社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2005－ＣＳＥＣ－30 (10)
2005／7／21

Koblitz曲線におけるSPA防御法

桶屋 勝幸[†] 高木 剛[††] ヴィオムカミーユ[†]

† (株) 日立製作所 システム開発研究所 〒215-0013 川崎市麻生区王禅寺1099番地
†† 公立はこだて未来大学 〒041-8655 北海道函館市亀田中野町116番地2
E-mail: †{ka-okeya,camille}@sdl.hitachi.co.jp, ††takagi@fun.ac.jp

あらまし Koblitz曲線はバイナリ曲線の一つで、スカラー倍算を効率よく計算できるという特徴がある。そのため、ICカードなどの小型軽量装置への実装に適する。しかしながら、それら装置ではサイドチャネル攻撃が脅威となる。本稿では、Koblitz曲線においてサイドチャネル攻撃に対する防御法を2つ提案する。双方とも、単純電力解析 (SPA) への対策として固定パターンにエンコードする。ひとつは、既存防御法をKoblitz曲線向けに拡張したものである。もうひとつは、スカラー値を左から右 (left-to-right) にエンコードし、容易に格納することができる。また、ECDSAの署名生成向けにランダムにスカラー値を生成することもできる。

キーワード 楕円曲線暗号、Koblitz曲線、ICカード、サイドチャネル攻撃、SPA防御法

Defeating Simple Power Analysis on Koblitz Curves

Katsuyuki OKEYA [†], Tsuyoshi TAKAGI [††], and Camille VUILLAUME [†]

† Hitachi, Ltd., Systems Development Laboratory, Kawasaki, Japan.
†† Future University - Hakodate, Japan.
E-mail: †{ka-okeya,camille}@sdl.hitachi.co.jp, ††takagi@fun.ac.jp

Abstract Koblitz curves belong to a special class of binary curves on which the scalar multiplication can be computed very efficiently. For this reason, they are suitable candidates for implementations on low-end processors. However, such devices are often vulnerable to side channel attacks. In this paper, we propose two countermeasures against side channel attacks on Koblitz curves. Both of them utilize a fixed-pattern recoding to defeat simple power analysis. Our first technique extends a known countermeasure to the special case of Koblitz curves. In our second technique, the scalar is recoded from left to right, and can be easily stored or even randomly generated.

Key words elliptic curve cryptosystems, Koblitz curves, smartcard, side channel attacks, SPA countermeasure

1. Introduction

Since they have been proposed, elliptic curve cryptosystems (ECC) have been thoroughly studied, because in contrary to RSA-type cryptosystems, they are well-suited for implementations on memory-constrained devices and low-end processors. However, when no coprocessor is available to speed-up modular multiplications, elliptic curves defined over prime fields are still to slow for practical implementations on such low-end processors. One possible solution is to use elliptic curves defined over binary fields: such binary curves rely on simple and carry-less operations such as shift and xor. Indeed, binary curves can be implemented easily on general-purpose processors, and the simplicity of their atomic operations results in a considerable speed-up compared to elliptic curves defined over prime fields.

But some special binary curves called Koblitz curves are even faster. Indeed, on Koblitz curves, with the right representation of the scalar, it becomes possible to replace the elliptic doubling operation by much faster operations, namely squares. Since their efficient arithmetic has been pointed out [11], no significant security flaw or practical attack has be found on Koblitz curves.

Side channel attacks are powerful attacks which use a priori innocuous information such as timings or power consumption to break implementations of cryptosystems [10]. On light and specialized cryptodevices such as smartcards, side channel attacks are a major threat. There are two types of attack strategies based power consumption analysis: simple power analysis (SPA) and differential power analysis (DPA). In the

frame of SPA, the attacker uses only one power trace to guess the secret information, whereas he is allowed to use a statistical tool in order to extract information from several power traces in the frame of DPA [10].

On Koblitz curves, the standard DPA countermeasures can be deployed: among others, randomized projective coordinates, randomized scalar or randomized base point [3]. However, SPA resistance is problematic: the known countermeasures for general curves are either based on dummy operations [2], [3] or scalar recoding using properties of binary representations [15]. First, even though dummy-based countermeasures are straight-forwardly applicable, they should be avoided. Indeed, fault attacks can exploit dummy operations: one type of attacks is defeated, but only to let in a new powerful attack with devastating consequences [18]. Second, to benefit from the fast arithmetic of Koblitz curves, that is, replacing elliptic doublings by squares, one has to first recode the scalar to a new representation. On general curves, scalars are usually represented using a binary representation $d = \sum_{i=0}^{n-1} d_i 2^i$, or a close variant of it. However, on Koblitz curves, a totally different representation is deployed: $d = \sum_{i=0}^{n-1} d_i \tau^i$, where τ is solution of a quadratic equation. As a consequence, many tricks which manipulate the binary representation are not applicable anymore.

