(Virtually) Free Randomization Techniques for Elliptic Curve Cryptography

Mathieu Ciet[1] and Marc Joye[2]

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

Abstract. Randomization techniques play an important role in the protection of cryptosystems against implementation attacks. This paper studies the case of elliptic curve cryptography and propose three novel randomization methods, for the elliptic curve point multiplication, which do *not* impact the overall performance.

Our first method, dedicated to elliptic curves over prime fields, combines the advantages of two previously known solutions: randomized projective coordinates and randomized isomorphisms. It is a generic point randomization and can be related to a certain multiplier randomization technique. Our second method introduces new elliptic curve models that are valid for all (non-supersingular) elliptic curves over binary fields. This allows to use randomized elliptic curve isomorphisms, which in turn allows to randomly compute on elliptic curves with affine coordinates. Our third method adapts a double ladder attributed to Shamir. We insist that all our randomization methods share the common feature to be *free*: the cost of our randomized implementations is virtually the same as the cost of the corresponding non-randomized implementations.

Keywords: Randomization, elliptic curve cryptography, implementation attacks, side-channel analysis, elliptic curve models, point multiplication algorithms.

1 Introduction

The celebrated RSA cryptosystem is the most largely deployed cryptosystem but things are becoming to change. More and more applications propose to use the elliptic curve digital signature algorithm (ECDSA) to sign digital documents or messages.

Elliptic curve cryptography bases its security on the hardness of computing discrete logarithms. More precisely, the elliptic curve discrete logarithm problem (ECDLP) consists in recovering the value of multiplier k, given points $\boldsymbol{P}$ and $\boldsymbol{Q} = [k]\boldsymbol{P}$ on an elliptic curve. There are two main families of elliptic curves used

S. Qing, D. Gollmann, and J. Zhou (Eds.): ICICS 2003, LNCS 2836, pp. 348–359, 2003.

in cryptography [1]: elliptic curves over large prime fields and non-supersingular elliptic curves defined over binary fields.

Although an elliptic curve cryptosystem may be mathematically sound and meets standard security requirements, it may totally succumb to implementation attacks. A powerful implementation attack, due to Kocher *et al.* [16,17], monitors certain side-channel information (e.g., running time or power consumption) during the course of a crypto-algorithm and thereby tries to deduce some sensitive data. For example, the double-and-add algorithm (Fig. 1-a) —i.e., the additive analogue of the so-called square-and-multiply algorithm— used for computing $\boldsymbol{Q} = [k]\boldsymbol{P}$ does not behave regularly. This is even more true for elliptic curves as the classical formulæ for point doubling and point addition are different. To thwart simple power analysis (SPA) [17] (i.e., side-channel leakage from a single power trace), this algorithm is usually replaced with the 'double-and-add *always*' algorithm [6] (Fig. 1-b). Throughout this paper, we will use this latter algorithm to benchmark our randomization methods (as well as NAF based variants; cf. Appendix A).

Input: $\boldsymbol{P}, k = (1, k_{\ell-2}, \ldots, k_0)_2$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow \boldsymbol{P}$
for $i = \ell - 2$ **down to** 0 **do**
  $\boldsymbol{R_0} \leftarrow [2]\boldsymbol{R_0}$
  if $(k_i = 1)$ **then** $\boldsymbol{R_0} \leftarrow \boldsymbol{R_0} + \boldsymbol{P}$
endfor
return $\boldsymbol{R_0}$

(a) Double-and-add algorithm

Input: $\boldsymbol{P}, k = (1, k_{\ell-2}, \ldots, k_0)_2$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P}$

$\boldsymbol{R_0} \leftarrow \boldsymbol{P}$
for $i = \ell - 2$ **down to** 0 **do**
  $\boldsymbol{R_0} \leftarrow [2]\boldsymbol{R_0}$
  $b \leftarrow \neg k_i$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_b} + \boldsymbol{P}$
endfor
return $\boldsymbol{R_0}$

(b) 'Double-and-add *always*' algorithm

Fig. 1. Binary point multiplication algorithms

Resistance against SPA does not imply resistance against the more sophisticated differential power analysis (DPA) [17]. In [6], Coron explains how to mount a DPA-type attack against the 'double-and-add always' algorithm. At step i, this attack requires to form two sets of points: the first set is comprised of points $\boldsymbol{P_j}$ such that $\Gamma\left(\sum_{t=i}^{\ell-1}[k_t 2^{t-i}]\boldsymbol{P_j}\right) = 0$ and the second set of points $\boldsymbol{P_j}$ such that $\Gamma\left(\sum_{t=i}^{\ell-1}[k_t 2^{t-i}]\boldsymbol{P_j}\right) = 1$ where $\Gamma(\boldsymbol{P})$ denotes a Boolean selection function (e.g., the value of any specific bit in the binary representation of $\boldsymbol{P}$). To avoid this attack, one has to prevent the attacker to form the two sets. This can be achieved by randomizing point $\boldsymbol{P}$ or multiplier k; or better, as recently exemplified by Goubin [10], by randomizing both $\boldsymbol{P}$ and k.

