Elliptic Curve Cryptosystems in the Presence of Permanent and Transient Faults

Mathieu Ciet[1,⋆] and Marc Joye[2]

[1] Université catholique de Louvain, UCL Crypto Group
Place du Levant 3, 1348 Louvain-la-Neuve, Belgium
ciet@dice.ucl.ac.be
[2] Gemplus, Card Security Group
La Vigie, Av. du Jujubier, ZI Athélia IV, 13705 La Ciotat Cedex, France
marc.joye@gemplus.com — http://www.geocities.com/MarcJoye/

Abstract. Elliptic curve cryptosystems in the presence of faults were studied by Biehl, Meyer and Müller (2000). The first fault model they consider requires that the input point $\boldsymbol{P}$ in the computation of $d\boldsymbol{P}$ is *chosen* by the adversary. Their second and third fault models only require the knowledge of $\boldsymbol{P}$. But these two latter models are less 'practical' in the sense that they assume that only a *few* bits of error are inserted (typically exactly one bit is supposed to be disturbed) either into $\boldsymbol{P}$ just prior to the point multiplication or during the course of the computation in a chosen location.

This paper relaxes these assumptions and shows how *random* (and thus unknown) errors in either coordinates of point $\boldsymbol{P}$, in the elliptic curve parameters or in the field representation enable the (partial) recovery of multiplier d. Then, from multiple point multiplications, we explain how this can be turned into a total key recovery. Simple precautions to prevent the leakage of secrets are also discussed.

Keywords. Elliptic curve cryptography, fault analysis, fault attacks, physical security, information leakage.

1 Introduction

Elliptic curve cryptography was introduced in the mid 1980s independently by Koblitz [13] and Miller [20] as a promising alternative for cryptographic protocols based on the discrete logarithm problem in the multiplicative group of a finite field (e.g., Diffie-Hellman key exchange or ElGamal encryption/signature). The

⋆ The work described in this paper has been supported [in part] by the Commission of the European Communities through the IST Programme under Contract IST-1999-12324, http://www.cryptonessie.org/. The information in this document is provided as is, and no guarantee or warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at his sole risk and liability. The views expressed are those of the authors and do not represent an official view/position of the NESSIE project (as a whole).

security of elliptic curve cryptosystems relies on the hardness of solving the elliptic curve discrete logarithm problem (ECDLP): given points $\boldsymbol{P}$ and $\boldsymbol{Q} = d\boldsymbol{P}$ on an elliptic curve, one has to recover multiplier d.

1.1 Physical security

Kocher *et al.* introduced the notion of *side-channel analysis* in [14, 15] and showed the importance for an implementation of being resistant against side-channel analysis (e.g., secret leakage from power consumption).
Resistance against fault analysis [7, 5] is another threat and should be taken into account, since sensitive information may leak when the cryptosystem operates under unexpected conditions.

The security of point multiplication on elliptic curves in the presence of faults was considered by Biehl, Meyer and Müller [4]. They extended fault attacks initially mounted against the RSA cryptosystem to schemes using elliptic curves. Remarking that elliptic curve parameter a_6 is not used in the classical addition formulæ on elliptic curves, they made the key observation that a random element $(\tilde{x}, \tilde{y})$ (not on the original elliptic curve) may define a group for which it is computationally feasible to solve the ECDLP. Therefore, by forcing the input point as $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y})$, they were able to derive information about d from the output value $d\widetilde{\boldsymbol{P}}$.

Many elliptic curve cryptosystems do not allow to choose (or force) the input point. In the second part of their paper, Biehl *et al.* suppose that an error of one bit occurs and then try to recover it by successive search.

1.2 Our contribution

In this paper, we rather consider a more practical scenario and give several refinements of [4].

Typically, in a cryptographic device, the system parameters are stored in non-volatile memory (e.g., EEPROM) and are then transferred into working memory (e.g., RAM), when needed, for the elliptic curve computations. In our first model, we assume that there is a *permanent fault*, in a *unknown* position, in *any* system parameter defining the elliptic curve over which the computations are carried out. In our second model, we analyze the consequences of faults during the transfer of the system parameters into working memory. We call such a fault a *transient fault*. In both models, we explain how this may yield information on (secret) multiplier d.

The rest of this paper is organized as follows. In the next section, we review the basics of elliptic curve cryptography. Then, in Section 3, we exhibit how the presence of faults in the public parameters of an elliptic curve cryptosystem may expose the secret key. Finally, we conclude in Section 4.