In this paper, we propose two new countermeasures against side channel attacks on Koblitz curves. The first technique extends the mechanisms of a countermeasure for general curves, namely the SPA-resistant NAF_w [15], to the special arithmetic of Koblitz curves. The original SPA-resistant NAF_w utilizes special properties of binary expansions to generate a secure representation. We show how to transpose the mechanisms of the countermeasure to Koblitz curves. The second technique utilizes a two-round recoding. First, it generates a zero-free representation using the principles of our first countermeasure. Second, it applies a windowing technique in order to take advantage of pre-computed points and consequently reduce computational costs. Then, we emphasize interesting properties of our schemes. On elliptic curves, left-to-right computations are usually faster, because they are compatible with the most efficient coordinate systems [4]. Thus, it is preferable to use a left-to-right recoding approach as well: the recoding and the scalar multiplication can be combined, and no memory is needed to store the scalar in multiple representations. We show practical situations where our ideas are compatible with a left-to-right recoding. First, when the scalar is fixed (e.g. EC-ElGamal decryption), using our representation, the scalar can be recoded (off-line) and stored once for all, and a windowing technique can be applied on the fly. In this situation, on-line computations are fully left-to-right. Second, when a secret ephemeral is needed (in EC-DSA signature generation or EC-DH), we can generate this ephemeral on the fly while computing the scalar multiplication, already recoded with our technique, allowing left-to-right strategies to work in harmony with the recoding stage. Therefore, in all practical situations where SPA-resistance is needed, our countermeasures can be deployed to protect secret information against side channel attacks, providing a high security level, great efficiency, smart and small memory usage.

2. Preliminaries

In this section, we discuss known facts: we introduce Koblitz curves and discuss the feasibility of side channel attacks on them.

2.1 Koblitz Curves

Koblitz curves are binary elliptic curves which offer a very efficient arithmetic with no significant security drawback [11]. They are defined over a binary field $\mathbb{F}_{2^m}$ by the equation: $\mathcal{E}_a = y^2 + xy = x^3 + ax^2 + 1$, where $a \in \{0, 1\}$. We denote by $\mathcal{E}_a(\mathbb{F}_{2^m})$ the additive group of the points of the elliptic curve over $\mathbb{F}_{2^m}$, along with the point of infinity $\mathcal{O}$, neutral element of the addition law. The main interest of Koblitz curves is that elliptic point doublings can be replaced by the efficiently computable Frobenius automorphism $\Phi : (x, y) \mapsto (x^2, y^2)$. Since the quadratic equation $(x^4, y^4) + 2(x, y) = \mu(x^2, y^2)$ where $\mu = (-1)^{1-a}$ holds for all points on the curve, the Frobenius map can be regarded as $\tau = (\mu + \sqrt{-7})/2$, solution of the equation $\Phi^2 = \mu\Phi - 2$. The Lucas sequence U_w is useful to compute with powers of τ:

$$U_0 = 0, U_1 = 1 \text{ and } U_{w+1} = \mu U_w - 2U_{w-1} \text{ for } w \geq 1. \quad (1)$$

The approach for fast computations over Koblitz curves is to convert a scalar d to a radix τ expansion such as $d = \sum_{i=0}^{j} d_i \tau^i$, $d_i \in \{0, \pm 1\}$. However, in order to fully take advantage of the Frobenius map, the τ expansion must be sparse and short. In [17], Solinas proposed two efficient algorithms to satisfy these properties: partial reduction modulo $\delta = (\tau^m - 1)/(\tau - 1)$ and radix-τ NAF recoding. Additionally, a width w radix τ NAF expansion (TNAF_w) can be constructed from the following map:

$$\Phi_w : u_0 + u_1 \cdot \tau \in \mathbb{Z}[\tau] \mapsto u_0 + u_1 \cdot t_w \bmod 2^w \in \mathbb{Z}/2^w\mathbb{Z}, \quad (2)$$

where $t_w = 2\frac{U_{w-1}}{U_w} \bmod 2^w$ and the notation mods 2^w refers to the signed representatives modulo 2^w. Interestingly, the elements $u_0 + u_1 \cdot \tau$ with odd u_0 correspond exactly to the odd representatives modulo 2^w under Φ_w. Thus, the map Φ_w defines (easily computable) congruence classes in the ring $\mathbb{Z}[\tau]$. Then, the digits of the TNAF_w are computed by iterating the procedure described in Algorithm 1.

Since the digits u of the TNAF_w belong to the set $u \in$

Algorithm 1: Conversion to TNAF_w [17]

INPUT: Scalar d, width w, per-curve parameters μ, $\delta = \frac{\tau^n - 1}{\tau - 1}$;
OUTPUT: $(d_l^{(w)}, \ldots, d_0^{(w)})$, TNAF_w representation of d;

(1) $u_0 \leftarrow d \bmod \delta$ with [17]; $u_1 \leftarrow 0$; $l \leftarrow 0$
(2) **while** $u_0 \neq 0$ or $u_1 \neq 0$ **do**
  (a) **if** $u_0 \bmod 2 \neq 0$ **then** $u \leftarrow \Phi(u_0 + \tau u_1)$ **else** $u \leftarrow 0$;
  (b) $u_0 \leftarrow u_0 - u$; $d_i \leftarrow u$; $r \leftarrow u_0/2$;
  (c) $u_0 \leftarrow u_1 + \mu r$; $u_1 \leftarrow -r$; $d_l^{(w)} \leftarrow u$; $l \leftarrow l + 1$;
(3) **return** $(d_{l-1}^{(w)}, \ldots, d_0^{(w)})$;

$\{\pm 1, \pm 3, \ldots, \pm(2^{w-1} - 1)\}$, only $2^{w-2} - 1$ non-trivial points must be pre-computed, with $2^{w-2} - 1$ point additions and one point doubling. Therefore, the total cost of the scalar multiplication using TNAF_w is on average:

$$\begin{aligned} C_{NAF} &= (m+a)\,\mathrm{ECFRB} \\ &+ \left(\tfrac{m+a}{w+1} + 2^{w-2} - 1\right)\mathrm{ECADD} + \mathrm{ECDBL} \end{aligned} \quad (3)$$

where ECFRB, ECADD and ECDBL stand for the computational cost of the Frobenius map, point additions and point doublings, respectively.

