Algorithmes de multiplication scalaire réguliers et efficaces

Alexandre VENELLI

09/12/2010
Séminaire ATI

Sommaire

- Cryptosystèmes à base de courbes elliptiques (ECC)
- Attaques par canaux cachés sur ECC
- Algorithmes classiques de multiplication scalaire résistants aux attaques par canaux cachés
- Propositions d'algorithmes efficaces et résistants

Courbes elliptiques

- On ne considère que les courbes sous forme de Weierstrass simplifiée comme spécifié dans les normes internationales (FIPS, ANSI, SEC)

- Courbe elliptique définie sur un corps de caractéristique 2 :

$$E:\ y^2 + xy = x^3 + ax^2 + b \quad (a, b \in GF(2^n), b \neq 0)$$

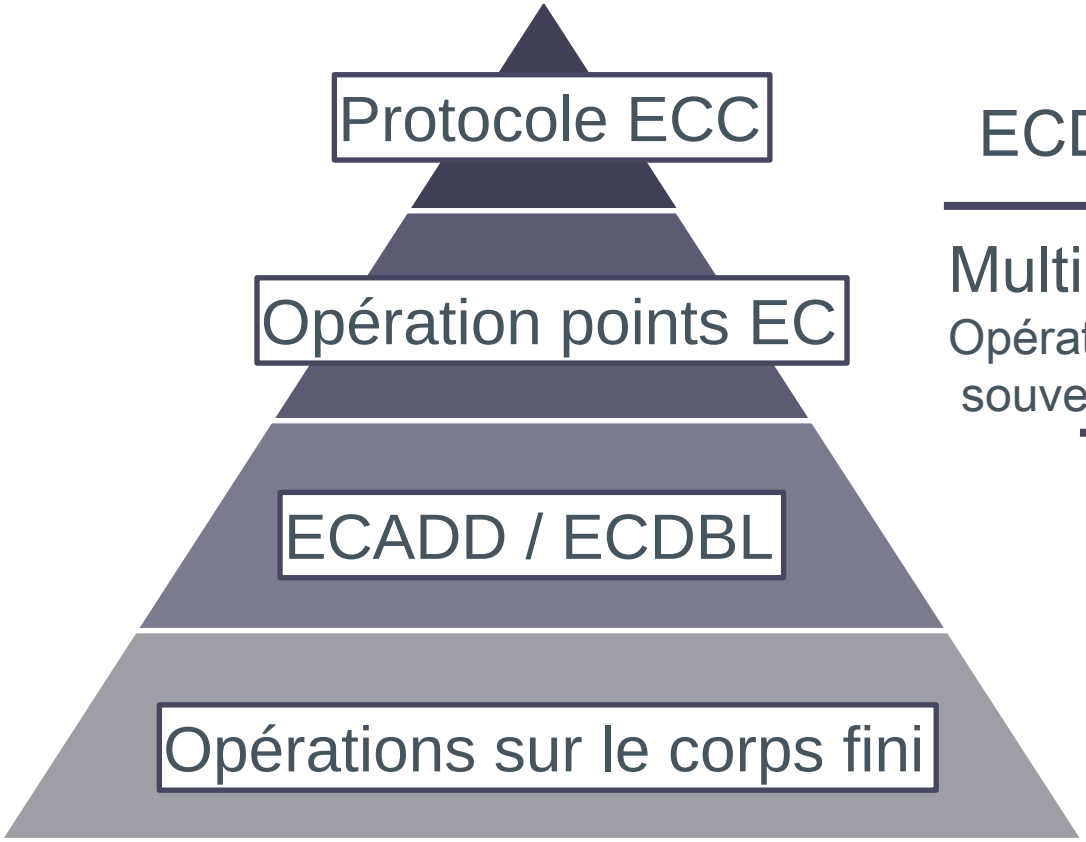
- Courbe elliptique définie sur un corps de grande caractéristique :

$$E:\ y^2 = x^3 + ax + b \quad (a, b \in GF(p), 4a^3 + 27b^2 \neq 0, p > 3)$$

Opérations sur les points

- Soit $P_1 = (x_1, y_1), P_2 = (x_2, y_2), P_3 = (x_3, y_3) \in E$
- Doublement de point (ECDBL) : $P_3 = P_1 + P_1 = 2P_1$
- Addition de points (ECADD) : $P_3 = P_1 + P_2 \quad (P_1 \neq P_2)$
- Sur GF(p), avec des coordonnées projectives jacobiennes :
  - ECDBL = 1M+8S
  - ECADD = 11M+5S
- Sur GF(2^n), avec des coordonnées projectives López-Dahab :
  - ECDBL = 3M+5S
  - ECADD = 13M+4S
- Référence : http://www.hyperelliptic.org/EFD/

Hiérarchie des opérations ECC



ECDSA, ECDH, …

Multiplication scalaire : kP

Opération fondamentale qui consomme souvent le plus de temps

Addition de points

Doublement de point

GF addition :	a + b mod p
GF soustraction:	a – b mod p
GF multiplication :	a * b mod p
GF inverse :	1 / a mod p