In the last four years, several randomization methods have been proposed (e.g., [6,14]). This paper proposes further randomization methods that all have in common to be (virtually) free, leading to performances surpassing those of prior art. It is organized as follows. In the next section, we deal with elliptic curves over large prime fields. We propose a generic and free method for point randomization and compare it with a previous multiplier randomization. In Section 3, we introduce new models for elliptic curves over binary fields. Based on these, we propose free point randomization methods allowing to work in affine coordinates and so answer a problem left open in [14]. In Section 4, we present a *regular* variant of Shamir's double ladder. Our variant allows to construct a free multiplier randomization method. Finally, we conclude in Section 5.

2 Point Randomization over Large Prime Fields

This section deals with point randomization techniques for elliptic curves defined over large prime fields. The case of elliptic curves over binary fields is treated in Section 3.

Let $\mathbb{F}_p$ be a (large) prime field with $p > 3$. An elliptic curve over $\mathbb{F}_p$ is given by the points $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ satisfying the Weierstraß equation

$$E_{/\mathbb{F}_p} : y^2 = x^3 + ax + b \tag{1}$$

along with point $\boldsymbol{O}$ at infinity.

2.1 Previous Work

For preventing DPA-type attacks, Coron [6] suggests to represent base-point $\boldsymbol{P} = (x, y) \in E \setminus \{\boldsymbol{O}\}$ with an equivalent projective representation as $\boldsymbol{P^*} = (r^2x, r^3y, r)$ —where r is randomly chosen in $\mathbb{F}_p^\times$— and to compute $\boldsymbol{Q^*} := [k]\boldsymbol{P^*} = (X_k^*, Y_k^*, Z_k^*)$ in Jacobian coordinates. The result of the point multiplication, $\boldsymbol{Q} = [k]\boldsymbol{P}$, is then obtained as $\boldsymbol{Q} = \big(X_k^*/(Z_k^*)^2, Y_k^*/(Z_k^*)^3\big)$ if $Z_k \neq 0$ and $\boldsymbol{Q} = \boldsymbol{O}$ otherwise. The same technique applies if $\boldsymbol{P^*}$ is represented with homogeneous coordinates instead of Jacobian coordinates. We refer the reader to [6] for detail.

Another efficient means for randomizing base-point $\boldsymbol{P}$, proposed by Joye and Tymen [14], consists in working with isomorphic curves. All elliptic curves defined by the Weierstraß equations

$$E^{(u)}_{/\mathbb{F}_p} : y^2 = x^3 + u^4ax + u^6b$$

with $u \in \mathbb{F}_p^\times$ are isomorphic to the initial elliptic curve given by Eq. (1). So the evaluation of $\boldsymbol{Q} = k\boldsymbol{P}$ can be carried out by picking a random $r \in \mathbb{F}_p^\times$, computing $\boldsymbol{Q^*} := k\boldsymbol{P^*} = (x_k^*, y_k^*)$ on $E^* := E^{(r)}$ where $\boldsymbol{P^*} = (r^2x, r^3y)$ and finally obtaining $\boldsymbol{Q} = (r^{-2}x_k^*, r^{-3}y_k^*)$. This technique naturally extends to projective coordinates [14].

If we compare the two methods, depending on the implementation, both have advantages. For efficiency reasons, point multiplications on elliptic curves over large prime fields are done using Jacobian coordinates [5] and curve parameter a is suggested to be selected as $a = -3$ [1].

The first method —randomized projective representations— allows to keep the value of $a = -3$. The second method —randomized isomorphic elliptic curves— allows in commonly used point multiplication algorithms to simplify the addition formulæ by taking the Z-coordinate of base-point $\boldsymbol{P}$ equal to 1. Assuming that $\boldsymbol{Q} = [k]\boldsymbol{P}$ is computed with the 'double-and-add always' algorithm, the performances of the two methods are summarized in Table 1. The cost of pre- and post-computations are neglected. The bit-length of k is denoted by $|k|_2$.