2.2 Side Channel Attacks on Koblitz Curves

Side channel attacks are a serious threat for light embedded devices running cryptographic applications: such devices often leak critical information through unexpected and a priori innocuous channels: timings or power consumption, for instance [10]. One can classify side channel attacks relying on power analysis into two categories [10]. The first class of attacks is called simple power analysis (SPA): in this situation, the attacker attempts to reveal the secret parameter with one single power trace. The second class is called differential power analysis (DPA): the attacker is allowed to gather several power traces and analyzes them with the help of a statistical tool.

In general, SPA on ECC utilizes the fact that point additions and doublings have different implementations, leading to different power traces [3]. By recognizing the operation chain from the power consumption, the attacker can reveal the secret information. To implement Koblitz curves, a polynomial basis is generally preferred. In this case, point doublings are replaced by the Frobenius map, whose computational cost is small but not negligible [7]. In other words, it is realistic to expect that the computation of one single Frobenius map can be detected within one power trace: in this case, SPA is straight-forwardly applicable.

In [3], Coron extended DPA to elliptic curve cryptosystems. On the one hand, DPA is not applicable to EC-DSA, where the base point is fixed and public, and the scalar is a random ephemeral, different for each signature: on the contrary, the settings of DPA require a fixed scalar and variable base points. On the other hand, the scalar multiplications in EC-ElGamal decryption can be attacked by DPA. Therefore, depending on the situation, DPA resistance is not always required. When necessary, the standard DPA countermeasures such as randomized projective coordinates, randomized scalar, randomized base point or randomized field parameter have to be deployed [3], [8], [14].

Another class of attacks closely related to side-channel attacks consists in injecting faults during computations, for instance by feeding the power supply with higher voltage. Then, by analyzing the output of the cryptosystem, the attacker can sometimes recover critical information about the secret parameters [1]. A simple countermeasure is to check for faulty results and discard them. However, if the fault is injected on a dummy operation, the result is not faulty: with these safe-error fault attacks, it becomes possible to detect dummy operations [18]. As a consequence, it is preferable to avoid dummy operations in practical implementations.

3. Proposed SPA-Resistant Techniques

In the following, we describe two methods to protect the scalar multiplication on Koblitz curves against side-channel attacks.

3.1 Motivation

Since standard DPA countermeasures such as randomization of the coordinates of the base point are straightforwardly applicable on Koblitz curves, achieving DPA resistance is not a major problem. Moreover, depending on applications, DPA attacks are not always applicable: for instance, the signature scheme EC-DSA is immune to such attacks. On the other hand, SPA is always applicable and the countermeasures for general curves do not take advantage of the complex multiplication, which is the key of the efficiency on Koblitz curves. Currently, there is no scalar multiplication scheme on Koblitz curves which is (1) efficient (2) resistant to SPA attacks (3) flexible (4) free from dummy operations. By efficient, we refer to an acceptable overhead in terms of memory and computational cost compared to unprotected scalar multiplication schemes. Thus, the countermeasure has to make use of the fast arithmetic of Koblitz curves (i.e. replacing point doublings by the Frobenius). Our SPA attack model assumes that the attacker is able to distinguish individual τ multiplications. Hence, we consider the standard case of a software implementation using a polynomial basis. By flexible, we mean here that a window method can be deployed, allowing trade-offs between memory consumption and computational cost. Finally, we avoid dummy operations since they are vulnerable to safe-errors attacks [18].

On general elliptic curves, the SPA-resistant NAF_w is a fast SPA countermeasure which recodes the scalar using a

fixed pattern [15]. Depending on a system parameter, the width w of the algorithm, some points are pre-computed in order to speed-up the computations: w is the trade-off parameter between memory and efficiency. Thus, the countermeasure seems to satisfy our criteria. Unfortunately, the recoding of the scalar is based on properties of binary expansions; on Koblitz curves, such binary expansions are replaced by τ expansions and the countermeasure is not applicable anymore. In the following, we show how to adapt the SPA-resistant NAF_w to the case of Koblitz curves.

3.2 SPA-resistant TNAF_w

Recall that to generate the TNAF_w, one computes representatives of congruent classes modulo τ^w with the map $\Phi_w : d_0+d_1\cdot\tau \in \mathbb{Z}[\tau] \mapsto d_0+d_1\cdot t_w \bmod 2^w$. Then, $d-\Phi_w(d)$ is divisible by τ^w. However, we aim at generating a fixed pattern: we look for a new set of representatives modulo τ^w, which verify $d-u$ is divisible by τ^w, but additionally $d-u$ is *not* divisible by τ^{w+1}.

Proposition 1. *Let the map Ψ_w be defined as:*

$$\begin{array}{rccc} \Psi_w : & \mathbb{Z}[\tau] & \to & \mathbb{Z}/2^w\mathbb{Z} \\ & d_0+d_1\tau & \mapsto & (d_0+d_1 t_{w+1} \bmod 2^{w+1}) - 2^w \end{array} \quad (4)$$

For any $d \in \mathbb{Z}[\tau]$, $d-\Psi_w(d)$ is divisible by τ^w but not by τ^{w+1}.

