State-of-the-art of secure ECC implementations: a survey on known side-channel attacks and countermeasures

Junfeng Fan[†], Xu Guo[‡], Elke De Mulder[†], Patrick Schaumont[‡], Bart Preneel[†] and Ingrid Verbauwhede[†]
[†] *Katholieke Universiteit Leuven, ESAT/SCD-COSIC and IBBT*
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
[‡] *Bradley Department of Electrical and Computer Engineering*
Virginia Tech, Blacksburg, VA 24061, USA

***Abstract*—Implementations of cryptographic primitives are vulnerable to physical attacks. While the adversary only needs to succeed in one out of many attack methods, the designers have to consider all the known attacks, whenever applicable to their system, simultaneously. Thus, keeping an organized, complete and up-to-date table of physical attacks and countermeasures is of paramount importance to system designers.**

This paper summarizes known physical attacks and countermeasures on Elliptic Curve Cryptosystems. Instead of repeating the details of different attacks, we focus on a systematic way of organizing and understanding known attacks and countermeasures. Three principles of selecting countermeasures to thwart multiple attacks are given. This paper can be used as a road map for countermeasure selection in a first design iteration.

***Keywords*-Side-channel attacks; Elliptic curve Cryptosystems;**

I. Introduction

Traditional cryptanalysis assumes that an adversary only has access to input and output pairs, but has no knowledge about internal states of the device. However, the advent of side channel analysis showed that a cryptographic device can leak critical information. By monitoring the timing, power consumption, electromagnetic (EM) emission of the device or by inserting faults, adversaries can gain information about internal data or operations and extract the key out of the cryptographic device without mathematically breaking the primitives.

With new tampering methods and new attacks being continuously proposed and accumulated, designing a *secure* cryptosystem becomes increasingly difficult. While the adversary only needs to succeed in one out of many attack methods, the designers have to prevent all the applicable attacks simultaneously. Moreover, countermeasures of one attack may surprisingly benefit another attack. As a result, keeping abreast of the most recent developments in the field of implementation attacks and with the corresponding countermeasures is a never ending task.

In this paper we provide a systematic overview of implementation attacks and countermeasures of one specific cryptographic primitive: Elliptic Curve Cryptography (ECC). However, this paper has no intention to propose new attacks or new countermeasures. Instead, we describe several general principles for countermeasure selection. The survey in this paper can be used as a tool for selecting countermeasures in a first design iteration.

This survey has been influenced by Avanzi's report [1] and the books by Blake et al. [2] and Avanzi et al. [3]. All of them give an excellent overview of side-channel attacks on ECC and HECC up to their point of publication. This paper, however, differs from previous work in at least three aspects. Firstly, it includes recently reported attacks such as carry-based attack [4]. Secondly, we focus on the interaction of known attacks and countermeasures in a systematic way. Thirdly, this survey proposes some guidelines for selecting countermeasures. We would like to stress that, just as what was stressed in previous reports [1]–[3], perfect (fully secure and low-cost) countermeasures do not exist up to now.

The rest of this paper is organized as follows. Section II gives a short introduction on the background of ECC and implementation attacks. Section III and IV gives details on known passive and active attacks on the Elliptic Curve Scalar Multiplication (ECSM), respectively. In Section V, we discuss a systematic way to select countermeasures. Section VI gives some research directions on this topic. We conclude the paper in Section VII.

II. Background

We give a brief introduction to Elliptic Curve Cryptography and implementation attacks in this section. A comprehensive introduction to ECC can be found in [2], [3]. For a thorough summary of power analysis attacks, by far the most popular class of implementation attacks, we refer the reader to [5].

Throughout this paper we assume the notations below are defined as follows:

- $\mathbb{K}$: a finite field;
- $char(\mathbb{K})$: the characteristic of $\mathbb{K}$;
- $E(a_1, a_2, a_3, a_4, a_6)$: an elliptic curve with coefficients a_1, a_2, a_3, a_4, a_6;
- $P(x, y)$: a point with coordinates (x, y);
- $\mathcal{O}$: point at infinity;

978-1-4244-7812-5/10/$26.00 ©2010 IEEE

- $E(\mathbb{K})$: a group formed by the points on an elliptic curve E defined over the finite field $\mathbb{K}$;
- $\#E$: the number of points on the curve E, i.e. the order of the curve E;
- $weak$ curve: a curve whose order does not have big prime divisors;
- the order of point P: the smallest integer r such that $rP = \mathcal{O}$;
- coordinate system: a system to represent a point in an n-dimensional space;
- affine coordinates: a point is represented with a two-tuple of numbers (x, y);
- projective coordinates: a point (x, y) is represented as (X, Y, Z), where $x = X/Z, y = Y/Z$;
- Jacobian projective coordinates: a point (x, y) is represented as (X, Y, Z), where $x = X/Z^2, y = Y/Z^3$.

A. Elliptic Curve Cryptosystems

An elliptic curve E over a field $\mathbb{K}$ can be defined by a *Weierstrass* equation.

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad (1)$$

where $a_1, a_2, a_3, a_4, a_6 \in \mathbb{K}$ and $\Delta \neq 0$. Here Δ is the discriminant of E. A *Weierstrass* equation can be simplified by applying change of coordinates. If $char(\mathbb{K})$ is not equal to 2 or 3, then E can be transformed to

$$y^2 = x^3 + ax + b \quad (2)$$

where $a, b \in \mathbb{K}$. If $char(\mathbb{K}) = 2$, then E can be transformed to

$$y^2 + xy = x^3 + ax^2 + b \quad (3)$$

if E is non-supersingular.

For cryptographic use, we are only interested in elliptic curves over a finite field. Elliptic curves defined over both prime fields and binary extension fields are used in reality. Given two points, $P(x_1, y_1)$ and $Q(x_2, y_2)$, the sum of P and Q is again a point on the same curve under the addition rule. The set of points (x, y) on E together with the point at infinity form an abelian group. The security of ECC is based on the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP), namely, finding out k for two given points P and Q such that $Q = kP$. The variable k is called the scalar. In most cases the secrecy of k is protected.

B. Scalar Multiplication

A cryptographic device for ECC is supposed to perform scalar multiplication efficiently and securely.

Given the a point $P \in E(\mathbb{K})$ and a scalar k, the computation kP is called point multiplication or scalar multiplication. Algorithm 1 shows the Left-To-Right binary method for scalar multiplication.

Algorithm 1 Left-To-Right (downwards) binary method for point multiplication

Input: $P \in E(\mathbb{F})$ and integer $k = \sum_{i=0}^{l-1} k_i 2^i$.
Output: kP.