2 Elliptic Curve Cryptography

An elliptic curve over a field $\mathbb{K}$ is the set of points $(x, y) \in \mathbb{K} \times \mathbb{K}$ satisfying the Weierstraß equation

$$E_{/\mathbb{K}} : y^2 + a_1\, xy + a_3\, y = x^3 + a_2\, x^2 + a_4\, x + a_6 \tag{1}$$

along with the point $\boldsymbol{O}$ at infinity. This set of points form an *additive group* where $\boldsymbol{O}$ is the neutral element and where the group law is given by the 'chord-and-tangent' rule. Given two points $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2)$, the sum $\boldsymbol{P_3} = \boldsymbol{P_1} + \boldsymbol{P_2} = (x_3, y_3)$ is defined as

$$x_3 = \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \text{ and } y_3 = (x_1 - x_3)\lambda - y_1 - a_1 x_3 - a_3 \tag{2}$$

with $\lambda = \begin{cases} \frac{3x_1^2 + 2a_2x_1 + a_4 - a_1y_1}{2y_1 + a_1x_1 + a_3} & \text{if } \boldsymbol{P_1} = \boldsymbol{P_2}, \\ \frac{y_1 - y_2}{x_1 - x_2} & \text{otherwise.} \end{cases}$

Remark 1. When the characteristic of $\mathbb{K}$, $\operatorname{Char}\mathbb{K}$, is a 'large' prime (namely, ≥ 5), one can take $a_1 = a_2 = a_3 = 0$ in the Weierstraß parameterization, without loss of generality. Likewise when $\operatorname{Char}\mathbb{K} = 2$, one can take $a_1 = 1$ and $a_3 = a_4 = 0$, provided that the elliptic curve is non-supersingular. This is the case of interest for cryptographic applications since supersingular curves are prone to the MOV attack [18].

The scalar multiplication (or point multiplication) is the operation consisting in computing $\boldsymbol{Q} = d\boldsymbol{P} = \boldsymbol{P} + \boldsymbol{P} + \cdots + \boldsymbol{P}$ (d times). A related operation is the *discrete logarithm*, which consists in computing d from $\boldsymbol{P}$ and $\boldsymbol{Q} = d\boldsymbol{P}$, and is the basis for the security of cryptographic schemes based on elliptic curves over finite fields [13, 20]. The discrete logarithm problem on a *general* elliptic curve (ECDLP) is particularly attractive since there is no known sub-exponential algorithm to solve it. The best method available for attacking the ECDLP is Pollard's method [21, 22] and variants thereof.

3 Permanent and Transient Faults

Consider the elliptic curve $E(a_1, a_2, a_3, a_4, a_6)$ over $\mathbb{K}$ given by Eq. (1). In [4], Biehl, Meyer and Müller observe that parameter a_6 is not involved in the addition formulæ (cf. Eq (2)). Consequently, if a cryptographic device (e.g., a smart card) receives on input a 'point' $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y}) \in \mathbb{K} \times \mathbb{K}$ but $\widetilde{\boldsymbol{P}} \notin E$ then the scalar multiplication $d\widetilde{\boldsymbol{P}}$ will take place over the curve $\widetilde{E}(a_1, a_2, a_3, a_4, \widetilde{a}_6)$ with

$$\widetilde{a}_6 = \tilde{y}^2 + a_1\, \tilde{x}\tilde{y} + a_3\, \tilde{y} - \tilde{x}^3 - a_2\, \tilde{x}^2 - a_4\, \tilde{x}$$

instead of over the original curve $E(a_1, a_2, a_3, a_4, a_6)$.

Assume that point $\widetilde{\boldsymbol{P}}$ is chosen so that $\widetilde{E}(a_1, a_2, a_3, a_4, \widetilde{a}_6)$ is an elliptic curve whose order has a small (or smooth) factor r and that the order of $\widetilde{\boldsymbol{P}}$ as an element of $\widetilde{E}$, $\operatorname{ord}_{\widetilde{E}}(\widetilde{\boldsymbol{P}})$, is equal to r. Then, provided that discrete logarithms are computable in $\langle\widetilde{\boldsymbol{P}}\rangle$, the subgroup of order r generated by $\widetilde{\boldsymbol{P}}$, the value of $d \pmod r$ can be recovered.[3] Therefore, repeating the process with sufficiently many different *chosen* points $\widetilde{\boldsymbol{P}}_i$ yields the values of $d \bmod r_i$ (where $r_i = \mathrm{ord}_{\widetilde{E}_i}(\widetilde{\boldsymbol{P}}_i)$) from $d\widetilde{\boldsymbol{P}}_i$, and so the whole value of d, by Chinese remaindering.

Most elliptic curve cryptosystems take a *determined* point $\boldsymbol{P}$ on input, making the previous attack inapplicable (a noteworthy exception is the basic ElGamal encryption [9]). Indeed, being a system parameter, point $\boldsymbol{P}$ is stored in the non-volatile memory of the cryptographic device and is read from that memory for the computation of $d\boldsymbol{P}$.

In this section, we analyze the implications of *permanent* faults (intentional or accidental) in non-volatile memory, where the system parameters are stored. We alternately consider permanent faults in the representation of point $\boldsymbol{P}$, in the definition field $\mathbb{K}$, and in curve parameters $\{a_1, \ldots, a_6\}$. Our analysis also extends to *transient* faults originating from a perturbation of the reading in non-volatile memory, and resulting in faulty values for the system parameters actually used in working memory, throughout the computation. The difference between the two models is that a transient fault, as defined, is fixed throughout a *single* execution (but may vary from one execution to another one) whereas a permanent fault always sticks invariant.