Proof. Recall that d is divisible by τ^w iff $\Phi_w(d) = 0$ [17]. We first prove that $d-\Psi_w(d)$ is divisible by τ^w, in other words, that $\Phi_w(d-\Psi_w(d)) = 0$. Let $d = d_0 + d_1\cdot\tau \in \mathbb{Z}[\tau]$. Then, $\Phi_w(d-\Psi_w(d)) = d_1(t_w - t_{w+1}) \bmod 2^w$. Moreover, $t_w - t_{w+1} = (U_w^2 - \mu U_{w-1}U_w + 2U_{w-1}^2)/(U_w U_{w+1})^{-1} \bmod 2^w$. We can write that $U_w^2 - \mu U_{w-1}U_w + 2U_{w-1}^2 = |U_w - U_{w-1}\cdot\tau|$ and $U_w - U_{w-1}\cdot\tau = \tau^{w-1}$. Because $|\tau^{w-1}| = 2^{w-1}$, $t_w - t_{w+1} = -2^w U_w^{-1} U_{w+1}^{-1} = 0 \bmod 2^w$. It follows that $\Phi_w(d-\Psi_w(d)) = 0$. Besides, it is trivial that: $\Phi_{w+1}(d-\Psi_w(d)) = -2^w \neq 0 \bmod 2^{w+1}$. Then, $\Phi_{w+1}(d-\Psi_w(d)) \neq 0$ and $d-\Psi_w(d)$ is not divisible by τ^{w+1}. □

As a consequence, Ψ_w can be used to generate SPA-resistant TNAF_w expansions. Note that the input of the recoding algorithm is $\rho = r_0 + r_1\cdot\tau \in \mathbb{Z}[\tau]$, corresponding to an integer d which was first reduced modulo δ in order to generate a shorter recoding. More precisely, the SPA-resistant TNAF_w recoding has $\lceil (m+a)/w \rceil$ non-zero digits; with the original TNAF_w, thanks to the reduction modulo δ, the recoded scalar has $m+a$ digits.

Based on the recoding computed by Algorithm 2, Algorithm 3 protects the scalar multiplication on Koblitz curves against SPA.

Proposition 2 (τ-SPA resistance). *The ability to distinguish individual τ multiplications and point additions in power traces confers no advantage for finding the scalar in Algorithm 3.*

Algorithm 2: Conversion to SPA-resistant TNAF_w

INPUT: $\rho = r_0 + r_1\cdot\tau \in \mathbb{Z}[\tau]$ with r_0 odd, width w;
OUTPUT: $(d_l^{(w)}, \ldots d_0^{(w)}) = \mathrm{TNAF}_w(\rho)$;

(1) $c_0 \leftarrow r_0$; $c_1 \leftarrow r_1$, $l \leftarrow 0$;
(2) **while** $c_1 \neq 0$ **or** $|c_0| > 2^w$ **do**
  (a) $u \leftarrow \Psi_w(c_0 + c_1\cdot\tau)$; $d_l^{(w)} \leftarrow u$; $c_0 \leftarrow c_0 - u$; $l \leftarrow l+1$;
  (b) **for** j **from** 1 **to** w **do** $(c_0, c_1) \leftarrow (c_1 + \mu c_0/2, -c_0/2)$;
(3) $d_l^{(w)} \leftarrow \Psi_w(c_0 + c_1\cdot\tau)$; $l \leftarrow l+1$;
(4) **return** $(d_{l-1}^{(w)}, \ldots d_0^{(0)})$;

Algorithm 3: Scalar multiplication using SPA resistant TNAF_w

INPUT: a scalar d, base point P, width w;
OUTPUT: multiplied point $Q = dP$;

(1) pre-compute $3P, 5P, \ldots, (2^w-1)P$; $\rho \leftarrow d \bmod \delta$;
(2) **if** ρ is divisible by τ **then** $\rho' \leftarrow \rho + 1$; **else** $\rho' \leftarrow \rho + \tau$;
(3) compute $(d_{l-1}^{(w)}, \ldots d_0^{(w)})$, from ρ' with Algorithm 2;
(4) $Q \leftarrow \mathcal{O}$;
(5) **for** i **from** l **down to** 0 **do**
  (a) **for** j **from** 1 **to** w **do** $Q \leftarrow \tau Q$;
  (b) **if** $d_i^{(w)} > 0$ **then** $Q \leftarrow Q + d_i^{(w)} P$; **else** $Q \leftarrow Q - (-d_i^{(w)})P$;
(6) **if** ρ is divisible by τ **then** $Q \leftarrow Q - P$; **else** $Q \leftarrow Q - \tau P$;
(7) **return** Q;

Proof. Since the main loop (i.e. Step 5) is τ-SPA resistant because the scalar representation has a fixed pattern, the only concern is that the scalar multiplication scheme requires an "odd" input ρ (i.e. indivisible by τ). If ρ is divisible by τ, one can add 1 to $\rho = d \bmod \delta$, and adjust the result of the scalar multiplication by subtracting P. To prevent attackers from distinguishing the cases where ρ is odd or even, one can always add τ to ρ if it is odd, and subtract τP from the result of the scalar multiplication $Q = (\rho+\tau)P$. □

It follows that our scheme is τ-SPA resistant, that is, assuming strong abilities for the attacker, who is able to distinguish individual τ multiplications. In fact, in our attack model, the information arisen from SPA is of *no* use for attackers. In terms of computational cost, Algorithm 3 computes $\lceil (m+a)/w \rceil$ point additions in the main loop. The pre-computation of $2^{w-1}-1$ points requires $2^{w-1}-1$ point additions and 1 point doubling. Making the cases ρ odd and ρ even indistinguishable by SPA requires one more point addition. Then, the total computational cost of Algorithm 3 is:

$$\begin{array}{rcl} C_{STNAF_w} & = & (m+a)\mathrm{ECFRB} \\ & + & \left(2^{w-1} + \left\lceil \frac{m+a}{w} \right\rceil\right)\mathrm{ECADD} + \mathrm{ECDBL}. \end{array} \quad (5)$$

Note that the computational cost of ECFRB is small but not negligible.