1: $R \leftarrow \mathcal{O}$.
2: **for** $i = l - 1$ downto 0 **do**
3: $\quad R \leftarrow 2R$.
4: $\quad$ If $k_i = 1$ then $R \leftarrow R + P$.
5: **end for**

Return R.

Given two points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ on an elliptic curve E defined over binary extension fields, one can compute $P_3(x_3, y_3) = P_1 + P_2$ as follows:

$$\begin{aligned} x_3 &= \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \\ y_3 &= -y_1 - (x_3 - x_1)\lambda - a_1x_3 - a_3 \end{aligned}$$

where

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

These formulas use coordinates in affine form, and they require the division operation. Because the coordinates are represented as finite-field elements, this division operation needs to be implemented as an finite-field inversion, a costly and complex operation. For example, in $GF(2^{163})$, one inversion corresponds to 8-10 finite field multiplications even when using an efficient algorithm such as Itoh-Tsujii [6].

C. Implementations and Physical attacks

Cryptographic transformations can be implemented in both software and hardware. While software implementations, running on general purpose microprocessors, are flexible and can be easily updated, hardware implementations, either on FPGAs or ASICs, can achieve higher performance.

Figure 1 shows the architecture of an ECC processor. Note that each component here may refer to different types of realizations. For example, the Arithmetic Logic Unit (ALU) can be a standard ALU of a general purpose processor or a dedicated field multiplier. The temporary storage can be a RAM or a register file. A non-volatile memory, e.g. flash ROM, is normally used to store curve parameters.

An ECSM process starts with loading certain configurations (the definition of the curve, the underlying field, the coordinate system, the base point P) and the scalar k. While the base point P can be read either from the ROM or from outside, the scalar k is normally stored or generated inside the chip and should be protected. The output point, $Q = kP$, is not completely visible from outside. For example, ElGamal decryption algorithm only returns the x-coordinate of Q.

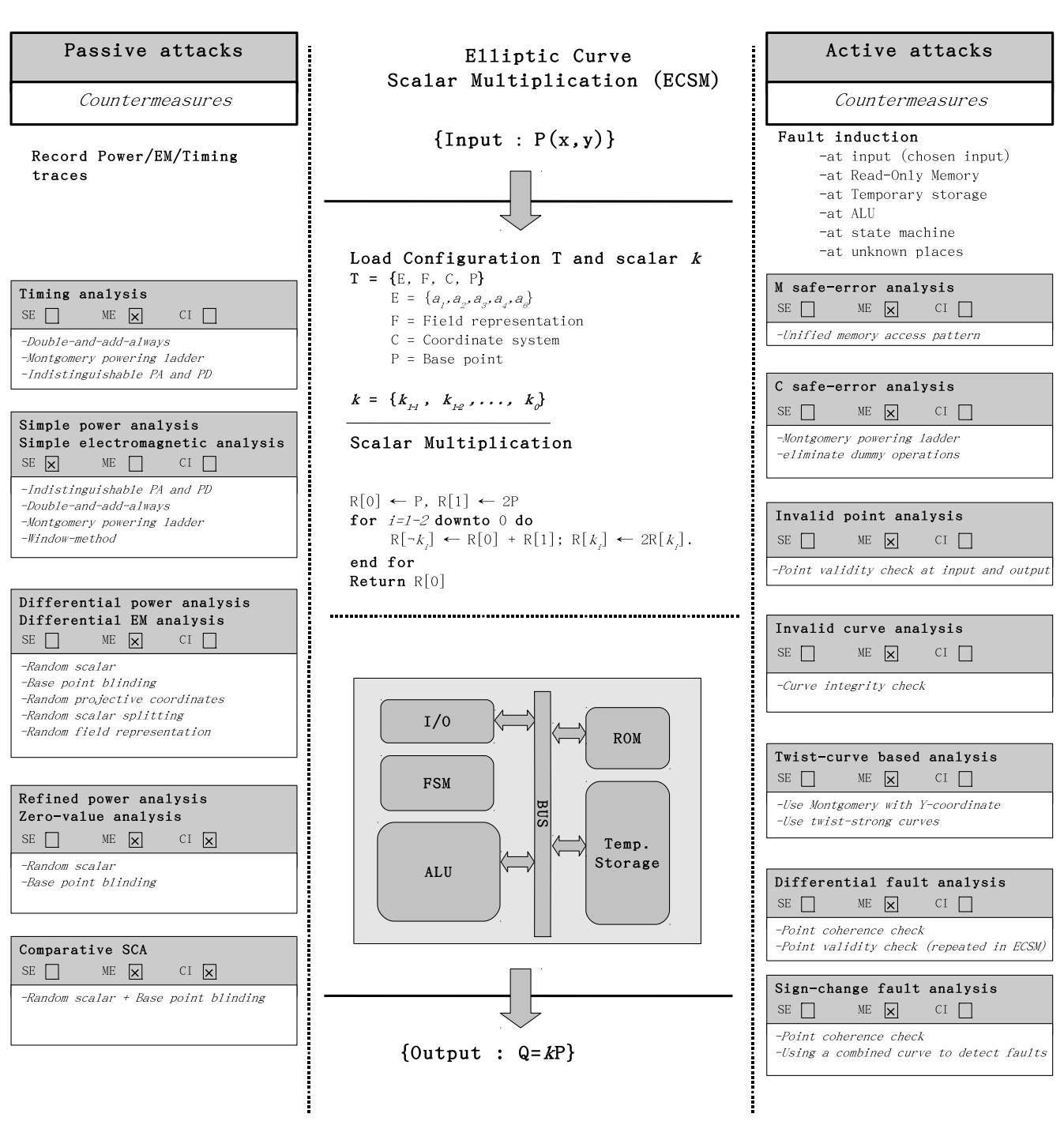

Figure 1: Elliptic curve processor architecture and related physical attacks. (SE = Single Execution, ME = Multiple Executions, CI = Chosen Input)

In practice, execution of ECSM leaks information of k in many ways. Figure 1 also shows various side-channel attacks on ECSM. We group the diversity of attacks into two main categories: passive and active attacks. Passive attacks do not require meddling with the device's input/outputs or working environment. Active attacks on the other hand try to induce and exploit abnormal behavior in the device.

An important criterion to judge the cost of a specific side-channel attack is how many executions are required to reveal the complete key stream. In Fig. 1, each attack is tagged with either SE (Single Execution) or ME (Multiple Executions). Another important criterion is that some attacks, such as doubling attack and refined power analysis, require the freedom of choosing the base point while some do not. The base point can be fixed or stored internally in some implementations, which makes attacks with this requirement significantly harder to mount.

In order to counteract passive side-channel attacks at least one of the following links has to be removed:

1) The relation between the data or operations inside the device and the physical leakages (e.g. power traces, EM radiation traces, timing, etc.).
2) The relation between the hypothetical data and the actual data calculated in the device.

