SMART-CARD IMPLEMENTATION OF ELLIPTIC CURVE CRYPTOGRAPHY AND DPA-TYPE ATTACKS

Marc Joye
Gemplus, Card Security Group
La Vigie, Avenue des Jujubiers, ZI Athélia IV, 13705 La Ciotat Cedex, France
marc.joye@gemplus.com

Abstract This paper analyzes the resistance of smart-card implementations of elliptic curve cryptography against side-channel attacks, and more specifically against attacks using differential power analysis (DPA) and variants thereof. The use of random curve isomorphisms is a promising way (in terms of efficiency) for thwarting DPA-type for elliptic curve cryptosystems but its implementation needs care.

Various generalized DPA-type attacks are presented against *improper* implementations. Namely, a second-order DPA-type attack is mounted against an additive variant of randomized curve isomorphisms and a "refined" DPA-type attack against a more general variant. Of independent interest, this paper also provides an exact analysis of second-order DPA-type attacks.

Keywords: Smart-card implementations, elliptic curve cryptography, side-channel analysis, DPA-type attacks.

1. Introduction

With shorter key lengths, elliptic curve cryptography has received increased commercial acceptance and is already available in several smart-card products. It is supported by several standardization bodies, including ANSI, IEEE, ISO and NIST.

Because they better fit the constrained environment of smart cards, elliptic curve cryptosystems are particularly relevant to smart-card implementations. Until recently, efficient implementation meant fast running time and small memory requirements. Nowadays, an efficient implementation must also be protected against attacks and more particularly against side-channel attacks (e.g., based on timing analysis (TA) [11] or on simple/differential power analysis (SPA/DPA) [12]).

Two classes of elliptic curves are mainly used in cryptography: (non-supersingular) elliptic curves over binary fields (a.k.a. binary elliptic curves) and elliptic curves over large prime fields. The former class may be preferred for smart card implementations as arithmetic in characteristic two can be made very efficient [6, 14] (especially in hardware). See also [8].

In this paper, we show how to implement in a *proper* yet efficient way countermeasures against DPA-type attacks for binary elliptic curve cryptosystems. Of independent interest, we also provide an exact analysis of second-order DPA [12, 13].

The rest of this paper is organized as follows. In the next section, we briefly review known countermeasures meant to prevent DPA-type attacks. In Section 3, we detail the additive and multiplicative variants of point randomization using curve isomorphisms. We point out that the additive variant may succumb to a DPA-type attack if the slope, resulting in the addition of two elliptic curve points, is implemented in a straightforward way. Next, we mount a second-order DPA-type attack against another additive variant using a randomized slope in Section 4. In Section 5, we describe an attack against the multiplicative variant. This attack also applies to the more general randomized curve isomorphisms, combining both the additive and the multiplicative variants. Finally, we conclude in Section 6.

2. DPA-type Countermeasures

The basic operation in elliptic curve cryptography is the point multiplication: on input point P and scalar k, point $Q = [k]P$ is returned. DPA-type countermeasures include the randomization of k and/or P.

Let E be a nonsupersingular elliptic curve over $GF(2^n)$ given by the (short) Weierstrass equation

$$E : y^2 + xy = x^3 + a_2x^2 + a_6 \tag{1}$$

and let P and $Q = [k]P$ be points on E.

The usual way to randomize k in the computation of $Q = [k]P$ consists in adding a random multiple of the order of E (or of $\mathrm{ord}_E(P)$) [5]:

$$k^* := k + r\#E$$

for a random r and then Q is evaluated as $Q = [k^*]P$. Another option is to split k in two (or several) shares [2, 4] (see also [17]): $k = k_1^* + k_2^*$ with $k_1^* = k - r$ and $k_2^* = r$ for a random r and then $Q = [k_1^*]P + [k_2^*]P$. Further countermeasures dedicated to Koblitz curves (i.e., curves given by Eq. (1) with $a_2 \in GF(2)$ and $a_6 = 1$) are presented in [9, 10].

Input:	k and $P \in E$
Output:	$Q = [k]P$
1.	Choose a random curve isomorphism φ;
2.	Compute $P^* = \varphi(P)$ on $E^* = \varphi(E)$;
3.	Compute $Q^* = [k]P^* \in E^*$;
4.	Return $Q = \varphi^{-1}(Q^*) \in E$.