3.3 Zero-Free Representation

The recoding of the SPA-resistant TNAF$_w$ utilizes a right-to-left strategy: the scalar must be first recoded and stored in its new representation, wasting memory. However, we can partially fix the problem, using a two-round recoding. In the first round, the scalar is converted to a zero-free representation, using a right-to-left approach. In the second round, a windowing technique is applied in order to take advantage of pre-computed points and reduce computational costs. We will see that in many practical situations, it is not even necessary to compute the first round in the runtime.

First, we explain how to convert the scalar to a zero-free representation. The set of digits of the SPA-resistant TNAF$_w$ consists of $\pm 1, \pm 3, \ldots, \pm(2^w - 1)$. Especially, when $w = 1$, the representation uses only two digits, namely 1 and -1, and the scalar multiplication is carried out without pre-computations. Unfortunately, since zeros are absent from the new representation, about m point additions are necessary to compute the scalar multiplication, whereas the original TNAF needed only $m/3$ point additions. On the other hand, this zero-free representation has several interesting properties. First, it is SPA resistant. Second, since the only possible digits are 1 and -1, one can easily store the recoded scalar in memory by representing the digit 1 with the bit 0, and -1 with the bit 1, for instance. Additionally, a random representation can be easily generated from a random bit sequence.

1	−1	1	1	1	−1	1	−1	−1	−1	zerofree
1	−1	1	1	1	−1	1	−1	−1	−1	$w = 2$
0	1	0	3	0	1	0	1	0	−3	
1	−1	1	1	1	−1	1	−1	−1	−1	$w = 3$
0	0	3	0	0	5	0	0	1	−1	
1	−1	1	1	1	−1	1	−1	−1	−1	$w = 4$
0	0	0	5	0	0	0	5	−1	−1	

図 1 Windowing technique on zero-free representation

It remains to reduce computational costs. In fact, it is easy to apply a windowing technique to the zero-free representation, while preserving the original SPA-resistance: simply divide the scalar into windows of w consecutive digits, from left to right, and if the right-most window is not full, treat each digit independently in a window $w = 1$ (注1), as shown in Fig. 1. Then, the computational cost of the scalar multiplication (excluding pre-computations) is:

$$\begin{aligned} C_{ZF_w} &= (m + a)\text{ECFRB} \\ &+ \left(\left\lfloor \tfrac{m+a}{w} \right\rfloor + m + a - w \left\lfloor \tfrac{m+a}{w} \right\rfloor\right) \text{ECADD}. \end{aligned} \quad (6)$$

(注1) : The number of windows $w = 1$ does not depend on the scalar: attackers cannot use this information to mount attacks.

In each (full) window, the possible digits are $\pm\tau^{w-1} \pm \ldots \pm \tau \pm 1$, and ± 1 in the rightmost windows. Therefore, it suffices to pre-compute the points $\tau^{w-1}P \pm \ldots \pm \tau P \pm P$ and compute point additions or subtractions depending on the sign of the leftmost digit in each windows. Interestingly, these points can be pre-computed using simultaneous additions and subtractions: the computational cost of the simultaneous computation of $P + Q$ and $P - Q$ is ECAS $= 4M + I$ instead of $4M + 2I$, where M and I denote the cost of field multiplications and inversions, respectively. In a naive approach based on tree exploration, one computes P, then $\tau P \pm P$, after that $\tau^2 P \pm \tau P \pm P$, and so on. This technique requires $2^{w-1} - 1$ simultaneous additions-subtractions, which is slower than the $2^{w-1} - 1$ additions in the case of the SPA-resistant TNAF$_w$. However, in a more efficient approach, one can re-use partial results in order to compute the next steps. Figure 3.3 illustrates this pre-computation technique: the partial results at a given depth in the tree are re-combined using simultaneous additions-subtractions in order to expand the tree.

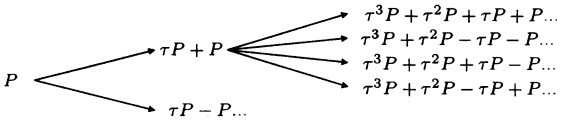

図 2 Efficient pre-computations in the zero-free method

With this pre-computation approach, at the ith level of the tree (starting from P, level 0), there are 2^{2^i-1} points, and 2^{2^i-2} simultaneous point additions-subtractions are required to compute them from level $i - 1$. As a consequence, the computational cost of pre-computations at level i, corresponding to a pre-computed table with width $w = 2^i$ is $1 + \sum_{j=1}^{i} 2^{2^j-2}$ECAS. This is expected to be faster than in the case of the SPA-resistant TNAF$_w$. In the general case, using a hybrid approach which combines levels of the tree at different depths (for example, $w = 5$ can be computed using $w = 2$ and $w = 3$), the pre-computations are indeed faster than those of the SPA-resistant TNAF$_w$.

4. Applications, Improvements and Comparisons

We describe applications of our techniques in the frame of standard ECC protocols, and show how they compare to other schemes.