With respect to power analysis, there are two methods to achieve this: rendering the power consumption constant (e.g., using a special logic style [7]) or randomizing the intermediate data during the scalar multiplication.

In order to counteract fault attacks, two types of methods are used: error-detection and error-tolerance. The first method detects faults inserted in the elliptic curve parameters or the point multiplications. If faults are detected, the execution is aborted. The second method chooses an elliptic curve such that even if faults are inducted in the scalar multiplication, the adversary can not derive scalar from the faulty results. For example, twist-strong curves are error-tolerant under twist-curve attack.

III. Passive Attacks and Countermeasures

An adversary has a wide range of choice in attack strategies.

A. Timing Attacks and Simple Side-Channel Analysis

Timing attacks exploit the timing variance with different inputs [8]. Careless implementations contain a vast number of sources of timing leakage. For example, timing variations can be caused by RAM cache or conditional branches. Although no papers have been published about a practical timing attack on ECC, many papers do mention the threat and provide the reader with suitable countermeasures.

Cryptographic implementations are vulnerable to simple power analysis attacks if the power traces show distinctive key-dependent patterns [9]. Alg. 1 shows the "double-and-add" algorithm for a point multiplication. The value of a key bit can be revealed if the adversary can tell the difference between point doubling and point addition from a power trace.

Algorithm 2 Add-and-double-always point multiplication [10]

Input: $P \in E(\mathbb{F})$ and integer $k = \sum_{i=0}^{l-1} k_i 2^i$.
Output: kP.

1: $R[0] \leftarrow \mathcal{O}$.
2: **for** $i = l - 1$ downto 0 **do**
3: $\quad R[0] \leftarrow 2R[0]$, $R[1] \leftarrow R[0] + P$.
4: $\quad R[0] \leftarrow R[k_i]$.
5: **end for**

Return $R[0]$.

The *double-and-add-always* algorithm, introduced in [10], ensures that the sequence of operations to compute a scalar multiplication is independent of the value of the secret scalar through insertion of a dummy point additions. Another way to prevent simple SCA is making point addition and doubling indistinguishable. For example, dummy operations can be added at the field arithmetic level. This has the advantage of less overhead. On the other hand, the Hamming weight of the secret scalar might still leak.

Instead of making the group operations indistinguishable, one can rewrite them as sequences of side-channel atomic blocks that are indistinguishable for simple SCAs [11]. Implementations based on the Montgomery ladder [12]–[14], shown as Alg. 3, are protected against timing attacks and simple SCA since the execution time of the scalar multiplication is inherently unrelated to the Hamming weight of the secret scalar.

The last type of countermeasures is the usage of unified formulae for point doubling and addition [15]. Unified point addition formulae use a single formula to calculate both the doubling and the addition, resulting in an single sequence of operations for both.

B. Template Attacks

A template attack [16] requires access to a fully controllable device, and proceeds on two phases. In the first phase, the profiling phase, the attacker constructs a precise

Algorithm 3 Montgomery powering ladder [12]

Input: $P \in E(\mathbb{F})$ and integer $k = \sum_{i=0}^{l-1} k_i 2^i$.
Output: kP.

1: $R[0] \leftarrow P$, $R[1] \leftarrow 2P$.
2: **for** $i = l - 2$ downto 0 **do**
3: $\quad R[\neg k_i] \leftarrow R[0] + R[1]$, $R[k_i] \leftarrow 2R[k_i]$.
4: **end for**

Return $R[0]$.

model of the wanted signal source, including a characterization of the noise. The second phase comprises the actual attack. Because of their reliance on data dependencies, template attacks exploit the so-called differential SCA type of leakage. The attack assumes that most of the side-channel information resides in the variance. So far not much research has been done on template attacks for public key algorithms. Medwed and Oswald [17] showed the feasibility of this type of attacks on an implementation of the ECSDA algorithm. In [18] a template attack on a masked Montgomery ladder implementation is presented. Template attacks, if feasible, are a major threat. Neither the double-and-add-always algorithm, nor blinding the scalar or base point resist template attacks. In fact, only randomizing the coordinates provides protection.

C. Differential Side-Channel Analysis

Differential side-channel attacks (DPA for differential power analysis and DEMA for differential electromagnetic analysis) use statistical techniques to pry the secret information out of the measurements [9]. A differential attack adheres to a fixed working principle: a cryptographic device, supplied with the fixed secret key k, is sequentially fed with N input points P_i, $i \in \{1, 2, .., N\}$. During the encryption of input P_i under key k, a measurement over time of the side-channel $m_i(t)$ is recorded and stored. The attacker then chooses an intermediate value of the algorithm which depends both on the input point P_i and a small part of the secret key k. For each key candidate k' for the partial key and for each input point P_i, the attacker calculates the intermediate value and transforms it to a hypothetical leakage value $L_{k',i}$ with the aid of a hypothetical leakage model. For the correct key guess $k' = k$ there will be a correlation between the measurements $m_i(t)$ and the hypothetical leakages $L_{k,i}$ at some time instance t. This relation is uncovered by using statistical distinguishers such as a difference of means test, Pearson correlation or Spearman's rank correlation.

A straightforward countermeasure against differential SCA is randomizing the intermediate data, thereby rendering the calculation of the hypothetical leakage values rather impossible. Coron [10] suggested three countermeasures to protect against differential SCA attacks:

1) Blinding the private scalar by adding a multiple of $\#E$. For any random number r and $k' = k + r\#E$, we have $k'P = kP$ since $(r\#E)P = \mathcal{O}$.
2) Blinding the point P, such that kP becomes $k(P + R)$. The known value $S = kR$ is subtracted at the end of the computation.
3) Randomizing the homogeneous projective coordinates (X, Y, Z) with a random $\lambda \neq 0$ to $(\lambda X, \lambda Y, \lambda Z)$. The random variable λ can be updated in every execution or after each doubling or addition.

Very similar, Joye and Tymen [19] suggested to make use of an elliptic curve isomorphism of the fixed curve or of an isomorphic representation of the field. Ciet and Joye [20] also suggested several similar randomization methods.

1) Random key splitting: $k = k_1 + k_2$ or $k = \lfloor k/r \rfloor r + (k \bmod r)$ for a random r.
2) Randomized EC isomorphism.
3) Randomized field isomorphism. We refer to the corresponding paper for a detailed explanation [19].

Coron's first two defense strategies were scrutinized in [21] and judged weak if implemented as presented. The latter three countermeasures are broken by an RPA attack in [22]. (See subsection III-E).