Table 1. Number of multiplications (in $\mathbb{F}_p$) for computing $\boldsymbol{Q} = [k]\boldsymbol{P}$ in Jacobian coordinates on an elliptic curve with parameter $a = -3$

Method	'double-and-add always' (Fig. 1-b)	NAF-based variants[1]	
		simple	HM ([12])
No randomization	$19 \cdot \|k\|_2$	$17\frac{1}{2} \cdot \|k\|_2$	$15 \cdot \|k\|_2$
Randomized representations ([6])	$24 \cdot \|k\|_2$	$20 \cdot \|k\|_2$	$17\frac{7}{9} \cdot \|k\|_2$
Randomized EC isomorphisms ([14])	$21 \cdot \|k\|_2$	$20\frac{1}{2} \cdot \|k\|_2$	$17\frac{2}{9} \cdot \|k\|_2$

2.2 New Method: $2\boldsymbol{P}^*$

We now present a new randomization method, applicable to most left-to-right point multiplication algorithms, that combines the advantages of the two aforementioned methods: the value of parameter a *and* the Z-coordinate of base-point $\boldsymbol{P}$ are unchanged.

Previously known solutions randomize the input base-point $\boldsymbol{P}$ as $\boldsymbol{P}^* := \Upsilon(\boldsymbol{P})$ and compute $[k]\boldsymbol{P}^*$ where from the value of $\boldsymbol{Q} := [k]\boldsymbol{P}$ is derived. Our idea is fairly simple yet very efficient. Instead of randomizing $\boldsymbol{P}$, we randomize $[2]\boldsymbol{P}$ by choosing the method of randomized projective coordinates for function Υ. This allows to keep the Z-coordinate of $\boldsymbol{P}$ equal to 1 throughout the point multiplication algorithm.

Figure 2 depicts a slight modification of the basic 'double-and-add always' algorithm (Fig. 1-b) including our randomization method. The NAF based variants (Appendix A) can be adapted similarly.

If Υ denotes the randomized projective representation method ([6]) then we need $19 \cdot |k|_2$ field multiplications for evaluating $\boldsymbol{Q} = [k]\boldsymbol{P}$ with our modified algorithm of Fig. 2 and $17\frac{1}{2} \cdot |k|_2$ (resp. $15 \cdot |k|_2$) with the corresponding adaptation of the NAF based variants, on an elliptic curve with parameter $a = -3$. In other words, as shown in Table 1, these algorithms have the *same* complexity as their deterministic (i.e., non-randomized) counterpart. Compared to the state-of-the-art, this translates into a speedup factor of $\approx 10\%$ for the 'double-and-add always' algorithm and of $\approx 13\%$ for the NAF based variants.

[1] The NAF based variants are described in Appendix A.

Input: $\boldsymbol{P}, k = (1, k_{\ell-2}, \ldots, k_0)_2$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P}$

$\boldsymbol{P}^* \leftarrow \Upsilon(\boldsymbol{P})$ [base-point randomization]
  $\boldsymbol{R_0} \leftarrow [2]\boldsymbol{P}^*$
  for $i = \ell - 2$ **down to** 1 **do**
    $b \leftarrow \neg k_i$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_b} + \boldsymbol{P}$
    $\boldsymbol{R_0} \leftarrow [2]\boldsymbol{R_0}$
  endfor
  $b \leftarrow \neg k_0$; $\boldsymbol{R_b} \leftarrow \boldsymbol{R_b} + \boldsymbol{P}$
return $\Upsilon^{-1}(\boldsymbol{R_0})$

Fig. 2. Randomized algorithm $2\boldsymbol{P}^*$

It is also worth noting that our randomization technique is *generic* in the sense that it applies to numerous point multiplication algorithms.

2.3 Interpretation

In our case, the randomization of base-point $\boldsymbol{P}$ can nicely be related to randomization techniques of multiplier k in the computation of $\boldsymbol{Q} = [k]\boldsymbol{P}$. This pushes a step further previous observations made by Okeya and Sakurai in [21].