4.1 Efficient Representation for Fixed Scalar

When the scalar is fixed (for instance the secret key in EC-ElGamal), part of the recoding work can be pre-processed off-line, that is, once for all during the initialization of the smartcard. In particular, instead of storing the integer value

of the secret key, one can reduce it modulo δ and store the corresponding element of the quadratic field $c_0 + c_1 \cdot \tau \in \mathbb{Z}[\tau]$, or even recode it off-line to an SPA-resistant representation (注2). Especially, we can compute the zero-free representation (that is, the SPA-resistant TNAF_1) off-line and choose the window size w in the runtime. Storing the recoded scalar takes only $m + a$ bits.

In Algorithm 4, we consider how to use the windowed zero-free method to compute the scalar multiplication with fixed scalar. Since the zero-free method is based on a fixed-pattern recoding, the scalar multiplication is SPA-resistant. However, to generate the zero-free representation, the SPA-resistant TNAF_1 needs an input $c_0 + c_1\tau$ where c_0 is odd (i.e. the scalar is not divisible by τ). Therefore, after reducing the scalar, we check whether c_0 is divisible by τ or not; if this is the case, we add 1 and set an additional parity bit d_p to 0, and if not, we add τ and set d_p to 1. After computing the scalar multiplication, we adjust the result by subtracting P or τP depending on the parity bit. Since the first (right-to-left) zero-free recoding round has been eliminated, the second (left-to-right) windowing round can be carried out on-the-fly during the scalar multiplication, with no further memory consumption.

Algorithm 4: Computing $Q = dP$ for fixed d

INPUT: Base point P, zero-free recoding $(d_{m+a-1} \ldots d_0)$, parity bit d_p, width w;
OUTPUT: $Q = dP$;

(1) pre-compute $\tau^{w-1}P \pm \tau^{w-2}P \pm \ldots \pm \tau P \pm P$;
(2) $Q \leftarrow \mathcal{O}$; $i \leftarrow m + a - 1$;
(3) **while** $i \geqq w$ **do**
  (a) **for** j **from** 0 **to** $w - 1$ **do** $Q \leftarrow \tau Q$;
  (b) $Q \leftarrow Q + (-1)^{d_i}\tau^{w-1}P + (-1)^{d_{i-1}}\tau^{w-2}P + \ldots + (-1)^{d_{i-w+1}}P$; $i \leftarrow i - w$;
(4) **for** j **from** 0 **to** i **do** $Q \leftarrow \tau Q + (-1)^{d_{i-j}}P$;
(5) **if** $d_p = 0$ **then** $Q \leftarrow Q - P$ **else** $Q \leftarrow Q - \tau P$; **return** Q

4.2 Random SPA-Resistant Representation for Secret Ephemerals

In many cryptographic protocols, a random ephemeral is needed. Since the knowledge of the ephemeral generally allows to recover the secret key, it is important to protect scalar multiplications with the ephemeral against SPA. One can always generate a random *integer* multiplier, reduce it modulo δ, convert it to an SPA-resistant representation and finally perform the scalar multiplication. However, it would be preferable to generate a random SPA-resistant representation instead of a random integer, and even better, to generate the successive coefficients of the representation on-the-fly, from left to right. Additionally, since the integer value of the random multiplier is often needed along with the multiplied point, we should compute this integer value without excessive overhead.

In the following, we present a method for generating a random windowed zero-free representation along with its integer value. Our technique takes w random bits and generates the coefficients of the windowed zero-free representation one after the other, on-the-fly and from left to right. To compute the integer value of the multiplier, we simply reverse Algorithm 2. This method employs only additions, shifts, and one final integer multiplication with τ to compute the integer value of the zero-free representation. Since Algorithm 2 is right-to-left, its reversed counterpart works left-to-right, and can also be executed on the fly.

Algorithm 5: Generating a random point $Q = kP$

INPUT: Base point P;
OUTPUT: Random integer k and the corresponding point $Q = kP$;

(1) pre-compute $\tau^{w-1}P \pm \tau^{w-2}P \pm \ldots \pm \tau P \pm P$;
(2) $Q \leftarrow \mathcal{O}$; $c_0 \leftarrow 0$; $c_1 \leftarrow 0$; $i \leftarrow m + a - 1$;
(3) **while** $i \geqq w$
  (a) pick w random bits $d_i d_{i-1} \ldots d_{i-w+1}$;
  (b) **for** j **from** 0 **to** $w - 1$ **do** $Q \leftarrow \tau Q$; $(c_0, c_1) \leftarrow (-2c_1 + (-1)^{d_{i-j}}, c_0 + \mu c_1)$;
  (c) $Q \leftarrow Q + (-1)^{d_i}\tau^{w-1}P + (-1)^{d_{i-1}}\tau^{w-2}P + \ldots + (-1)^{d_{i-w+1}}P$; $i \leftarrow i - w$;
(4) pick $i + 1$ random bits $d_i d_{i-1} \ldots d_0$ and a parity bit d_p;
(5) **for** j **from** 0 **to** i **do** $Q \leftarrow \tau Q + (-1)^{d_{i-j}}P$; $(c_0, c_1) \leftarrow (-2c_1 + (-1)^{d_{i-j}}, c_0 + \mu c_1)$;
(6) **if** $d_p = 0$ **then** $Q \leftarrow Q - P$; $c_0 \leftarrow c_0 - 1$; **else** $Q \leftarrow Q - \tau P$; $c_1 \leftarrow c_1 - 1$;
(7) $k \leftarrow c_0 + c_1 \cdot \tau \bmod \#\mathcal{E}_a$; **return** k and Q;