D. Comparative Side-Channel Attacks

Comparative SCA resides between a simple SCA and a differential SCA. Two portions of the same or different leakage trace are compared to discover the reuse of values. The umbrella term was introduced in [23], but the first reported attack belonging to this category is the doubling attack. The doubling attack [24] on ECC is an attack with chosen inputs and has been shown powerful to attack some classic SPA-protected algorithms such as left-to-right (downward) double-and-add-always algorithm. The attacker does not need to tell whether a computation being performed is a point doubling or addition. More precisely, for two point doublings $(2\times t_1)P$ and $(2\times t_2)P$, even if the attacker cannot tell the exact values of t_1 or t_2, the attacker can still detect if $t_1 = t_2$.

To thwart this attack, blinding techniques can be effective. Care has to be taken however that neither blinding the base point or the scalar is applied solely. This has been proven insecure [24]. Combined use strengthens the security.

E. Refined Power Analysis

A refined side-channel analysis attack (RPA is short for Refined Power Analysis) directs its attention to the existence of a point P_0 on the elliptic curve $E(\mathbb{K})$ such that one of the coordinates is 0 in $\mathbb{K}$ and $P_0 \neq \mathcal{O}$. Randomized projective coordinates, randomized EC isomorphisms and randomized field isomorphisms preserve this specific property of the point P_0. Feeding to a device a point P that leads to a special point $R(0, y)$ (or $R(x, 0)$) at step i under the assumption of some specific key bits will generate exploitable side-channel leakage [22], [25].

The attack can be thwarted by using either a cofactor variant of a protocol for points of "small order" or by using isogenous curves for points of "large order". The zero-value point attack (ZPA) generalizes this attack [26]: zero value points in intermediate results are also considered.

F. Carry-based Attack

The carry-based attack [4], reported by Fouque et al., does not attack the scalar multiplication itself but its countermeasures.

It relies on the carry propagation occurring when long-integer additions are performed as repeated sub-word additions. For instance, on an 8-bit processor, Coron's first countermeasure, $k' = k + r'$ where $r' = r\#E$, is normally performed with repeated 8-bit additions. Let k_i and r'_i denote the i^{th} sub-word of k and r', respectively. Note that k_i is fixed and r'_i is random in different executions. The crucial observation here is that, when adding k_i with r'_i, the probability of the carry out $c = 1$ depends solely on the value of k_i (the carry-in has negligible impact [4]). The adversary can then monitor the outgoing carry bit of the adder to estimate the probability of $c = 1$. With this probability, the value of k_i can be guessed with high confidence.

So far, no countermeasures have been proposed to thwart this attack.

G. EM Attacks

Most simple/differential analysis attacks and countermeasures summed up so far are based on power consumption leakage. Most often, electromagnetic radiation is considered as an extension of the power consumption leakage and the attacks/countermeasures are applied without change [27]. While this approach makes sense in most cases, electromagnetic radiation measurements can be made locally [4] and as such circumvent some countermeasures. Specifically crafted attacks or countermeasures for electromagnetic analysis have not been published.

IV. Fault Attacks and Countermeasures

Besides passive side-channel analysis, adversaries can actively disturb the cryptographic devices and use the erroneous output (or not even the output, but the reaction of the disturbed device) to derive the secret. In order to do so, the adversary needs to induce faults on the victim device. Various methods can be used, such as changing one memory bit with laser or violating setup time with glitches in clock. The difficulty in inducing a fault depends on its precision, both in time as well as in location. Random faults change an operation or a variable at some point during the execution of a cryptographic algorithm. Precise faults change a specific bit of a specific variable at a specific instance during the execution. Clearly, random faults are easier to introduce, and they are less costly, than precise faults.

In this section, we focus on fault attacks and countermeasures on ECSM. General tampering techniques and tamper-resistance methods will be briefly mentioned. Readers who are interested in these methods are referred to [28].

We divide fault attacks into three categories, namely, safe-error based analysis, weak-curve based analysis and differential fault analysis. Safe-error attacks are based on the observation that some errors will not change the results. Weak curve attacks try to move a scalar multiplication from a strong curve to a $weak$ curve. The differential fault attacks analyzes the difference between the correct output and erroneous output to retrieve the scalar bit-by-bit.

A. Safe-error analysis (M-type and C-type)

The concept of safe-error was introduced by Yen and Joye in [13], [29]. Two types of safe-error are reported: C safe-error and M safe-error. What makes safe-error analysis special is that the adversary is not interested in the erroneous results, but simply the fact that the output is affected or not.

1) C safe-error: The C safe-error makes use of dummy operations that are introduced to achieve SPA resistance. Taking the add-and-double-always algorithms (Alg. 2) as an example, the dummy addition in step 3 makes safe-error possible. The adversary can induce temporary faults in the ALU or memory during the dummy point addition. If the key bit, k_i, is 1, then final results will be faulty. Otherwise, the final results are not affected. The adversary can thus discover one key bit in one execution.

In order to thwart C safe-error analysis, dummy operations should be avoided. For example, instead of double-and-add-always algorithm, Montgomery's powering ladder should be used. If for certain reasons dummy operations can not be avoided, the key stream should be represented randomly in each point multiplication.

2) M safe-error: While the C safe-error attack explores the weakness of an algorithm, the M safe-error attack explores the possible safe-error in an implementation. The attack was first proposed by Yen and Joye [29] to attack RSA. However, it also applies to ECSM.

The basic observation of an M safe-error is that faults in some memory blocks will be cleared. Consider Alg. 3 as an example. We assume that a fault is inducted to y of $R[1]$ right after the calculation of λ during the point doubling in step 3. If $k_i = 1$, then the faults on y will be cleared. Otherwise, it propagates to the end of the ECSM. By simply checking whether the result is affected or not, the adversary can reveal k_i.

Joye and Yen [13] proposed a method to prevent this attack. The idea is to eliminate the possibility of inserting safe-errors. Using the modified Montgomery powering ladder [13], any fault in R[0] or R[1] will be detected regardless of the value of k_i.

B. Weak curve based analysis

In 2000, Biehl et al. [30] described a new type of fault attack on elliptic curve scalar multiplication. They observed that a_6 was not used in a point multiplication. An implementation of this algorithm for curve E generates correct results for any curve E' that differs from E only in a_6:

$$E' : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a'_6. \qquad (4)$$

Thus, the adversary can cheat an ECC processor with a point $P' \in E'(\mathbb{F})$ where E' is a cryptographically $weak$ curve.

