recovery:** We first extract the power consumption traces corresponding to the two initial components.
2. **Operation Decomposition:** Once these components are identified, we subdivide the operations (additions, multiplications, squarings, and subtractions) appearing in each component.

The BIG MAC attack assumes that the Hamming weight of a multiplication can be deduced from the power consumption trace. Two sets,  $U_1$  and  $U_2$ , are constructed as follows:

- **Initialization:** Both sets are initially empty.
- **Trace Assignment:** We identify the operations' traces that are assumed to involve common operands, given the components  $(\mathcal{A}_1, \mathcal{D})$ .
- **Hamming Weight Assignment:** Using the BIG MAC principle, the Hamming weights of these multiplications are recovered. The weight of one operand is placed in  $U_1$ , and the other in  $U_2$ .

After retrieving the Hamming weights of the eleven multiplications that are expected to share common operands, the Euclidean distance between  $U_1$  and  $U_2$  is calculated:

- **Small Distance:** If the distance is very small, it confirms the presence of common operands among the eleven multiplications. Thus, the sequence corresponds to  $(\mathcal{A}_1, \mathcal{D})$ , and  $k_0 = 1$ .
- **Large Distance:** Otherwise,  $k_0 = 0$ .

### 3.3 BIG MAC Attack On Binary Edwards Curve

By observing the scalar multiplication algorithm, 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  $\mathcal{D}$  correspond to the same point  $R$  or  $-R$ . We summarize in Table 8 the multiplications and squares that share a common operand. These common operands are highlighted with a box, and identical indices are assigned to common operands appearing in two multiplications. By examining the scalar multiplication algorithm and considering the atomicity countermeasure presented in this section, the possible operations of the three atomic components during the first iteration of the scalar multiplication algorithm could be  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ ,  $(\mathcal{D}, \mathcal{A}_1, \mathcal{A}_2)$ ,  $(\mathcal{D}, \mathcal{D}, \mathcal{D})$  or  $(\mathcal{D}, \mathcal{D}, \mathcal{A}_1)$ . The first case appear if the first bit of the secret key  $k_0 = 1$  and the other case appear if  $k_0 = 0$ .

To recover the secret key bit by bit, the principle of the BIG MAC attack is applied. Since the attack is recursive, we first explain how the initial bit is obtained, and subsequent bits are recovered in a similar manner. Recall that if the first three components are  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ , then the bit is 1.

The process unfolds as follows:

1. **Trace Recovery:** We first extract the power consumption traces corresponding to the three initial components.
2. **Operation Decomposition:** Once these components are identified, we subdivide the operations (additions, multiplications, squarings, and subtractions) appearing in each component.

The BIG MAC attack assumes that the Hamming weight of a multiplication can be deduced from the power consumption trace. Two sets,  $U_1$  and  $U_2$ , are constructed as follows:

- **Initialization:** Both sets are initially empty.
- **Trace Assignment:** We identify the operations' traces that are assumed to involve common operands, given the components  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ .
- **Hamming Weight Assignment:** Using the BIG MAC principle, the Hamming weights of these multiplications are recovered. The weight of one operand is placed in  $U_1$ , and the other in  $U_2$ .

After retrieving the Hamming weights of the thirty-eight multiplications that are expected to share common operands, the Euclidean distance between  $U_1$  and  $U_2$  is calculated:

- **Small Distance:** If the distance is very small, it confirms the presence of common operands among the eleven multiplications. Thus, the sequence corresponds to  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{D})$ , and  $k_0 = 1$ .
- **Large Distance:** Otherwise,  $k_0 = 0$ .