Proposition 3. *The distribution of random zero-free chains is close to the uniform distribution. In fact, its statistical distance* $\Delta(g) = \sum_{i=0}^{\#\mathcal{E}_a - 1} |P(g = i) - \frac{1}{\#\mathcal{E}_a}|$ *to the uniform distribution is bounded by:*

$$\Delta(g) \leqq \frac{1}{2}\sqrt{\sum_{i=1}^{\#\mathcal{E}_a - 1} \prod_{j=0}^{m+a-1} |\cos(\frac{2\pi \cdot i}{\#\mathcal{E}_a}(U_j\tau - 2U_{j-1}))|^2}. \quad (7)$$

Proof. To bound the statistical distance, we use the fact that for any random variable X, we have: $\Delta(X) \leqq \frac{1}{2}\sqrt{\sum_{i=1}^{\#\mathcal{E}_a - 1} |E[e^{\frac{2\pi \Im i X}{\#\mathcal{E}_a}}]|^2}$, where $\Im$ denotes the complex $\sqrt{-1}$ and i is used as index. See [9] for a proof. Obviously, the distribution of windowed zero-free τ expansions does not depend on the width w; more precisely, we only have to study the distribution of signed "binary" zero-free

(注2) : In this case, DPA countermeasures based on scalar blinding are not available, since the recoding is fixed. However, one can still deploy other types of countermeasures, such as randomized projective coordinates or randomized base point [3].

表 1 Approximated values of α, where $\Delta(g) \leq 2^{-m/\alpha}$ for several bitlengths m

m	109	113	131	163	233	239	277	283	359	409	571
α	4.0	4.3	4.2	3.9	3.6	3.5	3.5	3.6	3.3	3.2	3.0

representations, that is, the distribution of the SPA-resistant TNAF_1. Since all bits of the SPA-resistant TNAF_1 expansion $\sum_{j=0}^{m+a-1}(-1)^{d_j}\tau^j$ are chosen independently, we have $|E[e^{\frac{2\pi\Im i}{\#\mathcal{E}_a}(\sum_{j=0}^{m+a-1}(-1)^{d_j}\tau^j)}]| = \prod_{j=0}^{m+a-1}|E[e^{\frac{2\pi\Im\cdot i}{\#\mathcal{E}_a}(-1)^{d_j}\cdot\tau^j}]|$. Additionally, $|E[e^{\frac{2\pi\Im i}{\#\mathcal{E}_a}(-1)^{d_j}\cdot\tau^j}]| = |\frac{e^{\frac{2\pi\Im i}{\#\mathcal{E}_a}\tau^j}+e^{-\frac{2\pi\Im i}{\#\mathcal{E}_a}\tau^j}}{2}| = |\cos(\frac{2\pi\cdot i\cdot\tau^j}{\#\mathcal{E}_a})|$, and $\tau^j = U_j\tau - 2U_{j-1}$, which proves the result. □

Conjecture 1. *The statistical distance of g to the uniform distribution is bounded by $\Delta(g) \leq 2^{-m/5}$.*

We computed approximations of the sum by using a smaller pool of random values of $i \in \{1, 2, \ldots, \#\mathcal{E}_a - 1\}$. According to our numerical experimentations for several bitlengths m and 8192 random values of i, the experimental statistical distance is indeed smaller than $2^{-m/5}$: writing $\Delta(g) \approx 2^{-m/\alpha}$, the experimental value of α seems to decrease as m grows, which tends to show that our conjecture is reasonable.

4.3 Comparison with Known Methods

a) Fast scalar multiplication on Koblitz curves.

The fastest scalar multiplication on Koblitz curves is the TNAF_w [7]; although our proposed techniques do not intend to compete with the (insecure) TNAF_w, Table 2 puts in evidence the overhead introduced to achieve SPA-resistance. Roughly speaking, for the same memory consumption, the TNAF_w utilizes a window size $w+2$ where our techniques have w. This is the price to pay to achieve SPA resistance.

表 2 Compared computational costs ($m = 163$)

	insec. TNAF_w [7], [17]	SPA-res. TNAF_w Alg. 2, 3	zero-free method Alg. 4, 5
$w=1$	-	$1312M$, 0 bytes	$1312M$, 0 bytes
$w=2$	$437M$, 0 bytes	$680M$, 42 bytes	$670M$, 84 bytes
$w=3$	$338M$, 42 bytes	$488M$, 126 bytes	$490M$, 168 bytes
$w=4$	$296M$, 126 bytes	$424M$ 294 bytes	$398M$ 294 bytes
$w=5$	$301M$, 294 bytes	$456M$, 630 bytes	$442M$, 672 bytes

b) SPA-resistant methods.

SPA countermeasures on binary curves, such as the Montgomery ladder [13], are also applicable to Koblitz curves. However, the computational advantage introduced by τ multiplications is lost in that case. But still, the Montgomery ladder requires only 978 multiplications for $m = 163$, protects against SPA with no pre-computed points. On the one hand, when memory resources are extremely scarce, the Montgomery ladder performs better than our methods. On the other hand, when some memory is available for pre-computed tables, as soon as $w = 2$, our methods beat the Montgomery ladder.

Some SPA countermeasures using τ expansions were proposed in [6]. However, these countermeasures are not optimal in terms of memory and computational cost: our methods are more efficient. In [16], the SPA-resistance properties of the TNAF_w using a change-of-basis strategy is pointed out. In a normal basis, the operation $\tau^w P$ is a simple cyclic shift of the coordinates of P. Thus, they claim the time for computing $\tau^w P$ is independent from w, and therefore, the *position* of nonzero digits is concealed in the TNAF_w. However, it is controversial whether the cyclic shift has a static implementation in software, without using dummy operations. Second, the method leaks the number of nonzero digits of the TNAF_w representation. Even though this problem can be partially fixed by introducing additional operations, some information is still leaked. Third, they did not discuss how to efficiently store or randomly generate the TNAF_w. Finally, even though the method seems to have the same computational cost as the original TNAF_w, there are several drawbacks which may practically slow down the scheme. If dummy operations are used in order to have a static implementation of cyclic shifts and change-of-bases, the latter operations will run much slower. Additionally, since point additions are inserted to conceal the number of nonzero digits, it is not clear what the average cost of the countermeasure is.

c) Secret ephemerals and compact encoding.