3.1 Faults in the base point

Without loss of generality, we assume that the x-coordinate of point $\boldsymbol{P}$ is corrupted (the case of a corrupted y-coordinate is similar). The cryptographic device then computes $\widetilde{\boldsymbol{Q}} = d\widetilde{\boldsymbol{P}}$ with $\widetilde{\boldsymbol{P}} = (\tilde{x}, y)$. Contrary to the case considered in [4], the value of $\widetilde{\boldsymbol{P}}$ is *unknown* but fixed (in particular, there is no hypothesis on the number of flipped bits). It is however easy to recover the value of $\widetilde{\boldsymbol{P}}$ from the output value $\widetilde{\boldsymbol{Q}} = d(\tilde{x}, y) = (\tilde{x}_d, \tilde{y}_d)$. Point $\widetilde{\boldsymbol{Q}}$ defines the curve $\widetilde{E}$ on which the computation was carried out, i.e., $\widetilde{E}(a_1, a_2, a_3, a_4, \widetilde{a}_6)$ with

$$\widetilde{a}_6 = \tilde{y}_d^{\,2} + a_1\, \tilde{x}_d\, \tilde{y}_d + a_3\, \tilde{y}_d - \tilde{x}_d^{\,3} - a_2\, \tilde{x}_d^{\,2} - a_4\, \tilde{x}_d \ . \tag{3}$$

Therefore, since $\widetilde{\boldsymbol{P}} = (\tilde{x}, y) \in \widetilde{E}$, it follows that $\tilde{x}$ is a root in $\mathbb{K}$ of the polynomial (over $\mathbb{K}[X]$) given by

$$X^3 + a_2\, X^2 + (a_4 - a_1\, y)X + (\widetilde{a}_6 - y^2 - a_3\, y) \ .$$

Suppose first that this polynomial has a unique root, which must be $\tilde{x}$. Provided that $r = \mathrm{ord}_{\widetilde{E}}(\widetilde{\boldsymbol{P}})$ (i.e., the order of $\widetilde{\boldsymbol{P}} = (\tilde{x}, y)$ in $\widetilde{E}$) is small (or smooth) enough so that the discrete logarithm of $\widetilde{\boldsymbol{Q}}$ w.r.t. $\widetilde{\boldsymbol{P}}$ is computable, the value of $d \bmod r$ can be recovered.

If now the above polynomial has 2 or 3 roots, then there are 2 or 3 possible candidates for $\tilde{x}$. In the permanent-fault model, the ambiguity can however be

[3] Note that r needs not be prime — when r is a smooth composite number, discrete logarithms can be attacked with the Pohlig-Hellman algorithm.

removed by assuming that only a portion of x is corrupted (remember that parameter $\boldsymbol{P} = (x, y)$ is public). Hence, the candidate having the most bits matching those of x is likely to be $\tilde{x}$. In the transient-fault model, the whole value of x is likely to be corrupted; the actual value of $\tilde{x}$ is then known with probability 1, $\frac{1}{2}$ or $\frac{1}{3}$, according to the number of candidates. Next, once $\widetilde{\boldsymbol{P}} = (\tilde{x}, y)$ is known (or more exactly, guessed), the value of $d \bmod r$ can be recovered.

Remark 2. If it is the y-coordinate of $\boldsymbol{P}$ that is corrupted (i.e., $\widetilde{\boldsymbol{P}} = (x, \tilde{y})$) then $\tilde{y}$ is a root of a polynomial of degree 2 and the above methodology applies to recover the value of $d \bmod r$. (Note that an error in the y-coordinate only makes sense when there is no point compression[4] —with point compression, an error merely changes $\boldsymbol{P}$ into $-\boldsymbol{P}$.)
A more intricate case appears when both the x- and the y-coordinates of $\boldsymbol{P}$ are corrupted. We write $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y})$ the corresponding point. As before, the output value $d\widetilde{\boldsymbol{P}} = (\tilde{x}_d, \tilde{y}_d)$ yields the value of $\widetilde{a}_6$ from Eq. (3). But it seems that there is no way to recover the whole value of $\widetilde{\boldsymbol{P}}$; we only know that it lies on the curve $\widetilde{E}(a_1, a_2, a_3, a_4, \widetilde{a}_6)$. Further assumptions are needed to completely recover $\widetilde{\boldsymbol{P}}$. For example, in case of permanent fault, if it is possible to flip one bit of d, say bit d_i, during the computation of $d\widetilde{\boldsymbol{P}}$ then the value of $\pm 2^i \widetilde{\boldsymbol{P}}$ can be obtained as $d\widetilde{\boldsymbol{P}} - d'\widetilde{\boldsymbol{P}}$ where

$$d' = d + (\neg d_i - d_i)\, 2^i \ .$$

Successively halving $\pm 2^i \widetilde{\boldsymbol{P}}$ on $\widetilde{E}(a_1, a_2, a_3, a_4, \widetilde{a}_6)$ eventually yields the value of $\pm\widetilde{\boldsymbol{P}}$. Then, once $\pm\widetilde{\boldsymbol{P}}$ is retrieved, the value of $\pm d \bmod r$ can be recovered. This extends the attacks of [3, 12] in the sense that more information on secret d (i.e., $\pm d \bmod r$) can be recovered.