If an ECC processor does not check whether the base point, P, is a valid point on the specified curve E or not, the adversary can then choose a point $P' \in E'(\mathbb{K})$ and get the result of the scalar multiplication, $Q' = kP'$. The adversary can then solve DLP in a subgroup of order $r_{P'}$ (the order of P') to retrieve $k_r = k \bmod r_{P'}$. This process can be repeated to generate k_r for different r. At the end, the Chinese Remainder Theorem can be used to retrieve k. This attack also shows us that not all the fault-based attacks require expensive equipments or sophisticated tampering techniques, and that a naive implementation can be broken with almost negligible cost.

The method of $moving$ a scalar multiplication from a strong curve E to a weak curve E' was then extended. With the help of faults, the adversary makes use of invalid points [30], invalid curves [31] and twist curves [32] to hit a weak curve. These methods are described below.

1) Invalid point attacks : The idea of the invalid point attack is to let the scalar multiplication start with a point P' of a weak curve.

If the ECSM is performed without checking the validity of the base point, then no faults need to be inducted. If the ECC processor checks the validity of the base point, the adversary will try to change the point P right after the point validity check. Note this attack requires fault induction at a specific timing, thus is much more difficult than the one described above.

For some applications such as EC El-Gamal or ECDSA, y_2 is not present on the output. In this case, the adversary needs to derive $\{E', P'(x'_1, y_1), Q'(x'_2, y_2)\}$ from $\{E, P(x_1, y_1), x'_2\}$. Though it looks difficult, the adversary still has a non-negligible probability to succeed. Readers who are interested can find the complete method in [30].

A possible countermeasure, as suggested in [30], [31], is Point Validation (PV) before and after scalar multiplication. PV checks if a point lies on an elliptic curve or not. If the base point or result does not belong to the original curve, no output should be given.

2) Invalid curve attacks: Ciet and Joye [31] refined the attack in [30] by loosening the requirements on fault injection. They show that any $unknown$ faults, including permanent faults in non-volatile memory or transient faults caused on the bus, in any curve parameters, including field representation and curve parameters a_1, a_2, a_3, a_4, may cause information leakage on the scalar k.

Ciet and Joye suggested using error checking codes to ensure the integrity of curve parameters before scalar multiplication.

3) Twist curve based FA: In 2008, Fouque et al. [32] discovered a new way to hit a possibly $weak$ curve, the quadratic twist curve. They observed that a point multiplication routine for some curve E, without using the y-coordinate, gives correct results for ECSM on its twist curve $\tilde{E}$. They also noticed that the twist curves of many cryptographically strong curves are cryptographically weak (see [32] for details). Eq.5 defines the twist curve of E, where ε is a quadratic non-residue in $\mathbb{F}_p$.

$$\tilde{E} : (\varepsilon)y^2 = x^3 + ax + b \tag{5}$$

For elliptic curves defined over $\mathbb{F}_p$, a random $x \in \mathbb{F}_p$ corresponds to a point on either E or its twist. Since the order of E and $\tilde{E}$ are close, the probability is approximatively one half that a random abscissa corresponds to a point on E or $\tilde{E}$. As a result, the adversary has a probability of one half to hit a point on $\tilde{E}$ with a random fault on x-coordinate of P on E.

There are three possible methods to thwart this attack. The first one is to repeat point validity check during the scalar multiplication. The second one is to use y-coordinate all the time. Both methods have some overhead in terms of computation time and storage. The third one is to choose twist-secure curves, namely, curves whose twist curve are also cryptographically strong.

C. Differential FA

The Differential Fault Attack (DFA) uses the difference between the correct results and the faulty results to deduce certain bits of the scalar.

1) Biehl-Meyer-Müller DFA: Biehl et al. [30] reported the first DFA on an ECSM. We use an right-to-left multiplication algorithm to describe this attack. Let Q_i and R_i denote the value of Q and R at the end of the i^{th} iteration, respectively. Let $k(i) = k$ div 2^i. Let Q'_i be the value of Q if faults have been induced. The attack reveals k from the Most Significant Bits (MSB) to the Least Significant Bits (LSB).

1) Run ECSM once and collect the correct result (Q_n).
2) Run the ECSM again and induce an one-bit flip on Q_i, where $l - m \leq i < l$. We assume that m is $small$.
3) Note that Q_n=Q_i+$(k(i)2^i)P$ and Q'_n=Q'_i+$(k(i)2^i)P$. The adversary then tries all possible $k(i) \in \{0, 1, .., 2^m - 1\}$ to generate Q_i and Q'_i. The correct value of $k(i)$ will result in a $\{Q_i, Q'_i\}$ that have only one-bit difference.

The attack works for left-to-right multiplication algorithm as well. It also applies if k is encoded with any other

Algorithm 4 Right-To-Left (upwards) binary method for point multiplication

Input: $P \in E(\mathbb{F})$ and integer $k = \sum_{i=0}^{l-1} k_i 2^i$.
Output: kP.

1: $R \leftarrow P$, $Q \leftarrow \mathcal{O}$.
2: **for** $i = 0$ to $l - 1$ **do**
3: If $k_i = 1$ then $Q \leftarrow Q + R$.
4: $R \leftarrow 2R$.
5: **end for**

Return R.

deterministic codes such as Non-Adjacent-Form (NAF) and w-NAF. It is also claimed that a fault induced at random moments during an ECSM is sufficient [30].

To thwart this attack, the validity of the intermediate results (Q_i and R_i in Algorithm 4) should be regularly checked. Another possible countermeasure is to randomize the scalar k such that the adversary can does not gain more bits of k in repeated executions.

2) Sign change FA: In 2006, Blömer et al. [33] proposed the sign change fault (SCF) attack. It attacks implementations where scalar is encoded in Non-Adjacent Form (NAF). When using curves defined over the prime field, the sign change of a point implies only a sign change of its y-coordinate. The SCF attack does not force the elliptic curve operations to leave the original group $E(\mathbb{F}_p)$, thus P is always a valid point.

A straightforward countermeasure against an SCF attack is to use Montgomery ladder algorithm that does not use the y-coordinate for computing ECSM (e.g. Montgomery Scalar Multiplication with López-Dahab coordinates [14]). Another countermeasure presented by Blömer et al. [33] uses a second elliptic curve whose order is a small prime number to verify the final results.

V. Selection of Countermeasures

One can not simply integrate all the countermeasures discussed above to thwart all attacks. The reasons for this are manifold. The complexity and extra overhead added by countermeasures can significantly increase the design and manufacturing cost. Another important reason is that a countermeasure against one attack may benefit another one. Thus, countermeasures should be carefully selected such that they do not add extra vulnerabilities. In this section, we discuss the cross relationship between known attacks and countermeasures.

A. Countermeasures versus Attacks

Table I summarizes the most important attacks and their countermeasures. The different attacks, grouped into passive attacks and active attacks are listed column-wise, while each row represents one specific countermeasure. Let A_j and C_i denote the attack in the j^{th} column and countermeasure in the i^{th} row, respectively. The grid (i, j), the cross of the i^{th} row and the j^{th} column, shows the relation between A_j and C_i.