Step	ECADD - part 1 $\mathcal{A}_1$	ECADD - part 2 $\mathcal{A}_2$	ECDBL - $\mathcal{D}$
1	*	*	$A = \boxed{X_1}_{26}^2$
2	*	*	$B = \boxed{A}_{34}^2$
3	*	*	$C = \boxed{Y_1}_{25}^2$
4	*	*	$D = \boxed{C}_{38}^2$
5	$W_1 = X_1 + Y_1$	$T_{10} = Y_2 + Z_2$	$T_1 = B + D$
6	$C = \boxed{Z_1}_{27,28,32,33} \times \boxed{Z_2}_{5,6,22}$	$T_{11} = \boxed{A}_{11} \times T_{10}$	*
7	$T_2 = \boxed{C}_{1,2}^2$	*	$E = \boxed{Z_1}_{22,23,24}^2$
8	*	*	$T_0 = \boxed{E}_{35,36}^2$
9	$T_6 = W_1 \times \boxed{C}_{1,3}$	*	$H = \boxed{A}_{34} \times \boxed{E}_{36,37}$
10	$E = d_1 \times T_2$	$I = d_1 \times T_7$	$F = d_1 \times \boxed{T_0}_{13}$
11	$T_7 = \boxed{Z_1}_{14,15,27,29} \times \boxed{C}_{2,3}$	$T_{17} = \boxed{S}_{19,20} \times \boxed{Y_1}_9$	$I = \boxed{C}_{38} \times \boxed{E}_{35,37}$
12	$W_2 = X_2 + Y_2$	$T_{12} = I + T_{11}$	$J = H + I$
13	$T_3 = d_2 \times \boxed{W_2}_4$	$T_{25} = \boxed{S}_{20,21} \times \boxed{X_1}_8$	$T_2 = d_2 \cdot J$
14	$T_0 = X_1 + Z_1$	$T_{18} = X_2 + Z_2$	$T_3 = F + J$
15	$A = \boxed{X_1}_8 \times T_{10}$	$T_{19} = \boxed{B}_{12} \times T_{18}$	$G = d_2/d_1 \times \boxed{T_0}_{13}$
16	$T_1 = Y_1 + Z_1$	$T_{20} = I + T_{19}$	$Z_3 = T_3 + J$
17	$B = \boxed{Y_1}_9 \times \boxed{T_1}_{34}$	$T_{13} = \boxed{X_2}_{26} \times T_{12}$	*
18	$D = \boxed{W_2}_4 \times \boxed{Z_2}_{5,7,23}$	$T_{21} = \boxed{Y_2}_{25} \times T_{20}$	*
19	$T_8 = \boxed{A}_{11} \times \boxed{D}_{10}$	$Y_3 = \boxed{S}_{19,21} \times \boxed{Z_1}_{28,29,30,31}$	*
20	$T_4 = d_1 \times \boxed{Z_2}_{6,7,24}$	$T_{15} = \boxed{V}_{18} \times \boxed{Z_1}_{14,16,30,32}$	*
21	$H = \boxed{T_1}_{34} \times T_4$	$T_{23} = \boxed{U}_{17} \times \boxed{Z_1}_{15,16,31,33}$	*
22	$T_5 = T_3 + T_4$	$T_{14} = H + T_{13}$	$K = G + T_2$
23	$U = E + T_8$	$T_{22} = H + T_{21}$	$T_4 = K + H$
24	$T_9 = \boxed{B}_{12} \times \boxed{D}_{10}$	$T_{16} = T_{14} \times T_{15}$	*
25	$V = E + T_9$	$X_3 = T_{16} + T_{17}$	$X_3 = T_4 + D$
26	$S = \boxed{U}_{17} \times \boxed{V}_{18}$	$T_{24} = T_{22} \times T_{23}$	*
27	*	$Y_3 = T_{24} + T_{25}$	$T_5 = K + I$
28	*	*	$Y_3 = T_5 + B$

Table 8: ECADD and ECDBL operations written with the same atomic pattern (\* represents a dummy operation)

We identify in Table 9 the common values shared between two multiplications, specify which partners share them, and indicate the steps where these common values appear

numerotation	common operand	atomic partner sharing common operand
1	$C$	$\mathcal{A}_1$ line 7 and $\mathcal{A}_1$ line 9
2	$C$	$\mathcal{A}_1$ line 7 and $\mathcal{A}_1$ line 11
3	$C$	$\mathcal{A}_1$ line 9 and $\mathcal{A}_1$ line 11
4	$W_2$	$\mathcal{A}_1$ line 13 and $\mathcal{A}_1$ line 18
5	$Z_2$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_1$ line 18
6	$Z_2$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_1$ line 20
7	$Z_2$	$\mathcal{A}_1$ line 18 and $\mathcal{A}_1$ line 20
8	$X_1$	$\mathcal{A}_1$ line 15 and $\mathcal{A}_2$ line 13
9	$Y_1$	$\mathcal{A}_1$ line 17 and $\mathcal{A}_2$ line 11
10	$D$	$\mathcal{A}_1$ line 19 and $\mathcal{A}_1$ line 24
11	$A$	$\mathcal{A}_1$ line 19 and $\mathcal{A}_2$ line 6
12	$B$	$\mathcal{A}_1$ line 15 and $\mathcal{A}_1$ line 24
13	$T_0$	$\mathcal{D}$ line 10 and $\mathcal{D}$ line 15
14	$Z_1$	$\mathcal{A}_1$ line 11 and $\mathcal{A}_2$ line 20
15	$Z_1$	$\mathcal{A}_1$ line 11 and $\mathcal{A}_2$ line 21
16	$Z_1$	$\mathcal{A}_2$ line 20 and $\mathcal{A}_2$ line 21
17	$U$	$\mathcal{A}_1$ line 26 and $\mathcal{A}_2$ line 21
1K8	$V$	$\mathcal{A}_1$ line 26 and $\mathcal{A}_2$ line 20
19	$S$	$\mathcal{A}_2$ line 11 and $\mathcal{A}_2$ line 19
20	$S$	$\mathcal{A}_2$ line 11 and $\mathcal{A}_2$ line 13
21	$S$	$\mathcal{A}_2$ line 13 and $\mathcal{A}_2$ line 19
22	$Z_2$	$\mathcal{A}_1$ line 6 and $\mathcal{D}$ line 7
23	$Z_2$	$\mathcal{A}_1$ line 18 and $\mathcal{D}$ line 7
24	$Z_2$	$\mathcal{A}_1$ line 24 and $\mathcal{D}$ line 7
25	$Y_2$	$\mathcal{A}_2$ line 18 and $\mathcal{D}$ line 3
26	$X_2$	$\mathcal{A}_2$ line 17 and $\mathcal{D}$ line 1
27	$Z_1$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_1$ line 11
28	$Z_1$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_2$ line 19
29	$Z_1$	$\mathcal{A}_1$ line 11 and $\mathcal{A}_2$ line 19
30	$Z_1$	$\mathcal{A}_2$ line 19 and $\mathcal{A}_2$ line 20
31	$Z_1$	$\mathcal{A}_2$ line 19 and $\mathcal{A}_2$ line 21
32	$Z_1$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_2$ line 20
33	$Z_1$	$\mathcal{A}_1$ line 6 and $\mathcal{A}_2$ line 21
34	$A$	$\mathcal{D}$ line 2 and $\mathcal{D}$ line 9
35	$E$	$\mathcal{D}$ line 8 and $\mathcal{D}$ line 11
36	$E$	$\mathcal{D}$ line 8 and $\mathcal{D}$ line 9
37	$E$	$\mathcal{D}$ line 9 and $\mathcal{D}$ line 11
38	$C$	$\mathcal{D}$ line 4 and $\mathcal{D}$ line 11