3.2 Faults in the definition field

We now suppose that the representation of field $\mathbb{K}$ in non-volatile memory is faulty (permanent fault) or that an error occurred in the transfer from non-volatile memory to working memory (transient fault). We have to handle the cases $\mathbb{K} = \mathbb{F}_p$ and $\mathbb{K} = \mathbb{F}_{2^q}$ separately since their internal representations are different.

Large prime field Let p be a large prime number. The field $\mathbb{F}_p$ is simply represented by the value of p, stored as a binary string in non-volatile memory. An error on $\mathbb{K}$ means in this case an error in the representation of p. We let $\tilde{p}$ denote the faulty value. The computations are then carried out modulo $\tilde{p}$ instead of modulo p.

[4] If the x-coordinate of point $\boldsymbol{P}$ is fixed, its y-coordinate satisfies a quadratic in y. Since a single bit is sufficient to distinguish between the two possible solutions of a quadratic, a point can be compressed into its x-coordinate and this additional bit.

As mentioned in Section 2, elliptic curves over a large prime field are given by the simplified equation

$$E_{/\mathbb{F}_p} : y^2 = x^3 + a_4\,x + a_6 \ .$$

Over $\mathbb{F}_{\tilde{p}}$, point $\boldsymbol{P} = (x, y)$ becomes $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y})$ where $\tilde{x} \equiv x \pmod{\tilde{p}}$ and $\tilde{y} \equiv y \pmod{\tilde{p}}$. The computation of $d\boldsymbol{P}$ on E is replaced by $d\widetilde{\boldsymbol{P}}$ on the curve $\widetilde{E}$ defined by[5]

$$\widetilde{E}_{/\mathbb{Z}_{\tilde{p}}} : y^2 = x^3 + \widetilde{a}_4\,x + \widetilde{a}_6$$

where $\widetilde{a}_4 \equiv a_4 \pmod{\tilde{p}}$. Since both $\widetilde{\boldsymbol{P}} = (\tilde{x}, \tilde{y})$ and $\widetilde{\boldsymbol{Q}} = d\widetilde{\boldsymbol{P}} = (\tilde{x}_d, \tilde{y}_d)$ are points on $\widetilde{E}$, we deduce that

$$\widetilde{a}_6 \equiv \tilde{y}^2 - \tilde{x}^3 - \widetilde{a}_4\,\tilde{x} \equiv y^2 - x^3 - a_4\,x \equiv \tilde{y}_d^{\,2} - \tilde{x}_d^{\,3} - a_4\,\tilde{x}_d \pmod{\tilde{p}} \ . \tag{4}$$

Letting $R = y^2 - x^3 - a_4\,x$ and $\widetilde{R}_d = \tilde{y}_d^{\,2} - \tilde{x}_d^{\,3} - a_4\,\tilde{x}_d$ (R and $\widetilde{R}_d$ are evaluated over $\mathbb{Z}$), the previous relation implies that $\tilde{p}$ divides $\Delta = \max(R, \widetilde{R}_d) - \min(R, \widetilde{R}_d)$. Consequently, the factorization of Δ will reveal the value of $\tilde{p}$ as the combination of the factors of Δ whose product has the most bits matching those of p. If no such $\tilde{p}$ is found (e.g., a transient fault gave rise to a random value for $\tilde{p}$), we have to assume that the original prime, p, satisfies additional properties. A concrete example is discussed in § 3.2.

Remark 3. Although being a $3k$-bit integer (where k is the bit-length of prime p), Δ is within the range of modern factorization algorithms since k is typically a 160-bit integer. Furthermore, when the fault is permanent, the length of the number to be factored can be lowered from an additional scalar multiplication with point $\widetilde{\boldsymbol{P}}$. Let $\widetilde{\boldsymbol{Q}}' = d'\widetilde{\boldsymbol{P}} = (\tilde{x}_{d'}, \tilde{y}_{d'})$ with $d' \neq d$. Then $(\tilde{x}_{d'}, \tilde{y}_{d'})$ satisfies Eq. (4) and, letting $\widetilde{R}_{d'} = \tilde{y}_{d'}^{\,2} - \tilde{x}_{d'}^{\,3} - a_4\,\tilde{x}_{d'}$ and $\Delta' = \max(R, \widetilde{R}_{d'}) - \min(R, \widetilde{R}_{d'})$, it follows that $\tilde{p}$ must divide $\gcd(\Delta, \Delta')$. If further (distinct) scalar multiplications are available, $\tilde{p}$ can be obtained easilier as a factor of $\gcd(\Delta, \Delta', \Delta'', \dots)$.