- $\surd$: C_i is an effective countermeasure against A_j.
- $\times$: C_i is attacked by A_j.
- **H**: C_i helps A_j.
- **?**: C_i might be an effective countermeasure against A_j, but the relation between C_i and A_j is unclear or unpublished.
- **–**: C_i and A_j are irrelevant (C_i is not effective against A_j).

It is important to make a difference between $\times$ and **–**. Here $\times$ means C_i is attacked by A_j, where **–** means that the use of C_i does not affect the effort or result of A_j at all. For example, scalar randomization using 20-bit random number is attacked by doubling attack, so we put a $\times$ at their cross. The Montgomery powering ladder is designed to thwart SPA, and it does not make a DPA attack harder or easier, so we put a **–** there.

Below we discuss each countermeasure and its relation with the listed attacks.

Indistinguishable Point Addition Formulae Indistinguishable group operations render a simple SCA impossible, but only if the underlying field arithmetic is implemented securely. This is discussed in [36], [37]. This method does not counteract differential SCA and RPA/ZPA [38].

Double-and-add-always The double-and-add-always algorithm is the main representative of the countermeasures that use dummy instructions or operations to withstand simple side-channel attacks.

The algorithm fails against doubling attacks. It does not remove vulnerabilities to differential SCA attacks. It also makes C safe-error fault attack possible.

Montgomery Powering Ladder The Montgomery powering ladder is an algorithm level countermeasure running in a fixed time without redundant operations, hence it is SCA resistant. It avoids the usage of dummy instructions and also resists the $normal$ doubling attack. However, it is attacked by the relative doubling attack proposed by Yen et al. [34]. This attack can reveal the relation between two adjacent secret scalar bits, thereby seriously decreases the number of key candidates.

With Montgomery powering ladder, y-coordinate is not necessary during the scalar multiplication, which prevents sign-change attacks. However, for curves that have weak twist curves, using Montgomery powering ladder without y-coordinate is vulnerable to twist curve attacks.

Random scalar split. This countermeasure can resist DPA/DEMA attacks since it has a random scalar for each execution. In [24], the authors have already analyzed the effectiveness of Coron's first countermeasure against the doubling attack. If we assume that the scalar k is randomly split into two full length scalars, the search space is extended to 2^{81} for a 163-bit k (the birthday paradox applies here). This is enough to resist the doubling attack. It can also help to thwart RPA/ZPA if it is used together with base point randomization [22], [26], [39].

However, this countermeasure is vulnerable to a carry-based attack if the key is split as follows: choosing a random number $r < \#E$, and $k_1 = r$, $k_2 = k - r$.

Scalar randomization. With respect to the resistance against passive SCA, the above analysis of the random scalar split countermeasure against DPA/DEMA and RPA/ZPA also applies here. However, as mentioned in [24] the 20-bit random value for blinding the scalar k is not enough to

Table I: Attacks versus Countermeasures

	Passive Attacks							Active Attacks						
								Safe-Error		Weak Curve			Differential	
	TA	SPA/SEMA	Template Attack	DPA/DEMA	Comparative SCA	RPA/ZPA	Carry-based Attack	M Safe-Error	C Safe-Error	Invalid Point	Invalid Curve	Twist Curve	Sign Change	Differential
Indistinguishable Point Addition Formulae [15]	√	√	–	–	**?**	–	–	–	–	–	–	–	–	–
Double-and-add-always [10]	√	√	–	–	× [24]	–	–	–	×**H** [29]	–	–	–	–	–
Montgomery Powering Ladder⊢ [13]	√	√	–	–	× [34]	× [26]	–	√	√	–	–	**H** [32]	√	–
Montgomery Powering Ladder⊣ [13]	√	√	–	–	× [34]	× [26]	–	√	√	–	–	√	–	–
Random scalar split [20]	–	–	**?**	√	**?**	√	×	–	**?**	–	–	√	**?**	**?**
Scalar randomization [10]	–	–	× [17]	× [21]	× [24]	√	× [4]	–	**?**	–	–	–	**?**	**?**
Base point blinding [10]	–	–	× [17]	× [21]	× [24]	√	–	–	–	**?**	–	–	–	**?**
Random Projective Coordinates [10]	–	–	√	√	**?**	× [22]	–	–	–	–	–	–	–	**?**
Randomized EC Isomorphisms [20]	–	–	**?**	√	**?**	× [22]	–	–	–	–	–	–	–	**?**
Randomized Field Isomorphisms [20]	–	–	**?**	√	**?**	× [22]	–	–	–	–	–	–	–	**?**
Point validity check [30]	–	–	–	–	–	–	–	–	**H**	√	**?**	√	×**H** [33]	√
Curve integrity check [31]	–	–	–	–	–	–	–	–	–	**?**	√	–	–	–
Coherence check [35]	–	–	–	–	–	–	–	–	**H**	–	**?**	–	√	√

Let A_j and C_i denote the attack in the j^{th} column and countermeasure in the i^{th} row, respectively.

- √: C_i is an effective countermeasure against A_j.
- ×: C_i is attacked by A_j.
- **H**: C_i helps A_j.
- **?**: C_i might be an effective countermeasure against A_j, but the relation between C_i and A_j is unclear or unpublished.
- –: C_i and A_j are irrelevant (C_i is not effective against A_j).

⊣: using y-coordinate

⊢: without using y-coordinate

resist the doubling attack.

Like random scalar split, it renders the safe-error and sign-change attacks more difficult. On the other hand, it is shown in [4] that the key randomization process, namely, $k' = k + r\#E$, leaks the scalar under the carry-based attack.

Base point blinding. For an ECSM, the scalar randomization and base point blinding are based on the same idea of randomizing one component of the point multiplication. Therefore, their effectiveness against various passive attacks is similar. It can resist DPA/DEMA as explained in [10]. In [24], the authors conclude that this countermeasure is still vulnerable to the doubling attack since the point which blinds P is also doubled at each execution. This countermeasure makes RPA/ZPA more difficult since it can break the assumption that the attacker can freely choose the base point (the base point is blinded).

This countermeasure might make the weak-curve based attacks more difficult since the attacker does not know the masking point R. In an attack based on an invalid point, the adversary needs to find out the faulty points P' and $Q' = kP'$. With the point blinding, it seams to be more difficult to reveal either P' or Q'. However, in the case of an invalid curve attack, base point blinding does not make a difference.

Random projective coordinates. This countermeasure is effective against differential SCA. It fails to resist the RPA as zero is not effectively randomized. Combination with a simple SCA countermeasure is essential.