Let E denote an elliptic curve over $\mathbb{F}_p$ with $\#E$ points. Instead of computing $\boldsymbol{Q} := [k]\boldsymbol{P}$ directly, Coron suggests in [6] to pick a short random number r (typically r is 32-bit integer) and then compute $\boldsymbol{Q}$ in a random way as

$$k^* := k + r \cdot \#E \quad \text{and} \quad \boldsymbol{Q} = [k^*]\boldsymbol{P} \ .$$

In order to optimize modular arithmetic, elliptic curves recommended in the cryptographic standards are defined over a prime field $\mathbb{F}_p$ where p is a generalized Mersenne prime, that is, a prime of the form $p = 2^\ell \pm 2^m \pm 1$ where m is relatively small. As a result, since from Hasse theorem we have $|\#E - p - 1| \le 2\sqrt{p}$, it follows that the binary representation of $\#E$ is likely to be a '1' followed by a long run of '0's. For example, in hexadecimal, the elliptic curve "secp160k1" from [2, Section 2.4] has

$$\#E = \texttt{01 00000000 00000000 0001B8FA 16DFAB9A CA16B6B3}_{16}$$

points. The randomized multiplier, k^*, then typically looks as

$$k^* := k + r \cdot \#E = (r)_2 \| k_{\ell-1} \cdots k_{\ell-t} \| \underbrace{\text{some bits}}_{:=\alpha} \ .$$

Observe that the t most significant bits of multiplier k appear in clear. If $[k^*]\boldsymbol{P}$ is evaluated with the 'double-and-add always' algorithm then, letting $k^* = r\,2^\ell + \lfloor k/2^{\ell-t} \rfloor 2^t + \alpha$, we first compute $\boldsymbol{P_1} := [r]\boldsymbol{P}$, and continue with $\lfloor k/2^{\ell-t} \rfloor 2^t + \alpha$ as the multiplier.

Remarking that with the 'double-and-add always' algorithm, (true/dummy) point additions are always performed with point $\boldsymbol{P}$ (not $\boldsymbol{P_1}$), our randomized algorithm $2\boldsymbol{P}^*$ (Fig. 2) can be seen, in the previous example, as a variation of the randomized multiplier method where $[2]\boldsymbol{P}^*$ plays the role of $\boldsymbol{P_1}$, for the leading bits of k.

3 Point Randomization over Binary Fields

3.1 Previous Work

The Weierstraß equation for non-supersingular elliptic curves over $\mathbb{F}_{2^m}$ is given by

$$E_{/\mathbb{F}_{2^m}} : y^2 + xy = x^3 + ax^2 + b \quad (\cup\{\boldsymbol{O}\}) \ . \tag{2}$$

The use of randomized projective representations ([6]) for preventing DPA-type attacks is not restricted to elliptic curves over prime fields and equally apply to elliptic curves over binary fields.

On the contrary, the method of randomized isomorphisms does not apply for elliptic curves over binary fields because the x-coordinate of a point is invariant through isomorphism, as noticed in [14]. This is most unfortunate because, over $\mathbb{F}_{2^m}$, affine coordinates lead to better performances [7].[2] The next section explains how to overcome this limitation without performance penalty.

3.2 New Representation

Rather than considering the short Weierstraß equation (Eq. (2)), we consider elliptic curves given by the extended model

$$\widehat{E}_{/\mathbb{F}_{2^m}} : y^2 + xy + \varrho y = x^3 + Ax^2 + Bx + C \quad (\cup\{\boldsymbol{O}\}) \tag{3}$$

with $\varrho, A, B, C \in \mathbb{F}_{2^m}$. As shown in the next proposition, this model is as general as the classical Weierstraß model.

Proposition 1. *The elliptic curves E and $\widehat{E}$ (given by Eq.* (2) *and Eq.* (3)*, respectively) are isomorphic over $\mathbb{F}_{2^m}$ if and only if there exists $\sigma \in \mathbb{F}_{2^m}$ such that*

$$\begin{cases} A = a + \varrho \\ B = \varrho^2 + \sigma \\ C = b + \varrho^2 a + \varrho^3 + \sigma^2 \end{cases} .$$

Furthermore, the isomorphism

$$\varphi : E \xrightarrow{\sim} \widehat{E}, \begin{cases} \boldsymbol{O} \longmapsto \boldsymbol{O} \\ (x, y) \longmapsto (x + \varrho, y + \sigma) \end{cases} . \tag{4}$$

[2] In [11], the authors suggest to use projective rather than affine coordinates. This comes from the ratio of inversion to multiplication. In [11] this ratio is roughly 10 to 1 whereas in [7] it is roughly 3 to 1. For hardware architectures affine coordinates are more suitable.