We now assume that the value of $\tilde{p}$ is known. Let $\tilde{p} = \prod_{i=1}^{l} \tilde{q}_i^{\,e_i}$ denote the prime factorization of $\tilde{p}$. Consider the direct product of groups

$$\mathbb{G} = \widetilde{E}_1(\mathbb{Z}/\tilde{q}_1^{\,e_1}\mathbb{Z}) \times \cdots \times \widetilde{E}_l(\mathbb{Z}/\tilde{q}_l^{\,e_l}\mathbb{Z}) \ .$$

Since the computation of $\widetilde{\boldsymbol{Q}} = k\widetilde{\boldsymbol{P}}$ was supposed successful, point $\widetilde{\boldsymbol{Q}}$ corresponds to a unique element of $\mathbb{G}$, namely $(\widetilde{\boldsymbol{Q}}_1, \dots, \widetilde{\boldsymbol{Q}}_l)$ with $\widetilde{\boldsymbol{Q}}_i = \widetilde{\boldsymbol{Q}} \bmod \tilde{q}_i^{\,e_i}$ [17, § 2.5]. Hence, if $r_i = \mathrm{ord}_{\widetilde{E}_i}(\widetilde{\boldsymbol{P}})$ denotes the order of $\widetilde{\boldsymbol{P}}$ viewed as an element of the elliptic curve $\widetilde{E}_i$ (i.e., $\widetilde{\boldsymbol{P}}_i = \widetilde{\boldsymbol{P}} \bmod \tilde{q}_i^{\,e_i}$), then solving the discrete logarithm

[5] When $\tilde{p}$ is not prime, $\mathbb{Z}_{\tilde{p}} = \mathbb{Z}/\tilde{p}\mathbb{Z}$ is no longer a field but a ring. As a result, the computation of $d\widetilde{\boldsymbol{P}}$ on $\widetilde{E}$ is not guaranteed to succeed. This occurs when the computation involves a point that is the point at infinity modulo a factor $\tilde{p}_1$ of $\tilde{p}$ but not modulo $\tilde{p}_2 = \tilde{p}/\tilde{p}_1$.

in $\langle \widetilde{\boldsymbol{P}}_i \rangle \subseteq \widetilde{E}_i$ yields the value of $d \bmod r_i$ so that by Chinese remaindering the value of

$$d \bmod \operatorname{lcm}(r_1, \ldots, r_l)$$

can be calculated.

Binary field The field $\mathbb{F}_{2^q}$ is usually regarded as a quotient $\mathbb{F}_2[X]/(\mathcal{P})$ where $\mathcal{P}$ is an irreducible polynomial of degree q over $\mathbb{F}_2$. Each element $\alpha \in \mathbb{F}_{2^q}$ is represented as a binary string $(\alpha_0 \ldots \alpha_{q-1})$ corresponding to polynomial $\sum_{i=0}^{q-1} \alpha_i X^i \pmod{\mathcal{P}(X)}$ in $\mathbb{F}_2[X]$. With this polynomial representation, the field $\mathbb{F}_{2^q}$ is determined by polynomial $\mathcal{P}$, which is stored in non-volatile memory as a binary string $(b_0 \ldots b_{q-1} b_q)$ corresponding to $\mathcal{P}(X) = \sum_{i=0}^{q} b_i X^i$ with $b_q = 1$. Over $\mathbb{F}_{2^q}$, a (non-supersingular) elliptic curve is given by the simplified Weierstraß equation

$$E_{/\mathbb{F}_{2^q}} : y^2 + xy = x^3 + a_2 x^2 + a_6 \quad .$$

Remark 4. There exist other representations for elements of $\mathbb{F}_{2^q}$. A common choice is the normal basis representation. We shall not consider this representation here, as optimized implementations of elliptic curve doubling with normal bases make use of parameter a_6 [1, §. A.10.2]. Our analysis, however, implies that parameter a_6 is not used. Moreover, we assume that affine coordinates are used, as a projective doubling needs the value of a_6 [1, §. A.10.6]. Furthermore, note that affine coordinates lead to faster arithmetic than projective coordinates [8] in binary fields with polynomial representation.

Suppose that $\mathbb{K} = \mathbb{F}_{2^q}$ is faulty, namely that there is an error in the representation of $\mathcal{P}$. As a result, computations are performed modulo $\widetilde{\mathcal{P}}(X) = \sum_{i=0}^{q} \tilde{b}_i X^i$ instead of modulo $\mathcal{P}(X)$. Viewing elements as polynomials over $\widetilde{\mathbb{K}}[X]$, similarly to prime case, polynomial $\widetilde{\mathcal{P}}$ can be recovered by observing that

$$\widetilde{a}_6 \equiv y^2 + xy + x^3 + a_2 x^2 \equiv \tilde{y}_d^2 + \tilde{x}_d \tilde{y}_d + \tilde{x}_d^3 + a_2 \tilde{x}_d^2 \pmod{\widetilde{\mathcal{P}}(X)}$$

where $(\tilde{x}_d, \tilde{y}_d) = d\widetilde{\boldsymbol{P}}$ and $\widetilde{\boldsymbol{P}} \equiv \boldsymbol{P} \pmod{\widetilde{\mathcal{P}}(X)}$. Hence, letting

$$\Delta(X) = y^2 + xy + x^3 + a_2 x^2 + \tilde{y}_d^2 + \tilde{x}_d \tilde{y}_d + \tilde{x}_d^3 + a_2 \tilde{x}_d^2 \quad (\text{over } \mathbb{F}_2[X]), \tag{5}$$

it follows that $\widetilde{\mathcal{P}}(X)$ divides $\Delta(X)$. So, given the factorization of Δ, trying all possible combinations yields $\widetilde{\mathcal{P}}$ as the polynomial whose representation best matches the representation of $\mathcal{P}$. Further, in case of permanent fault, an additional scalar multiplication, $\widetilde{\boldsymbol{Q}}' = d'\widetilde{\boldsymbol{P}}$ with $d' \not\equiv d$, eases the recovery of $\widetilde{\mathcal{P}}$ as a factor of $\gcd(\Delta, \Delta')$ where Δ' is defined from $\widetilde{\boldsymbol{Q}}'$.
When the fault is transient, it is always possible to distinguish $\widetilde{\mathcal{P}}$. In § 3.2, we present a technique that can successfully recover $\widetilde{\mathcal{P}}$ in most practical implementations.