Figure 1. Randomized evaluation of $Q = [k]P$.

Point P can be randomized using a randomized projective representation [5] or a randomized field or curve isomorphism [10]. The use of randomized curve isomorphisms leads to better performances (and is easier to implement) than the use of randomized field isomorphisms. Furthermore, it better suits binary curves as it allows to represent points with affine coordinates (and so runs faster [6, 14]). However, as we will demonstrate in the next section, its implementation needs care.

3. Randomized Curve Isomorphisms

Using randomized curve isomorphisms, a point P on an elliptic curve E is randomized as $P^* = \varphi(P)$ on $E^* = \varphi(E)$, for a random curve isomorphism φ and then $Q = [k]P$ is evaluated as $\varphi^{-1}([k]P^*)$.

More specifically, over $GF(2^n)$, a point $P = (x_P, y_P)$ on the elliptic curve

$$E : y^2 + xy = x^3 + a_2x^2 + a_6$$

is mapped to point $P^* = (u^2x_P + r, u^3y_P + u^2s\,x_P + t)$ on the isomorphic curve

$$E^* : y^2 + a_1^*xy + a_3^*y = x^3 + a_2^*x^2 + a_4^*x + a_6^* \tag{2}$$

where

$$\begin{cases} a_1^* &= u \\ a_2^* &= u^2a_2 + us + r + s^2 \\ a_3^* &= ur \\ a_4^* &= ut + r^2 \\ a_6^* &= u^6a_6 + u^2r^2a_2 + u(r^2s + rt) + r^3 + t^2 + r^2s^2 \end{cases} \tag{3}$$

and $r, s, t, u \in GF(2^n)$ with $u \neq 0$ (see [15, Table III.1.2]).

As already noted in [10], the short Weierstrass equation (i.e., Eq. (2) where $a_1^* = 1$ and $a_3^* = a_4^* = 0$) cannot be used for E^* as this implies $u = 1$ and $r = t = 0$, and hence let unchanged the x-coordinate of point P^*: $x(P^*) = x_P$.

3.1 Additive randomization of P

In [3], Ciet and Joye overcome the above limitation by working on the extended Weierstrass equation

$$y^2 + xy + a_3^* y = x^3 + a_2^* x^2 + a_4^* x + a_6^* , \tag{4}$$

which, from Eq. (3), corresponds to $u = 1$ (and thus $a_3^* = r$). For simplicity, they also set the value of s to 0. As a result, point $P = (x_P, y_P)$ is randomized into

$$P^* = (x_P + r, y_P + t) \ . \tag{5}$$

Although both the x- and y-coordinates of P are now randomized, this technique cannot be used naively in a point multiplication algorithm. Indeed, a closer look at the addition formulas shows that the slope given by the chord-and-tangent law, remains invariant.

Let $P_1^* = (x_1^*, y_1^*)$ and $P_2^* = (x_2^*, y_2^*)$ (with $P_1^* \neq -P_2^*$) denote points on the randomized elliptic curve given by Eq. (4). Then the sum $P_3^* = P_1^* + P_2^*$ is defined as (x_3^*, y_3^*) with

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

where $\lambda^* = \dfrac{y_1^* + y_2^*}{x_1^* + x_2^*}$ when $x_1^* \neq x_2^*$ and $\lambda^* = \dfrac{x_1^{*2} + a_4^* + y_1^*}{x_1^* + a_3^*}$ otherwise (see [15, III.2.3c]). Recall that (i) we are working in characteristic two, (ii) $x_i^* = x_i + r$ and $y_i^* = y_i + t$ for $i \in \{1, 2\}$, and (iii) $u = 1$, $r = a_3^*$ and $s = 0$. Hence, we see that the slope

$$\lambda^* = \begin{cases} \dfrac{y_1^* + y_2^*}{x_1^* + x_2^*} = \dfrac{y_1 + t + y_2 + t}{x_1 + r + x_2 + r} = \dfrac{y_1 + y_2}{x_1 + x_2} & \text{when } x_1^* \neq x_2^* \\ \dfrac{x_1^{*2} + a_4^* + y_1^*}{x_1^* + a_3^*} = \dfrac{x_1^2 + r^2 + t + r^2 + y_1 + t}{x_1 + r + r} = x_1 + \dfrac{y_1}{x_1} & \text{otherwise} \end{cases}$$

does not depend on randoms r and t (we write λ the corresponding value), and consequently may be subject to a DPA-type attack (see e.g. [5]).