Proof. This is an application of [18, Theorem 2.2]. □

Let $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2) \in \widehat{E} \setminus \{\boldsymbol{O}\}$. The inverse of $\boldsymbol{P_1}$ is $-\boldsymbol{P_1} = (x_1, x_1 + y_1 + \varrho)$. If $\boldsymbol{P_1} \neq -\boldsymbol{P_2}$ then $\boldsymbol{P_1} + \boldsymbol{P_2} = (x_3, y_3)$ where

$$x_3 = \lambda^2 + \lambda + A + x_1 + x_2 \quad \text{and} \quad y_3 = (x_1 + x_3)\lambda + x_3 + y_1 + \varrho$$

with $\lambda = \begin{cases} \frac{y_1+y_2}{x_1+x_2} & \text{if } x_1 \neq x_2\,, \\ x_1 + \varrho + \frac{y_1+\varrho^2+B}{x_1+\varrho} & \text{otherwise}\,. \end{cases}$

Neglecting (field) additions (i.e., XORs), the addition formulæ on our extended model only requires an additional squaring for the computation of ϱ^2, compared to the formulæ in classical Weierstraß model [1, § A.10]. If the value of ϱ^2 is precomputed or if normal bases [9] are used, its cost can be neglected too.

Consequently, the computation of $\boldsymbol{Q} = [k](x, y)$ can be carried as follows:

1. Randomly choose $\varrho, \sigma \in \mathbb{F}_{2^m}$;
2. Form $\boldsymbol{P^*} = (x + \varrho, y + \sigma)$;
3. Compute $\boldsymbol{Q^*} := [k]\boldsymbol{P^*}$ on $\widehat{E}$;
4. If $\boldsymbol{Q^*} = \boldsymbol{O}$ output $\boldsymbol{O}$
   else $\boldsymbol{Q} = (x_k^*, y_k^*)$ and output $\boldsymbol{Q} = (x_k^* + \varrho, y_k^* + \sigma)$.

A better way for eliminating the additional cost due to the computation of ϱ^2, valid in *all* cases, is to replace the extended model of Eq. (3) by the corresponding quartic form. This is achieved by replacing (x, y) with $(x, y + x^2)$. Doing so, we obtain an elliptic curve, isomorphic to Eq. (3), given by the equation

$$\widehat{E}^Q_{/\mathbb{F}_{2^m}} : y^2 + xy + \varrho y = x^4 + (A + \varrho)x^2 + Bx + C \ . \tag{5}$$

The sum of two points $\boldsymbol{P_1} = (x_1, y_1)$ and $\boldsymbol{P_2} = (x_2, y_2) \in \widehat{E}^Q \setminus \{\boldsymbol{O}\}$ is given by

$$x_3 = \lambda^2 + \lambda + A + x_1 + x_2 \quad \text{and} \quad y_3 = (x_1 + x_3)(\lambda + x_1 + x_3) + x_3 + y_1 + \varrho$$

with $\lambda = \begin{cases} x_1 + x_2 + \frac{y_1+y_2}{x_1+x_2} & \text{if } x_1 \neq x_2\,, \\ \frac{y_1+B}{x_1+\varrho} & \text{otherwise}\,. \end{cases}$.

These formulæ only involve 1 squaring, 2 multiplies and 1 inversion to add or double points, as for the classical Weierstraß model. Neglecting the cost of (field) additions, the computation of $\boldsymbol{Q} = [k](x, y)$ can thus be evaluated *in a random way and without penalty* as:

1. Randomly choose $\varrho, \sigma \in \mathbb{F}_{2^m}$;
2. Form $\boldsymbol{P^*} = (x + \varrho, y + \sigma + x^2 + \varrho^2)$;
3. Compute $\boldsymbol{Q^*} := [k]\boldsymbol{P^*}$ on $\widehat{E}^Q$;
4. If $\boldsymbol{Q^*} = \boldsymbol{O}$ output $\boldsymbol{O}$
   else $\boldsymbol{Q} = (x_k^*, y_k^*)$ and output $\boldsymbol{Q} = (x_k^* + \varrho, y_k^* + \sigma + (x_k^*)^2 + \varrho^2)$.

4 Multiplier Randomization

A very natural way [4] to randomize multiplier k consists in choosing a random integer r of the size of k and to compute $\boldsymbol{Q} := [k]\boldsymbol{P}$ as $\boldsymbol{Q} = [k-r]\boldsymbol{P} + [r]\boldsymbol{P}$. Another possibility is to write k as $k = \lfloor k/r \rfloor r + (k \bmod r)$ for a random r. Letting $\boldsymbol{S} := [r]\boldsymbol{P}$, we can obtain $\boldsymbol{Q} = [k]\boldsymbol{P}$ as

$$\boldsymbol{Q} = [k_1]\boldsymbol{P} + [k_2]\boldsymbol{S} \tag{6}$$

where $k_1 := k \bmod r$ and $k_2 := \lfloor k/r \rfloor$.

The randomized splitting of k is generally disregarded as it appears to double the running time: two point multiplications have to be computed instead of one.