If polynomial $\mathcal{P}(X) = \sum_{i=0}^{q} b_i X^i$ representing the field $\mathbb{F}_{2^q}$ is modified into polynomial $\widetilde{\mathcal{P}}(X) = \sum_{i=0}^{q} \tilde{b}_i X^i$, it is very probable that $\widetilde{\mathcal{P}}$ is no longer irreducible over $\mathbb{F}_2$. Hence, we can write $\widetilde{\mathcal{P}}$ as the product

$$\widetilde{\mathcal{P}} = \widetilde{\mathcal{P}}_1^{\,e_1} \cdots \widetilde{\mathcal{P}}_l^{\,e_l}$$

where $\widetilde{\mathcal{P}}_i$ are distinct irreducible polynomials in $\mathbb{F}_2[X]$ and e_i are positive integers.

From the structure of the class residue ring $\mathbb{F}_2[X]/(\widetilde{\mathcal{P}})$, we can collect information modulo the irreducible factors $\widetilde{\mathcal{P}}_i$. Let $\widetilde{\boldsymbol{P}}_i$ (resp. $\widetilde{\boldsymbol{Q}}_i$) be the representative of $\widetilde{\boldsymbol{P}}$ (resp. $\widetilde{\boldsymbol{Q}}$) viewed as as an element in the field $\mathbb{F}_{2^{\tilde{q}_i}} \cong \mathbb{F}_2[X]/(\widetilde{\mathcal{P}}_i)$ where $\tilde{q}_i = \deg(\widetilde{\mathcal{P}}_i)$, i.e., $\widetilde{\boldsymbol{P}}_i = \widetilde{\boldsymbol{P}} \bmod \widetilde{\mathcal{P}}_i$ (resp. $\widetilde{\boldsymbol{Q}}_i = \widetilde{\boldsymbol{Q}} \bmod \widetilde{\mathcal{P}}_i$). Therefore, the discrete logarithm of $\widetilde{\boldsymbol{Q}}_i$ w.r.t. $\widetilde{\boldsymbol{P}}_i$ in the group $\langle \widetilde{\boldsymbol{P}}_i \rangle \subseteq \widetilde{E}_i(\mathbb{F}_{2^{\tilde{q}_i}})$ is $d \bmod r_i$, where $r_i = \mathrm{ord}_{\widetilde{E}_i}(\widetilde{\boldsymbol{P}}_i)$ and $\widetilde{E}_i = \widetilde{E}_i(\widetilde{a}_{2,i}, \widetilde{a}_{6,i})$ with

$$\widetilde{a}_{2,i} \equiv a_2 \pmod{\widetilde{\mathcal{P}}_i(X)} \text{ and } \widetilde{a}_{6,i} \equiv y^2 + xy + x^3 + a_2 x^2 \pmod{\widetilde{\mathcal{P}}_i(X)} \ .$$

Hence, Chinese remaindering on each subgroup $\widetilde{E}_i$ yields the value of

$$d \bmod \mathrm{lcm}(r_1, \ldots, r_l) \ .$$

Remark that, when $\tilde{q}_i$ is composite, the computation of discrete logarithms in $\widetilde{E}_i(\mathbb{F}_{2^{\tilde{q}_i}})$ can be speeded up by Weil descent [11] (see also [16, 19, 23] for a thorough analysis and [10] for recent developments).

Unknown error The above analysis implies that the error on the representation of $\widetilde{\mathbb{K}}$ is known. When only a portion of the representation of $\mathbb{K}$ is damaged, we choose for $\widetilde{\mathbb{K}}$ the candidate best matching $\mathbb{K}$. In some cases, the error on (the representation of) $\mathbb{K}$ is completely random so that it is no longer possible to determine the resulting $\widetilde{\mathbb{K}}$. We can merely restrict $\widetilde{\mathbb{K}}$ as a set obtained by combining several known prime factors (case $\mathbb{K} = \mathbb{F}_p$) or irreducible polynomials (case $\mathbb{K} = \mathbb{F}_{2^q}$).

In this paragraph, we take advantage of some particularities of elliptic curves recommended for practical implementations to recover the right combination leading to $\widetilde{\mathbb{K}}$ when the 'best-matching' strategy does not apply.