Point validity check. This countermeasure checks if a certain point is on the authentic curve or not. It is an effective countermeasure against invalid point attacks. If the y-coordinate is used, it is also effective against a twist-curve attack. However, it is not effective against invalid curve attacks, sign-change attacks and C safe-error attacks.

Curve integrity check. The curve integrity check is to detect fault injections on curve parameters. Before starting an ECSM the curve parameters will be read from the non-volatile memory (possibly on the data bus), which are vulnerable to permanent or transient faults. So, the integrity of the curve parameters (including the base point) needs to be verified using a CRC (cyclic redundancy check) code before an ECSM execution.

Coherence check. A coherence check verifies the intermediate or final results with respect to a valid pattern. If an ECSM uses the Montgomery powering ladder, we can use the fact that the difference between $R[0]$ and $R[1]$ is always P. This can be used to detect faults during an ECSM [35].

B. Selecting countermeasure.

After analyzing the existing attacks and countermeasures, a natural question is whether there exists a set of countermeasures that resists all the existing passive and active attacks. While unified countermeasures to tackle both the passive and active attacks are attractive, they are very likely weaker than what is expected. Baek and Vasyltsov extended Shamir's trick, which was proposed for RSA-CRT, to secure ECC from DPA and FA [40]. However, Joye showed in [41] that a non-negligible portion of faults was undetected using the unified countermeasure and settings in [40]. In this section, we describe several principles to choose countermeasures.

Complete: A complete picture of attacks and countermeasures is the perfect base to select countermeasures. As we pointed out above, an adversary needs to succeed in only one out of many possible attack methods to win. Keeping a summary of up-to-date attacks and countermeasures is important for cryptosystem designers.

Specific: Whenever selecting countermeasures for a cryptosystem, a detailed description of the cryptosystem should be explicitly defined. A set of countermeasures that can thwart all known attacks is neither easy to find nor efficient in terms of area and performance. Within restricted boundaries, countermeasure selection is much easier and more efficient. For example, RPA and comparative SCA assume that the attacker can choose the base point freely. If an ECC processor is targeting an application where the base point is fixed, then an RPA and doubling attack can not apply.

Additive: The selected countermeasures should be additive. Suppose that we choose countermeasures from Table I, we could proceed in two steps.

The first step is a column-wise selection. We inspect each column and select a countermeasure that suffices to thwart the attack in this column. If we have chosen two countermeasures, C_a and C_b, and their relation with A_j is as follows: $(a, j) = \surd$, $(b, j) = \times$. In this case, we need to study whether C_a covers C_b or not. **H** in the table should be avoided whenever possible. If eventually we can not get rid of all the **H**, extra countermeasures should be added to cover it.

The second step is to check if the selected countermeasures are additive. Using multiple countermeasures simultaneously might introduce new vulnerabilities. Thus, we need to evaluate the selected countermeasures as a new countermeasure.

VI. Outlook

Though physical security of cryptographic hardware or software has been intensively studied in the last ten years, known methods to protect physical attacks are far from satisfactory. For future research, we believe the following topics are important to improve our understanding in physical security of cryptosystems.

- Mathematical model to evaluate the effectiveness of attacks and countermeasures. For example, an attack requires certain amount of information leakage to reveal the scalar, which sets up an upper bound of information leakage for an effective countermeasure. Models that

allow a quantitative evaluation of physical information leakage are still missing.

- A framework to evaluate the effectiveness of a set of countermeasures. As shown above, multiple countermeasures are always used together to thwart multiple attacks. However, the current method for choosing countermeasures is rather ad-hoc.
- Grids containing **?** in Table I. Table I shows that we only understand a small part of the complete picture, and many interesting attack-countermeasure pairs have not been studied yet.
- A higher level approach to secure elliptic curve scalar multiplication. For example, many attacks are effective only when the same scalar is used for hundreds of times with different base points. A slight modification on high level protocols might prevent those attacks.
- System integration of multiple countermeasures. In [42], the researchers suggested a combined countermeasure and discussed the system integration cost. A perfect countermeasure is probably useless if it is too complex to implement.

VII. Conclusion

In this paper we give a systematic overview of the existing implementation attacks and countermeasures on ECC. While we have no intentions to provide new countermeasures, we do give a complete overview of the wide range of attacks and the common classes of countermeasures. We strongly believe that keeping track of the ever evolving field of implementation attacks is of crucial importance to a cryptosystem designer. This paper provides a digest of existing attacks and countermeasures.

Table I can be used for countermeasures selection. We also plan to keep this work updating in a more open environment (e.g. an ePrint version of this paper updated once new attacks and countermeasures are found) and extend it also for other similar cases, such as cryptographic parings.

Acknowledgment

This work was supported in part by K.U. Leuven-BOF (OT/06/40), by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy), by FWO project G.0300.07, by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II, and in part by the US National Science Foundation through grant 0644070 and 0541472, by Virginia Tech Pratt Fund.

References

[1] R. Avanzi, "Side Channel Attacks on Implementations of Curve-Based Cryptographic Primitives," Cryptology ePrint Archive, Report 2005/017, available from http://eprint.iacr.org/.

[2] I. Blake, G. Seroussi, N. Smart, and J. W. S. Cassels, *Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series)*. New York, USA: Cambridge University Press, 2005.

[3] R. M. Avanzi, H. Cohen, C. Doche, G. Frey, T. Lange, K. Nguyen, and F. Vercauteren, *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. CRC Press, 2005.

[4] P. Fouque, D. Réal, F. Valette, and M. Drissi, "The Carry Leakage on the Randomized Exponent Countermeasure," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 5154. Springer, 2008, pp. 198–213.

[5] S. Mangard, E. Oswald, and T. Popp, *Power analysis Attacks: Revealing the Secrets of Smart Cards*. Secaucus, NJ, USA: Springer, 2007.

[6] Y. Asano, T. Itoh, and S. Tsujii, "Generalised fast algorithm for computing multiplicative inverses in GF(2^m)," *Electronics Letters*, vol. 25, no. 10, pp. 664–665, 1989.

[7] K. Tiri and I. Verbauwhede, "A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation," in *DATE '04: Proceedings of the conference on Design, automation and test in Europe*. IEEE Computer Society, 2004, pp. 246–251.

[8] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *CRYPTO'96:Advances in Cryptology*, ser. LNCS, N. Koblitz, Ed., vol. 1109. Springer, 1996, pp. 104–113.

[9] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *CRYPTO*, ser. LNCS, vol. 1666. Springer, 1999, pp. 388–397.

[10] J. Coron, "Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems," in *Cryptographic Hardware and Embedded Systems, CHES*, ser. LNCS, vol. 1717. Springer, 1999, pp. 292–302.