However, as noted by Shamir (see [8]), if one has to evaluate $y := g^k h^d$ in a group G, the intermediate values g^k and h^d are not needed [25].

The next figure describes a *regular* variant of Shamir's double ladder, using additive notations and where G is the group of points of an elliptic curve. We let ℓ denote the bit-length of $\max(k, d)$ —and thus $k_{\ell-1}$ and/or $d_{\ell-1}$ are equal to 1.

Input: $\boldsymbol{P}, k = (k_{\ell-1}, k_{\ell-2}, \ldots, k_0)_2$, $\boldsymbol{S}, d = (d_{\ell-1}, d_{\ell-2}, \ldots, d_0)_2$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P} + [d]\boldsymbol{S}$

```
R1 ← P; R2 ← S; R3 ← P + S; c ← 2d_{ℓ−1} + k_{ℓ−1}; R0 ← Rc
for i = ℓ − 2 down to 0 do
    R0 ← [2]R0
    b ← ¬(k_i ∨ d_i); c ← 2d_i + k_i; Rb ← Rb + Rc
endfor
return R0
```

Fig. 3. Regular variant of Shamir's double ladder

Applied to the evaluation of Eq. (6), we see that this variant only requires one point doubling and one point addition per bit, that is, exactly the *same* cost as the 'double-and-add always' algorithm. The NAF based variants (Appendix A) can be adapted along the same lines.

5 Conclusion

This paper dealt with randomization techniques for elliptic curve cryptography; three free novel methods were presented:

– randomized algorithm $2\boldsymbol{P}^*$;
– randomized isomorphisms in affine coordinates;
– randomized algorithm based on Shamir's ladder.

Furthermore, we gave an original interpretation of certain point randomization techniques in terms of multiplier randomizations. We also introduced new models for elliptic curves over binary fields.

Acknowledgements. Part of this work was done while the first author was visiting Gemplus. Thanks go to David Naccache, Philippe Proust and Jean-Jacques Quisquater for making this arrangement possible.

References

1. IEEE Std 1363-2000. *IEEE Standard Specifications for Public-Key Cryptography.* IEEE Computer Society, August 29, 2000.
2. SECG: Standard for Efficient Cryptography Group. *SEC 1: Elliptic Curve Cryptography.* Certicom Research, Version 1.0, September 20, 2000. Available at URL `http://www.secg.org/secg_docs.htm`.
3. Ian Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic Curves in Cryptography*, volume 265 of *London Mathematical Society.* Cambridge University Press, 2000.
4. Christophe Clavier and Marc Joye. Universal exponentiaion algorithm. In Ç.K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 300–308. Springer-Verlag, 2001.
5. Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. Efficient elliptic curve using mixed coordinates. In K. Ohta and D. Pei, editors, *Advances in Cryptology - ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 51–65. Springer-Verlag, 1998.
6. Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems (CHES '99)*, volume 1717 of *Lecture Notes in Computer Science*, pages 292–302. Springer-Verlag-Verlag, 1999.
7. Erik De Win, Serge Mister, Bart Preneel, and Michael Wiener. On the performance of signature schemes based on elliptic curves. In J.-P. Buhler, editor, *Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*, pages 252–266. Springer-Verlag-Verlag, 1998.
8. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
9. Shuhong Gao and Hendrik W. Lenstra, Jr. Optimal normal bases. *Designs, Codes and Cryptography*, 2:315–323, 1992.
10. Louis Goubin. A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems. In Y. Desmedt, editor, *Public Key Cryptography (PKC 2003)*, volume 2567 of *Lecture Notes in Computer Science*, pages 199–210. Springer-Verlag, 2003.
11. Darrel Hankerson, Julio López Hernandez, and Alfred Menezes. Software implementation of elliptic curve cryptography over binary fields. In Ç.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 1–24. Springer-Verlag, 2000.
12. Yvonne Hitchcock and Paul Montague. A new elliptic curve scalar multiplication algorithm to resist simple power analysis. In L.M. Batten and J. Seberry, editors, *Information Security and Privacy (ACISP 2002)*, volume 2384 of *Lecture Notes in Computer Science*, pages 214–225. Springer-Verlag, 2002.
13. Kouichi Itoh, Jun Yajima, Masahiko Takenaka, and Naoya Torii. DPA countermeasures by improving the window method. In B.S. Kaliski Jr., Ç.K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 303–317. Springer-Verlag, 2003.