Over prime fields, the security of elliptic curve cryptography does not depend on the form of the primes. For example, in order to optimize the efficiency of field arithmetic, the elliptic curves recommended by the NIST [2][6] are defined over $\mathbb{F}_p$ where p is a (generalized) Mersenne prime, that is, a prime of the form

$$p = 2^{\omega_0} - \sum_{i=1}^{B} \pm 2^{\omega_i}$$

[6] The 5 NIST elliptic curves over large prime fields are defined with the Mersenne primes $2^{192} - 2^{64} - 1$ (curve P-192), $2^{224} - 2^{96} + 1$ (curve P-224), $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$ (curve P-256), $2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$ (curve P-384) and $2^{521} - 1$ (curve P-521) [2].

where B is chosen small [24]. Hence, if p is Mersenne prime with $B \le 4$ (this includes all NIST curves), it can be stored much more economically as $\{\omega_0, \sigma_1\|\omega_1, \sigma_2\|\omega_2, \sigma_3\|\omega_3, \sigma_4\}$ and reconstructed in working memory as

$$p = 2^{\omega_0} + \sum_{i=1}^{3} \sigma_i\, 2^{\omega_i} + \sigma_4 \tag{6}$$

with $\sigma_1, \sigma_2, \sigma_3 \in \{-1, 0, 1\}$ and $\sigma_4 \in \{-1, 1\}$. In this case, an error on p means an error on ω_0, σ_4 and/or $\sigma_i\|\omega_i$ for some $1 \le i \le 3$. Then, instead of testing the combinations of factors of Δ having the most bits matching those of p (cf. § 3.2), we can conclude that the faulty $\tilde{p}$ is a number obtained as a combination of the factors of Δ, whose product can be written as a Mersenne number (non necessarily prime) satisfying Eq. (6). This additional assumption drastically reduces the number of candidates for $\tilde{p}$ and may allow to recover the value of $\tilde{p}$ even if the error is random (i.e., unknown).

A similar assumption holds for binary fields. For every q up to 1000, there exists an irreducible pentanomial $\mathcal{P}(X) = X^q + X^{q_1} + X^{q_2} + X^{q_3} + 1$ to represent the elements of $\mathbb{F}_{2^q}$ as polynomials in $\mathbb{F}_2[X]/(\mathcal{P})$ [1, § A.8]. Since the reduction of polynomials modulo a pentanomial is efficient, we may assume that only pentanomials are accepted for representing $\mathbb{F}_{2^q}$ (again, this includes all NIST curves[7]) and that they are stored in a compact way as $\{q, q_1, q_2, q_3\}$. An error on $\mathcal{P}$ translates in an error on q, q_1, q_2 and/or q_3, which can be easilier recovered by testing the factors of Δ (cf. § 3.2) that combine into a pentanomial $\widetilde{\mathcal{P}}(X) = X^{\tilde{q}} + X^{\widetilde{q_1}} + X^{\widetilde{q_2}} + X^{\widetilde{q_3}} + 1$.

3.3 Faults in the curve parameters

As parameter a_6 is not needed in the addition formulæ, a modification of its value does not affect the computation of $d\boldsymbol{P}$. So we only consider the situation of an error occurring in curve parameters a_1, a_2, a_3 or a_4.

To fix the ideas, we assume that parameter a_4 is faulty (the other cases are similar). We let $\widetilde{a}_4$ denote the corrupted value. As previously, since a_6 is not employed in the addition formulæ, computations are performed on the curve $\widetilde{E}(a_1, a_2, a_3, \widetilde{a}_4, \widetilde{a}_6)$. Furthermore, since both $\boldsymbol{P} = (x, y)$ and $\widetilde{\boldsymbol{Q}} = d\boldsymbol{P} = (\tilde{x}_d, \tilde{y}_d)$ lie on the curve $\widetilde{E}$, we have the system of equations

$$\begin{cases} \widetilde{a}_4\, x + \widetilde{a}_6 = y^2 + a_1\, xy + a_3\, y - x^3 - a_2\, x^2 \\ \widetilde{a}_4\, \tilde{x}_d + \widetilde{a}_6 = \tilde{y}_d^2 + a_1\, \tilde{x}_d \tilde{y}_d + a_3\, \tilde{y}_d - \tilde{x}_d^3 - a_2\, \tilde{x}_d^2 \end{cases}$$

whose resolution (over $\mathbb{K}$) gives the values of $\widetilde{a}_4$ and $\widetilde{a}_6$.

[7] The 10 NIST elliptic curves over binary fields are defined with the irreducible polynomials $X^{163} + X^7 + X^6 + X^3 + 1$ (curves K- and B-163), $X^{233} + X^{74} + 1$ (curves K- and B-233), $X^{283} + X^{12} + X^7 + X^5 + 1$ (curves K- and B-283), $X^{409} + X^{87} + 1$ (curves K- and B-409) and $X^{571} + X^{10} + X^5 + X^2 + 1$ (curves K- and B-571) [2].

Remark 5. In the (very) unlikely event where the two equations are linearly dependent, the process can be re-iterated with another scalar multiplication.