Table 9: operations written with the same operand )

In this paper, we focused on the BIG MAC attack on Edwards curves. First, we proposed atomicity formulas for Edwards curves, Twisted Edwards curves, and binary Edwards curves. Using the principles of the BIG MAC attack introduced by Danger et al., we identified eleven multiplications and squarings with common operands in our atomicity formulas for Edwards curves, eleven multiplications and squarings with common operands for Twisted Edwards curves, and thirty-eight multiplications and doublings with common operands for binary Edwards curves. We explained how an attacker could exploit these common operands to recursively recover the bits of the secret key.

We would like to apply this attack to hyperelliptic curves.

## References

1. Bernstein, D. J., Birkner, P., Joye, M., Lange, T., & Peters, C. (2008). Twisted Edwards curves. In S. Vaudenay (Ed.), *Progress in cryptology – AFRICACRYPT 2008* (pp. 389–405). Springer. [https://doi.org/10.1007/978-3-540-68164-9\\_26](https://doi.org/10.1007/978-3-540-68164-9_26)
2. Danger, J.-L., Guilley, S., Hoogvorst, P., Murdica, C., & Naccache, D. (2016). Improving the Big Mac attack on elliptic curve cryptography. In P. Y. A. Ryan, D. Naccache, & J.-J. Quisquater (Eds.), *The new codebreakers: Essays dedicated to David Kahn on the occasion of his 85th birthday* (pp. 374–386). Springer. [https://doi.org/10.1007/978-3-662-49301-4\\_24](https://doi.org/10.1007/978-3-662-49301-4_24)
3. Chevallier-Mames, B., Ciet, M., & Joye, M. (2004). Low-cost solutions for preventing simple side-channel analysis: Side-channel atomicity. *IEEE Transactions on Computers*, 53(6), 760–768. <https://doi.org/10.1109/TC.2004.13>
4. Bauer, A., Jaulmes, E., Prouff, E., Reinhard, J. R., & Wild, J. (2019). *Horizontal collision correlation attack on elliptic curves*. Cryptology ePrint Archive. <https://eprint.iacr.org/2019/523>
5. Takemura, Y., Hakuta, K., & Shinohara, N. (2020). ECC atomic block with NAF against strong side-channel attacks on binary curves. *International Journal of Networking and Computing*, 10(2), 277–292. <https://doi.org/10.15017/4100465>
6. Joye, M. (2008). Fast point multiplication on elliptic curves without precomputation. In J. von zur Gathen, J. L. Imaña, & Ç. K. Koç (Eds.), *Arithmetic of finite fields: 2nd International Workshop, WAIFI 2008* (pp. 36–46). Springer. [https://doi.org/10.1007/978-3-540-69499-1\\_4](https://doi.org/10.1007/978-3-540-69499-1_4)
7. Giraud, C., & Verneuil, V. (2010). Atomicity improvement for elliptic curve scalar multiplication. In D. Gollmann, J.-L. Lanet, & J. Iguchi-Cartigny (Eds.), *Smart card research and advanced application: 9th IFIP WG 8.8/11.2 International Conference, CARDIS 2010* (pp. 80–101). Springer. [https://doi.org/10.1007/978-3-642-12510-2\\_7](https://doi.org/10.1007/978-3-642-12510-2_7)
8. Walter, C. D. (2001). Sliding windows succumbs to Big Mac attack. In Ç. K. Koç, D. Naccache, & C. Paar (Eds.), *Cryptographic hardware and embedded systems – CHES 2001* (pp. 286–299). Springer. [https://doi.org/10.1007/3-540-44709-1\\_24](https://doi.org/10.1007/3-540-44709-1_24)