14. Marc Joye and Christophe Tymen. Protections against differential analysis for elliptic curve cryptography: An algebraic approach. In Ç.K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems (CHES 2001)*, volume 2162 of *Lecture Notes in Computer Science*, pages 377–390. Springer-Verlag-Verlag, 2001.
15. Neal Koblitz. CM-curves with good cryptographic properties. In J. Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 279–287. Springer-Verlag, 1992.
16. Paul Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In N. Koblitz, editor, *Advances in Cryptology – CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer-Verlag, 1996.
17. Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.
18. Alfred J. Menezes. *Elliptic curve public key cryptosystems*. Kluwer Academic Publishers, 1993.
19. François Morain and Jørge Olivos. Speeding up the computations on an elliptic curve using addition-subtraction chains. *Inform. Theor. Appl.*, 24:531–543, 1990.
20. Katsuyuki Okeya, Kunihiko Miyazaki, and Kouichi Sakurai. A fast scalar multiplication method with randomized projective coordinates on a Montgomery-form elliptic curve secure against side channel attacks. In K. Kim, editor, *Information and Communications Security*, volume 2288 of *Lecture Notes in Computer Science*, pages 428-439. Springer-Verlag, 2002.
21. Katsuyuki Okeya and Kouichi Sakurai. Power analysis breaks elliptic curve cryptosystems even secure against the timing attack. In B.K. Roy and E. Okamoto, editors, *Progress in Cryptology – INDOCRYPT 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 178–190. Springer-Verlag, 2000.
22. Richard Schroeppel, Hilarie Orman, Sean W. O'Malley, and Oliver Spatscheck. Fast key exchange with elliptic curve systems. In D. Coppersmith, editor, *Advances in Cryptography – CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 43–56. Springer-Verlag, 1995.
23. Jerome A. Solinas. An improved algorithm for arithmetic on a family of elliptic curves. In B.S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 357–371. Springer-Verlag, 1997.
24. Jerome A. Solinas. Efficient arithmetic on Koblitz curves. *Designs, Codes and Cryptography*, 19:195–249, 2000.
25. Jerome A. Solinas. Low-weight binary representations for pairs of integers. Technical Report CORR 2001-41, CACR, Waterloo, 2001. Available at URL `http://www.cacr.math.uwaterloo.ca/~techreports/2001/corr2001-41.ps`.

A NAF-Based Regular Point Multiplication Algorithms

The computation of the inverse of a point $\boldsymbol{P} = (x, y)$ on an elliptic curve is free. So, the m-ary point multiplication algorithms for computing $\boldsymbol{Q} = [k]\boldsymbol{P}$ can be speeded up by using a signed representation for k. In particular, for $m = 2$, a non-adjacent form (NAF) representation —that is, representing k as $k = \sum_{i=0}^{\ell} \kappa_i \, 2^i$ with $\kappa_i \in \{-1, 0, 1\}$ and $\kappa_i \cdot \kappa_{i-1} = 0, \forall i$— gives rise to a speedup factor of $\approx 11\%$ [19].

At first glance, NAFs do not seem to help in reducing the complexity of the 'square-and-multiply always' algorithm. However, the non-adjacency property, $\kappa_i \cdot \kappa_{i-1} = 0$, can be exploited by scanning two digits per iteration. We consider the following three cases and the corresponding operations to be performed (point doublings and point additions/subtractions are respectively denoted by D and A and underlined symbols represent dummy operations):

- $(\kappa_i, \kappa_{i-1}) = (0, 0)$: $\mathsf{D}\ \mathsf{D}\ \underline{\mathsf{A}}\ \underline{\mathsf{D}}$;
- $(\kappa_i, \kappa_{i-1}) = (0, \pm 1)$: $\mathsf{D}\ \mathsf{D}\ \mathsf{A}\ \underline{\mathsf{D}}$;
- $(\kappa_i, \kappa_{i-1}) = (\pm 1, 0)$: $\underline{\mathsf{D}}\ \mathsf{D}\ \mathsf{A}\ \mathsf{D}$.

The cases $(\kappa_i, \kappa_{i-1}) = (\pm 1, \pm 1)$ and $(\kappa_i, \kappa_{i-1}) = (\pm 1, \mp 1)$ never occur. The resulting algorithm is depicted on the next figure. Function sign$(\cdot)$ returns the sign of an integer (i.e., if $a \geq 0$ then $\mathrm{sign}(a) = 0$ and $\mathrm{sign}(a) = 1$ if $a < 0$).