[11] B. Chevallier-Mames, M. Ciet, and M. Joye, "Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity," *IEEE Trans. Computers*, vol. 53, no. 6, pp. 760–768, 2004.

[12] P. Montgomery, "Speeding the Pollard and elliptic curve methods of factorization," *Mathematics of Computation*, vol. 48, no. 177, pp. 243–264, 1987.

[13] M. Joye and S.-M. Yen, "The Montgomery Powering Ladder," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 2523. Springer, 2002, pp. 291–302.

[14] J. López and R. Dahab, "Fast Multiplication on Elliptic Curves over GF(2m) without Precomputation," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 1717. Springer, 1999, pp. 316–327.

[15] E. Brier and M. Joye, "Weierstraß Elliptic Curves and Side-Channel Attacks," in *Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems*, vol. 2274. Springer, 2002, pp. 335–345.

[16] S. Chari, J. R. Rao, and P. Rohatgi, "Template Attacks," in *Cryptographic Hardware and Embedded Systems, CHES*, ser. LNCS, vol. 2523, 2002, pp. 13–28.

[17] M. Medwed and E. Oswald, "Template Attacks on ECDSA," in *Information Security Applications, WISA*, vol. 5379, 2008, pp. 14–27.

[18] C. Herbst and M. Medwed, "Using Templates to Attack Masked Montgomery Ladder Implementations of Modular Exponentiation," in *Information Security Applications, WISA*, vol. 5379, 2008, pp. 1–13.

[19] M. Joye and C. Tymen, "Protections against Differential Analysis for Elliptic Curve Cryptography," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 2162. Springer, 2001, pp. 377–390.

[20] M. Ciet and M. Joye, "(Virtually) Free Randomization Techniques for Elliptic Curve Cryptography," in *Information and Communications Security (ICICS2006), LNCS 2836*. Springer, 2003, pp. 348–359.

[21] K. Okeya and K. Sakurai, "Power Analysis Breaks Elliptic Curve Cryptosystems even Secure against the Timing Attack," in *INDOCRYPT*, ser. LNCS, vol. 1977. Springer, 2000, pp. 178–190.

[22] L. Goubin, "A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems," in *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*, vol. 2567. Springer, 2003, pp. 199–210.

[23] N. Homma, A. Miyamoto, T. Aoki, A. Satoh, and A. Shamir, "Collision-based power analysis of modular exponentiation using chosen-message pairs," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 5154. Springer, 2008, pp. 15–29.

[24] P.-A. Fouque and F. Valette, "The Doubling Attack : Why Upwards Is Better than Downwards," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 2779. Springer, 2003, pp. 269–280.

[25] P.-Y. Liardet and N. P. Smart, "Preventing SPA/DPA in ECC Systems Using the Jacobi Form," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 2162. Springer, 2001, pp. 391–401.

[26] T. Akishita and T. Takagi, "Zero-Value Point Attacks on Elliptic Curve Cryptosystem," vol. 2851, pp. 218–233, 2003.

[27] E. D. Mulder, S. Örs, B. Preneel, and I. Verbauwhede, "Differential power and electromagnetic attacks on a FPGA implementation of elliptic curve cryptosystems," *Computers & Electrical Engineering*, vol. 33, no. 5-6, pp. 367–382, 2007.

[28] O. Kömmerling and M. Kuhn, "Design principles for tamper-resistant smartcard processors," in *USENIX workshop on Smartcard Technology – SmartCard'99*, 1999, pp. 9–20.

[29] S. M. Yen and M. Joye, "Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis," *IEEE Trans. Computers*, vol. 49, no. 9, pp. 967–970, 2000.

[30] I. Biehl, B. Meyer, and V. Müller, "Differential Fault Attacks on Elliptic Curve Cryptosystems," in *CRYPTO*, vol. 1880. Springer, 2000, pp. 131–146.

[31] M. Ciet and M. Joye, "Elliptic Curve Cryptosystems in the Presence of Permanent and Transient Faults," *Des. Codes Cryptography*, vol. 36, no. 1, pp. 33–43, 2005.

[32] P. Fouque, R. Lercier, D. Réal, and F. Valette, "Fault Attack on Elliptic Curve Montgomery Ladder Implementation," in *Fifth International Workshop on Fault Diagnosis and Tolerance in Cryptography - FDTC*, 2008, pp. 92–98.

[33] J. Blömer, M. Otto, and J.-P. Seifert, "Sign Change Fault Attacks on Elliptic Curve Cryptosystems," in *Fault Diagnosis and Tolerance in Cryptography(FDTC), LNCS 4236*. Springer, 2006, pp. 36–52.

[34] S.-M. Yen, L.-C. Ko, S.-J. Moon, and J. Ha, "Relative Doubling Attack Against Montgomery Ladder," in *Information Security and Cryptology, ICISC*, 2005.

[35] A. Dominguez-Oviedo, "On Fault-based Attacks and Countermeasures for Elliptic Curve Cryptosystems," Ph.D. dissertation, University of Waterloo, Canada, 2008.

[36] C. D. Walter, "Simple Power Analysis of Unified Code for ECC Double and Add," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 3156. Springer, 2004, pp. 191–204.

[37] D. Stebila and N. Thériault, "Unified Point Addition Formulæ and Side-Channel Attacks," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS, vol. 4249. Springer, 2006, pp. 354–368.

[38] T. Izu and T. Takagi, "Exceptional procedure attack on elliptic curve cryptosystems," in *Public Key Cryptography, PKC*, ser. LNCS, vol. 2567, 2003, pp. 224–239.

[39] J. Ha, J. Park, S. Moon, and S. Yen, "Provably Secure Countermeasure Resistant to Several Types of Power Attack for ECC," in *Information Security Applications (WISA)*, vol. 4867. Springer, 2007, pp. 333–344.

[40] Y.-J. Baek and I. Vasyltsov, "How to prevent dpa and fault attack in a unified way for ecc scalar multiplication c ring extension method," in *Information Security Practice and Experience(ISPEC2007), LNCS 4464*. Springer, 2007, pp. 225–237.

[41] M. Joye, "On the security of a unified countermeasure," in *FDTC '08: Proceedings of the 5th Workshop on Fault Diagnosis and Tolerance in Cryptography*. IEEE Computer Society, 2008, pp. 87–91.

[42] X. Guo, J. Fan, P. Schaumont, and I. Verbauwhede, "Programmable and Parallel ECC Coprocessor Architecture: Tradeoffs between Area, Speed and Security," in *Cryptographic Hardware and Embedded Systems - CHES*, ser. LNCS. Springer, 2009, pp. 289–303.