Two methods for generating ephemerals on Koblitz curves have been proposed. In [5], the Frobenius is utilized to increase the entropy of standard generators. While this idea leads to a very efficient scalar multiplication, our scheme has several advantages compared to [5]. First, our scheme is SPA-resistant, whereas side channel attacks are not discussed in [5]. Second, we can use the same (off-line) precomputation table for the known point scalar multiplications in the signature generation and verification of EC-DSA, which is impossible in the case of the generator proposed in [5]. In [9], a compact encoding of the NAF_2 is proposed; since their encoding can be randomly generated, they also discuss how to obtain random TNAF_2. Unfortunately, their idea seems only applicable for a width $w = 2$. On Koblitz curves, the computation cost of the scalar multiplication can be drastically reduced with window methods, therefore, this is an important drawback of their method. Besides, the

straight-forward implementation of the $TNAF_2$ is vulnerable to SPA.

5. Conclusion

We presented two new scalar multiplication methods on Koblitz curves: the SPA-resistant $TNAF_w$ and the windowed zero-free method. The first technique extends the mechanisms to the SPA-resistant NAF_w to the arithmetic of Koblitz curves, whereas the second technique is specifically designed for left-to-right computations in some practical situations. Both of our schemes are efficient, SPA-resistant, allow to flexibly choose how much memory is used in order to speed-up the computations, and free from dummy operations. Therefore, we claim that our schemes achieve a high security level for an acceptable overhead compared to insecure methods. Additionally, we proposed practical applications, such as fixed-scalar and random ephemeral multiplication schemes. In this cases, the windowed zero-free method can be optimized by introducing a full left-to-right and on-the-fly recoding.

Therefore, in all practical cases, we show that our countermeasures can protect the scalar multiplication against SPA on Koblitz curves, with low memory requirements and computational cost.

文　献

[1] Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. Proc *Eurocrypt'97*, LNCS **1233** (1997) 37–51.

[2] Chevallier-Mames, B., Ciet, M., Joye, M.: Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity. *IEEE Trans. Comput.*, **53**(6) (2004) 760–768.

[3] Coron, J.-S.: Resistance against differential power analysis for elliptic curve cryptosystems. Proc. *CHES'99*, LNCS **1717** (1999) 292–302.

[4] Cohen, H., Miyaji, A., Ono, T.: Efficient elliptic curve exponentiation using mixed coordinates. Proc. *Asiacrypt'98*, LNCS **1514**, (1998) 51–65.

[5] Coron, J.-S., M'Raïhi, D., Tymen, C.: Fast generation of pairs $(k, [k]P)$ for Koblitz elliptic curves. Proc. *SAC'01*, LNCS **2259** (2001) 151–164.

[6] Hasan, A.: Power analysis attacks and algorithmic approaches to their countermeasures for Koblitz curve cryptosystems. *IEEE Trans. Comput.*, **50**(10) (2001) 1071–1083.

[7] Hankerson, D., López, J., Menezes, A.: Software implementation of elliptic curve cryptography over binary fields. Proc. *CHES'00*, LNCS **1965** (2001) 1–24.

[8] Joye, M., Tymen, C.: Protections against differential analysis for elliptic curve cryptography. Proc. *CHES'01*, LNCS **2162** (2001) 377–390.

[9] Joye, M., Tymen, C.: Compact encoding of non-adjacent forms with applications to elliptic curve cryptography. Proc. *PKC'01*, LNCS **1992** (2001) 353–364.

[10] Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. Proc. *Crypto'99*, LNCS **1666** (1999) 388–397.

[11] Koblitz, N.: CM-curves with good cryptographic properties. Proc. *Crypto'91*, LNCS **576** (1992) 279–287.

[12] Koc96 Kocher, P.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. Proc. *Crypto'96*, LNCS **1109** (1996) 104–113

[13] López, J., Dahab. R.: Fast multiplication on elliptic curves over $GF(2^m)$ without precomputations. Proc. *CHES'99*, LNCS **1717** (1999) 316–327.

[14] Mamiya, H., Miyaji, A., Morimoto, H.: Efficient countermeasure against RPA, DPA, and SPA. Proc. *CHES'04*, LNCS **3156**, (2004) 343-356.

[15] Okeya, K., Takagi, T.: The width-w NAF method provides small memory and fast elliptic scalar multiplications secure against side channel attacks. Proc. *CT-RSA'03*, LNCS **2612** (2003) 328–342.

[16] Park, D. J., Sim, S. G., Lee, P. J.: Fast scalar multiplication method using change-of-basis matrix to prevent power analysis attacks on Koblitz curves. Proc. *WISA 2003*, LNCS **2908** (2003) 474–488.

[17] Solinas, J.: Efficient arithmetic on Koblitz curves. *Designs, Codes, and Cryptography*, **19**(2-3) (2000) 195–249.

[18] Yen, S.-M., Joye, M.: Checking before output may not be enough against fault-based cryptanalysis. *IEEE Trans. Comput.*, **49**(9), (2000) 967–970.