Randomly choosing s. The first idea that comes to mind is to randomly choose $s \in GF(2^n)$, leading to the more general randomization,

$$P^* = (x_P + r, y_P + sx_P + t) \ . \tag{6}$$

In this case, the slope λ^* involved in the addition of $P_1^* = (x_1^*, y_1^*)$ and $P_2^* = (x_2^*, y_2^*)$ becomes

$$\lambda^* = \begin{cases} \dfrac{y_1^* + y_2^*}{x_1^* + x_2^*} = \dfrac{y_1 + sx_1 + t + y_2 + sx_2 t}{x_1 + r + x_2 + r} \\ \qquad = \dfrac{y_1 + y_2}{x_1 + x_2} + s & \text{when } x_1^* \neq x_2^* \\ \dfrac{x_1^{*2} + a_4^* + y_1^*}{x_1^* + a_3^*} = \dfrac{x_1^2 + r^2 + t + r^2 + y_1 + sx_1 + t}{x_1 + r + r} \\ \qquad = x_1 + \dfrac{y_1}{x_1} + s & \text{otherwise} \end{cases}$$
$$= \lambda + s$$

and hence is masked with the value of s. It should be noted that the evaluation of λ^* needs to be carefully implemented. In particular, field registers cannot contain the values of non-randomized values (e.g., $x_1 + x_2$) as otherwise a DPA-type attack could still be mounted.

Unfortunately, as we will see in Section 4, such an implementation may succumb to a second-order DPA-type attack. It should however be noted that second-order DPA-type attacks are much harder to mount since the attacker needs to know where/when certain operations are done.

3.2 Multiplicative randomization of P

The coordinates of point $P = (x_P, y_P)$ can also be randomized in a multiplicative way. Randomly choosing $u \neq 0$ and $r = s = t = 0$, point P becomes

$$P^* = (u^2 x_P, u^3 y_P) \tag{7}$$

on the isomorphic curve

$$y^2 + u\,xy = x^3 + (a_2 u^2)x^2 + a_6 u^6 \ . \tag{8}$$

Let $P_1^* = (x_1^*, y_1^*)$ and $P_2^* = (x_2^*, y_2^*)$ (with $P_1^* \neq -P_2^*$) denote points on the randomized elliptic curve given by Eq. (8). Then the sum $P_3^* = P_1^* + P_2^*$ is defined as (x_3^*, y_3^*) with

$$x_3^* = \lambda^{*2} + \lambda^* u + a_2 u^2 + x_1^* + x_2^* \quad \text{and} \quad y_3^* = \lambda^*(x_1^* + x_3^*) + y_1^* + ux_3^*$$

where $\lambda^* = \dfrac{y_1^* + y_2^*}{x_1^* + x_2^*}$ when $x_1^* \neq x_2^*$ and $\lambda^* = \dfrac{x_1^{*2} + uy_1^*}{ux_1^*}$ otherwise. Again, denoting by λ the slope corresponding to the addition of P_1 and P_2 on the initial curve, we see that λ^* is multiplicatively blinded: $\lambda^* = u\lambda$.

4. A Second Order DPA-type Attack

Higher-order side-channel attacks [12] combine several samples within a single side-channel trace.

To ease the presentation, we consider the simplified Hamming-weight model for power leakage [13]. This model assumes that the instantaneous power consumption (C) is linearly related to the Hamming weight (H):

$$\mathsf{C} = \varepsilon \mathsf{H} + \ell \tag{9}$$

for some constants ε and ℓ.