After resolving the system of equations for $\tilde{a}_4$ and $\tilde{a}_6$, we compute the logarithm of $\widetilde{\boldsymbol{Q}}$ w.r.t. $\boldsymbol{P}$ in $\langle \boldsymbol{P} \rangle \subseteq \widetilde{E}(a_1, a_2, a_3, \widetilde{a}_4, \widetilde{a}_6)$ and get the value of $d \bmod r$, where $r = \mathrm{ord}_{\widetilde{E}}(\boldsymbol{P})$.

4 Concluding Remarks

We have shown that a (permanent) fault in the system parameters may enable one to recover the value of $d \pmod r$. However, it is fairly easy to avoid the leakage of $d \pmod r$ by checking the parameters for faults prior to the computation of $\boldsymbol{Q} = d\boldsymbol{P}$. This can be done by adding a CRC to each system parameter. Then, after reading a system parameter in non-volatile memory, its CRC is computed and compared with the CRC stored in non-volatile memory. Another possibility is to use curve parameter a_6 as an integrity check (i.e., a_6 is used to verify whether the coordinates of point $\boldsymbol{P} = (x, y)$ satisfy, over $\mathbb{K}$, the relation $y^2 + a_1\, xy + a_3\, y - x^3 - a_2\, x^2 - a_4\, x = a_6$).

To prevent the leakage of $d \pmod r$ in the presence of a transient fault, we have shown that the system parameters actually used (i.e., in working memory) must also be checked during the computation of $\boldsymbol{Q} = d\boldsymbol{P}$. For the transient faults, it is recommended to perform the check before outputting $\boldsymbol{Q}$.

More importantly, our analysis teaches that not only the secret parameters (e.g., the private key) but also *the public parameters must be checked for faults.*

Acknowledgements The first author would like to thank David Naccache, Philippe Proust and Jean-Jacques Quisquater for giving him the opportunity of doing this work at Gemplus.

References

1. IEEE Std 1363-2000. *IEEE Standard Specifications for Public-Key Cryptography.* IEEE Computer Society, August 29, 2000.
2. Federal Information Processing Standards Publication FIPS 186-2. *Digital Signature Standard (DSS)*, appendix 6: "Recommended elliptic curves for federal government use". National Institute of Standards and Technology, January 27, 2000. Available at URL `http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf`.
3. F. Bao, R.H. Deng, Y. Han, A. Jeng, A.D. Narasimbalu, and T. Ngair. Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols*, volume 1361 of *Lecture Notes in Computer Science*, pages 115–124. Springer-Verlag, 1997.
4. Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential fault attacks on elliptic curve cryptosystems. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 131–146. Springer-Verlag, 2000.

5. Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In B.S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer-Verlag, 1997.
6. Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer-Verlag, 1997.
7. Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of eliminating errors in cryptographic computations. *Journal of Cryptology*, 14(2):101–119, 2001. An earlier version appears in [6].
8. Erik De Win, Serge Mister, Bart Preneel, and Michael Wiener. On the performance of signature schemes based on elliptic curves. In J.-P. Buhler, editor, *Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*, pages 252–266. Springer-Verlag, 1998.
9. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):469–472, July 1985.
10. Steven D. Galbraith, Florian Hess, and Nigel P. Smart. Extending the GHS Weil descent attack. In L. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 29–44. Springer-Verlag, 2002.
11. Pierrick Gaudry, Florian Hess, and Nigel P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, 15(1):19–46, 2002.
12. Marc Joye, Jean-Jacques Quisquater, Feng Bao, and Robert H. Deng. RSA-type signatures in the presence of transient faults. In M. Darnell, editor, *Cryptography and Coding*, volume 1355 of *Lecture Notes in Computer Science*, pages 155–160. Springer-Verlag, 1997.
13. Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.
14. Paul Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In N. Koblitz, editor, *Advances in Cryptology – CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer-Verlag, 1996.
15. Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.
16. Markus Maurer, Alfred Menezes, and Edlyn Teske. Analysis of the GHS Weil descent attack on the ECDLP over characteristic two finite fields of composite degree. In C. Pandu Rangan and C. Ding, editors, *Progress in Cryptology – INDOCRYPT 2001*, volume 2247 of *Lecture Notes in Computer Science*, pages 195–213. Springer-Verlag, 2001.
17. Alfred J. Menezes. *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, 1993.
18. Alfred Menezes, Tatsuaki Okamoto, and Scott Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39:1639–1646, 1993.
19. Alfred Menezes and Minghua Qu. Analysis of the Weil descent attack of Gaudry, Hess and Smart. In D. Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 308–318. Springer, 2001.

20. Victor S. Miller. Use of Elliptic Curves in Cryptography. In H.C. Williams, editor, *Advances in Cryptology - Proceedings of CRYPTO 1985*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 1986.
21. John M. Pollard. Monte Carlo methods for index computation (mod p). *Mathematics of Computation*, 32:918–924, 1978.
22. John M. Pollard. Kangaroos, monopoly and discrete logarithms. *Journal of Cryptology*, 13(4):437–447, 2000.
23. Nigel P. Smart. How secure are elliptic curves over composite extension fields? In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 30–39. Springer-Verlag, 2001.
24. Jerome A. Solinas. Generalized Mersenne numbers. Technical Report CORR-99-39, Dept of C&O, University of Waterloo, Canada, 1999.