Input: $\boldsymbol{P}, k = (1, \kappa_{\ell-1}, \ldots, \kappa_0)_{\mathrm{NAF}}$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P}$

```
R_0 ← P; i ← ℓ − 1
while (i ≥ 1) do
   h ← |κ_i|; R_h ← [2]R_h; R_0 ← [2]R_0
   b ← ¬|κ_i + κ_{i−1}|; s ← ¬ sign(κ_i + κ_{i−1})
   R_s ← −R_s; R_b ← R_b + P; R_s ← −R_s
   h ← ¬h; R_h ← [2]R_h
   i ← i − 2
endwhile
h ← |i|; R_h ← [2]R_h
b ← h ∨ ¬|κ_0|; s ← ¬ sign(κ_0)
R_s ← −R_s; R_b ← R_b + P; R_s ← −R_s
return R_0
```

Fig. 4. Simple NAF-based variant of the 'double-and-add always' algorithm

This algorithm is highly regular: at each iteration, there are two point doublings followed by a point addition and a point doubling, whatever the values of scanned digits. The cost per digit is $\frac{3}{2}$ point doublings and $\frac{1}{2}$ point addition; this has to be compared to the 1 point doubling and 1 point addition of the 'double-and-add always' algorithm. In Jacobian coordinates, a point doubling costs 8 multiplies when parameter $a = -3$ and 10 multiplies in the general case whereas a point addition costs 11 multiplies, provided that the Z-coordinate of $\boldsymbol{P}$ is set to 1 and 16 multiplies in the general case. Therefore, the algorithm of Fig. 4 is up to $\approx 8\%$ faster with the same memory requirements (and $\approx 17\%$ faster with randomized representations; see Table 1).

A more involved algorithm using similar ideas was proposed by Hitchcock and Montague [12]. It basically corresponds to

- $(\kappa_i, \kappa_{i-1}) = (0, 0)$: D D $\underline{\text{A}}$;
- $(\kappa_i, \kappa_{i-1}) = (0, \pm 1)$: D D A ;
- $(\kappa_i) = (\pm 1)$: $\underline{\text{D}}$ D A .

According to [12], the expected cost per digit is $\frac{10}{9}$ point doublings and $\frac{5}{9}$ point addition. The corresponding number of field multiplications for computing $[k]\boldsymbol{P}$ is listed in Table 1. As presented in [12], a 'SPA-resistant NAF formatting' algorithm is needed prior to the computation of $\boldsymbol{Q} = [k]\boldsymbol{P}$. We give hereafter a variant that does not require a prior recoding.

Input: $\boldsymbol{P}, k = (1, \kappa_{\ell-1}, \ldots, \kappa_0)_{\mathrm{NAF}}$
Output: $\boldsymbol{Q} = [k]\boldsymbol{P}$

```
R_0 ← P; i = ℓ − 1
while (i ≥ 1) do
    h ← |κ_i|; R_h ← [2]R_h; R_0 ← [2]R_0
    b ← ¬|κ_i + κ_{i−1}|; s ← ¬ sign(κ_i + κ_{i−1})
    R_s ← −R_s; R_b ← R_b + P; R_s ← −R_s
    i ← i − 1 − ¬h
endwhile
h ← |i|; R_h ← [2]R_h
b ← h ∨ ¬|κ_0|; s ← ¬ sign(κ_0)
R_s ← −R_s; R_b ← R_b + P; R_s ← −R_s
return R_0
```

Fig. 5. Modified Hitchcock-Montague algorithm (without recoding algorithm)

There is an important class of elliptic curves, which consists of the so-called *anomalous binary curves* (ABC for short) first proposed by Koblitz [15]. An ABC curve over $\mathbb{F}_{2^n}$ is given by the Weierstraß equation

$$E_{/\mathbb{F}_{2^m}} : y^2 + xy = x^3 + ax^2 + 1 \quad \text{with } a \in \mathbb{F}_2 \ .$$

Let τ denote the Frobenius endomorphism, $\tau(x, y) := (x^2, y^2)$. In [22,23, 24], methods are proposed to decompose an integer k as $k = \sum_i \kappa_i \, \tau^i$ with $\kappa_i \in \{-1, 0, 1\}$ and $\kappa_i \cdot \kappa_{i-1} = 0$, and the double-and-add algorithm is replaced by a τ-and-add algorithm, where τ application consists in two squarings. This method is particularly useful when optimal normal bases are used for representing elements $\mathbb{F}_{2^m}$, see [9]. In that case, an adaptation of the simple NAF-based algorithm (Fig. 4) is more advantageous than the corresponding adaptation of the Hitchcock-Montague algorithm (Fig. 5) since, neglecting τ applications, the (expected) cost per digit amounts to $\frac{1}{2}$ point addition vs. $\frac{5}{9}$ point addition.