Using the additive randomization (see Eq. (5) or (6)), point $P = (x_P, y_P)$ on the elliptic curve E is randomized as $P^* = (x_{P^*}, y_{P^*}) = (x_P + r, y_P + s x_P + t)$ on the isomorphic elliptic curve $E^* = \varphi(E)$ given by Eq. (4). Point $Q = [k]P$ is then evaluated as $Q = \varphi^{-1}([k]P^*)$. Recall that the attacker's goal is to recover the value of $k = \sum_{i=0}^{m} k_i 2^i$ (or a part thereof) during the evaluation of Q. W.l.o.g., we assume that the point multiplication is carried out with the left-to-right binary algorithm and that the attacker already knows the leading bits of k: $k_m, k_{m-1}, \ldots, k_{t+1}$ with $t < m$. He now wants to know the value of the next bit of k, namely k_t.

Let $\mathsf{C}^{(r)}$ represent the instantaneous power consumption when random r is drawn in $GF(2^n)$. Let also $\mathsf{C}^{(x_{P^*})}$ represent the instantaneous power consumption when the x-coordinate of point $P^* \in E^*$ is handled in a register. For any point $P^* \in E^*$, we can write $x_{P^*} = x_P \oplus r$ since addition in $GF(2^n)$ is equivalent to a bit-wise XOR operation. From this observation, the attacker guesses that $k_t = 1$ and produces two equal-size sets, $\mathcal{S}_0$ and $\mathcal{S}_1$, of random points, defined as

$$\mathcal{S}_b = \big\{ P \in E \mid g\big(x([2K_{t+1} + 1]P)\big) = b \big\} \quad \text{with } b \in \{0, 1\}$$

where $K_{t+1} = \sum_{i=t+1}^{m} k_i 2^{i-(t+1)}$ and g is a Boolean selection function returning for $g\big(x([2K_{t+1} + 1]P)\big)$ the value of a given bit (in the representation) of the x-coordinate of point $[2K_{t+1} + 1]P$. Let $R := [K_t]P = (x_R, y_R)$ and $R^* = \varphi(R) = (x_R + r, y_R + s x_R + t)$. The next step consists in computing the two average differential power consumptions (in absolute value),

$$\Delta_0 := \big\langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \big\rangle_{\mathcal{S}_0} \quad \text{and} \quad \Delta_1 := \big\langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \big\rangle_{\mathcal{S}_1}$$

and the second-order DPA operator

$$\Delta^{(2)} := \Delta_1 - \Delta_0 \ .$$

If $\Delta^{(2)} \not\approx 0$ (i.e., if there are DPA peaks) then the guess of the attacker was right, i.e., $k_t = 1$; otherwise the attacker deduces that $k_t = 0$. See Appendix A for a detailed proof. The attack proceeds iteratively in the same way until the value of k_0 is recovered.

5. A Refinement of Goubin's Attack

In [7], Goubin observes that the x-coordinate of a point $P = (x_P, y_P)$ with $x_P = 0$ is not randomized using the multiplicative randomization (see Eq. (7)). The same holds when considering the y-coordinate of point $P = (x_P, y_P)$ with $y_P = 0$. Over binary fields, elliptic curve points on Weierstrass curve (1) with their x-coordinate equal to 0 are points of order two. They are easily avoided with the cofactor variant in cryptographic protocols [16, Section 3]. Points of large order with their y-coordinate equal to 0 can also be defended against by using the Montgomery ladder as the y-coordinate is not used in this point multiplication algorithm [16, Section 5].

Another way for thwarting Goubin's attack is to use the more general randomization $P^* = (u^2 x_P + r, u^3 y_P + u^2 s\, x_P + t)$ (see e.g. [1, § 2.3]). We will show that this method succumbs to a "refined" power analysis.

We assume that the cofactor variant is not applied and so points with their x-coordinate equal to 0 are valid inputs to the point multiplication algorithm. An elliptic curve over $GF(2^n)$ always possesses a point of order two. Namely, the point $P_2 = (0, {a_6}^{2^{n-1}})$ satisfies Weierstrass equation

$$y^2 + xy = x^3 + a_2 x^2 + a_6 \ .$$

As in the attack of the previous section, we assume w.l.o.g. that the point multiplication, $Q = [k]P$, is evaluated with a left-to-right binary algorithm. The attacker's goal is to recover the value of k_t in the binary representation of scalar $k = \sum_{i=0}^{m} k_i\, 2^i$.

Define $K_t = \sum_{i=t}^{m} k_i\, 2^{i-t}$. The attacker guesses that $k_t = 1$ and repeatedly feeds the point multiplication algorithm with point $P_2 = (0, a_6^{2^{m-1}})$. As P_2 is of order two, it is worth noting that $R := [K_t]P_2 = [K_t \bmod 2]P_2 = [k_t]P_2$. Hence, when $k_t = 1$, it follows that

$$R^* := \varphi(R) = \varphi(P_2) = (r, u^3 a_6^{2^{m-1}} + t)$$

for some randoms r, t and u.

Next, the attacker computes the average differential power consumption (remember that the computation of $Q = [k]P$ is randomized),

$$\Delta^{(1)} := \left\langle \left| C^{(x_{R^*})} - C^{(r)} \right| \right\rangle$$

where $\mathsf{C}^{(x_{R^*})}$ and $\mathsf{C}^{(r)}$ denote the instantaneous power consumptions when the x-coordinate of point $R^* = \varphi(R)$ is handled and when r is randomly drawn from $GF(2^n)$, respectively. With the (idealized) model given by Eq. (9), this yields $\Delta^{(1)} \approx 0$ when $k_t = 1$. If $\Delta^{(1)} \not\approx 0$ then the attacker can deduce that $k_t = 0$.

6. Conclusion

This paper analyzed the resistance of elliptic curve cryptosystems against DPA-type attacks. Several new attacks were mounted against implementations improperly using randomized curve isomorphisms as a means for thwarting DPA-type attacks.

Since all the attacks presented in this paper require averaging several side-channel traces with the *same* input multiplier, the lesson is that point randomization techniques should be always used in conjunction with multiplier randomization techniques.

Acknowledgments

The author would like to thank to Francis Olivier, Pascal Paillier and Berry Schoenmakers for useful comments.

Appendix: Second-Order DPA Peaks

This appendix explains in further details why second-order DPA peaks reveal the value of the multiplier-bit k_t in a second-order DPA. We use the notations of Section 4. We have to show that $\Delta^{(2)} \not\approx 0$ means that $k_t = 1$ and 0 otherwise.

Proof. Assume that $k_t = 1$. Then we have $K_t = 2K_{t+1} + 1$ and $R = [2K_{t+1} + 1]P$. Hence, we get

$$\begin{aligned} \left\langle \left| \mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)} \right| \right\rangle_{S_0} &= |\epsilon| \cdot \left\langle \left| \mathsf{H}^{(x_{R^*})} - \mathsf{H}^{(r)} \right| \right\rangle_{S_0} \\ &\approx |\epsilon| \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \end{aligned}$$

and

$$\left\langle \left| \mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)} \right| \right\rangle_{S_1} \approx |\epsilon| \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right]$$

provided that the number of power consumption traces is sufficiently large. Moreover, we have

$$\begin{aligned} &\mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \\ &= \Pr[g(r) = 0] \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0, g(r)=0}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] + \\ &\qquad \Pr[g(r) = 1] \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0, g(r)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \\ &= \frac{1}{2} \mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] + \frac{1}{2} \mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \\ &= \mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] . \end{aligned}$$

We also have

$$\begin{aligned} &\mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^n} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \\ &= \frac{1}{2} \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] + \frac{1}{2} \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] \\ &= \frac{1}{2} \mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] + \frac{1}{2} \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] . \end{aligned}$$

As $\mathrm{E}_{x_R, r \in \{0,1\}^n} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] > \mathrm{E}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right]$, this implies

$$\mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] > \mathop{\mathrm{E}}_{x_R, r \in \{0,1\}^{n-1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right]$$

and thus

$$\mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=1}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] > \mathop{\mathrm{E}}_{\substack{x_R, r \in \{0,1\}^n \\ g(x_R)=0}} \left[\left| \mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \right| \right] .$$

This in turn implies

$$\langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \rangle_{\mathcal{S}_1} \gtrapprox \langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \rangle_{\mathcal{S}_0}$$

and consequently $\Delta^{(2)} \not\approx 0$.

On the contrary, when $k_t = 0$, both sets $\mathcal{S}_0$ and $\mathcal{S}_1$ behave as random (i.e., uncorrelated) sets. Therefore, provided that the number of power consumption traces is sufficiently large, we have

$$\langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \rangle_{\mathcal{S}_1} \approx \langle |\mathsf{C}^{(x_{R^*})} - \mathsf{C}^{(r)}| \rangle_{\mathcal{S}_0}$$

and so $\Delta^{(2)} \approx 0$. □

From

$$\begin{cases} \mathrm{E}_{x_R, r \in \{0,1\}^n}[\mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \mid g(x_R) = 0] = 0 \\ \mathrm{E}_{x_R, r \in \{0,1\}^n}[\mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \mid g(x_R) = 1 \text{ and } g(r) = 0] = 1 \\ \mathrm{E}_{x_R, r \in \{0,1\}^n}[\mathsf{H}^{(x_R \oplus r)} - \mathsf{H}^{(r)} \mid g(x_R) = 1 \text{ and } g(r) = 1] = -1 \end{cases},$$

an analysis similar to the one given in [13] would erroneously deduce that $\Delta^{(2)} \approx |\epsilon|$ when $k_t = 1$ and $\Delta^{(2)} \approx 0$ otherwise. The conclusion, however, remains correct.

References

[1] R.M. Avanzi. Countermeasures against differential power analysis for hyperelliptic curve cryptosystems. In C.D. Walter, Ç.K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 366–381. Springer-Verlag, 2003.

[2] S. Chari, C.S. Jutla, J.R. Rao, and P. Rohatgi. Towards sound approaches to counteract power-analysis attacks. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer-Verlag, 1999.

[3] M. Ciet and M. Joye. (Virtually) free randomization techniques for elliptic curve cryptography. In S. Qing, D. Gollmann, and J. Zhou, editors, *Information and Communications Security (ICICS 2003)*, volume 2836 of *Lecture Notes in Computer Science*, pages 348–359. Springer-Verlag, 2003.

[4] C. Clavier and M. Joye. Universal exponentiation algorithm: A first step towards provable SPA-resistance. In Ç.K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 300–308. Springer-Verlag, 2001.

[5] J.-S. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES '99*, volume 1717 of *Lecture Notes in Computer Science*, pages 292–302. Springer-Verlag, 1999.

[6] E. De Win, S. Mister, B. Preneel, and M. Wiener. On the performance of signature schemes based on elliptic curves. In J.P. Buhler, editor, *ANTS-3: Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 252–266. Springer-Verlag, 1998.

[7] L. Goubin. A refined power analysis attack on elliptic curve cryptosystems. In Y.G. Desmedt, editor, *Public Key Cryptography - PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 199–211. Springer-Verlag, 2003.

[8] D. Hankerson, J. López Hernandez, and A. Menezes. Software implementation of elliptic curve cryptography over binary fields. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 1–24. Springer-Verlag, 2000.

[9] M.A. Hasan. Power analysis attacks and algorithmic approaches to their countermeasures for Koblitz cryptosystems. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 93–108. Springer-Verlag, 2000.

[10] M. Joye and C. Tymen. Protections against differential analysis for elliptic curve cryptography: An algebraic approach. In Ç.K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 377–390. Springer-Verlag, 2001.

[11] P. 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.

[12] P.C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. Wiener, editor, *Advances in Cryptology - CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.

[13] T.S. Messerges. Using second-order power analysis to attack DPA resistant software. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 238–251. Springer-Verlag, 2000.

[14] R. Schroeppel, H. Orman, S. O'Malley, and O. Spatscheck. Fast key exchange with elliptic curve systems. In D. Coppersmith, editor, *Advances in Cryptology - CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 43–56. Springer-Verlag, 1995.

[15] J.H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, 1986.

[16] N.P. Smart. An analysis of Goubin's refined power analysis attack. In C.D. Walter, Ç.K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 281–290. Springer-Verlag, 2003.

[17] E. Trichina and A. Bellezza. Implementation of elliptic curve cryptography with built-in countermeasures against side channel attacks. In B.S. Kaliski Jr., Ç.K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 98–113. Springer-Verlag, 2003.