Multiplication scalaire classique

- Multiplication scalaire : kP
  - Doublement-et-addition $P \in E, \quad k = \underbrace{(k_{n-1} \cdots k_0)_2}_{\text{représentation binaire}}, k_{n-1} = 1$

1. $Q \leftarrow P$
2. Pour $i = n-2$ jusqu'à 0

   $Q \leftarrow 2Q$ ECDBL

   si $k_i = 1$ alors $Q \leftarrow Q + P$ ECADD

3. Retourner Q

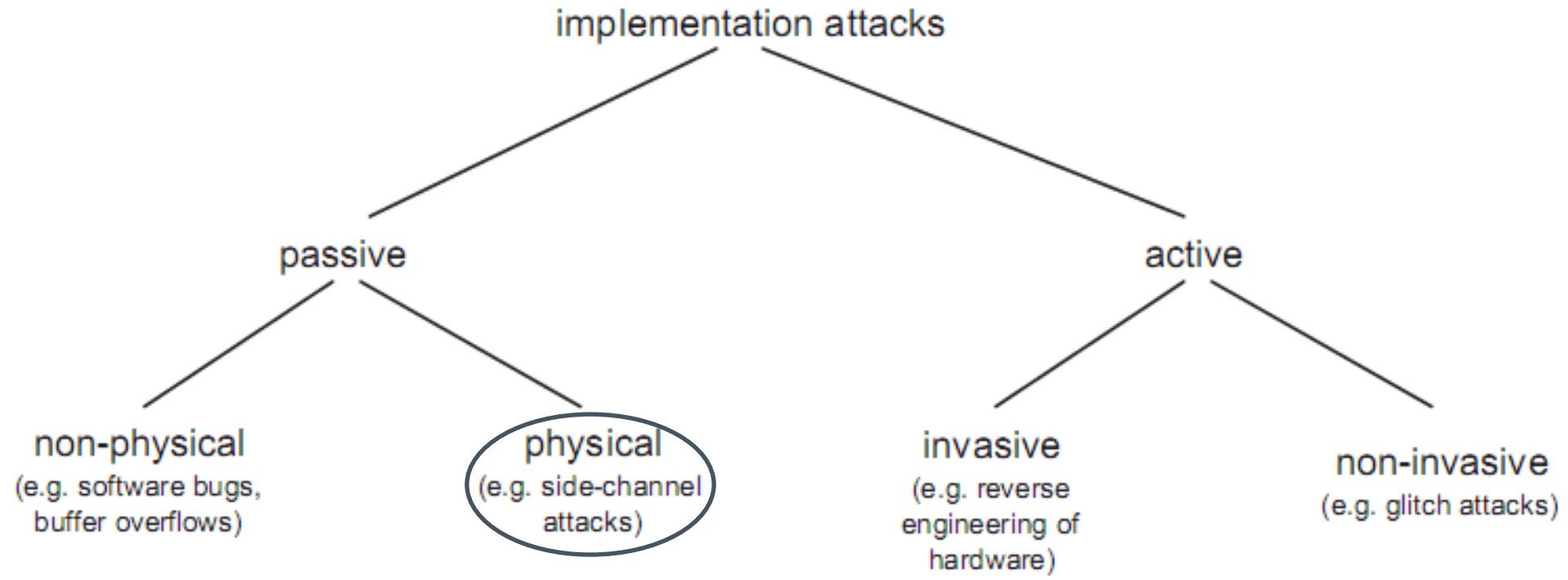
Addition « simplifiée »

- Aussi appelée addition co-Z
- Soit $P_1 = (X_1, Y_1, \boxed{Z}), P_2 = (X_2, Y_2, \boxed{Z}) \in E$

$$\mathrm{ZADDU}(P_1, P_2) \rightarrow (\tilde{P}_1, P_1 + P_2) \text{ avec } Z_{\tilde{P}_1} = Z_{P_1+P_2}$$

- Sur GF(p), coordonnées projectives jacobiennes :
  - ZADDU = 5M+2S (Méloni 2007)
- Sur GF(2^n), coordonnées projectives jacobiennes :
  - ZADDU = 7M+2S (Venelli et al. SAR-SSI 2010)
- Ne peut pas être utilisé tel quel dans un algorithme de multiplication scalaire

Attaques matérielles



Familles d'attaques par canaux cachés

- **Attaques par analyse simple de courant (SPA)**
  L'attaquant observe la consommation de courant du composant lors d'un calcul cryptographique et retrouve la clé secrète

- **Attaques par analyse différentielle de courant (DPA)**
  L'attaquant observe plusieurs courbes de consommation et retrouve le secret à l'aide d'outils statistiques

- **Attaques par injection de faute (FA)**
  L'attaquant utilise des résultats de calculs corrects, des résultats faux dus à une injection de faute et l'endroit précis où la faute a été effectuée pour retrouver le secret

SPA sur ECC (Coron 1999)

Algorithme 2: *Left-to-right* doublement-et-addition

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$
Sorties : $[k]P \in E$

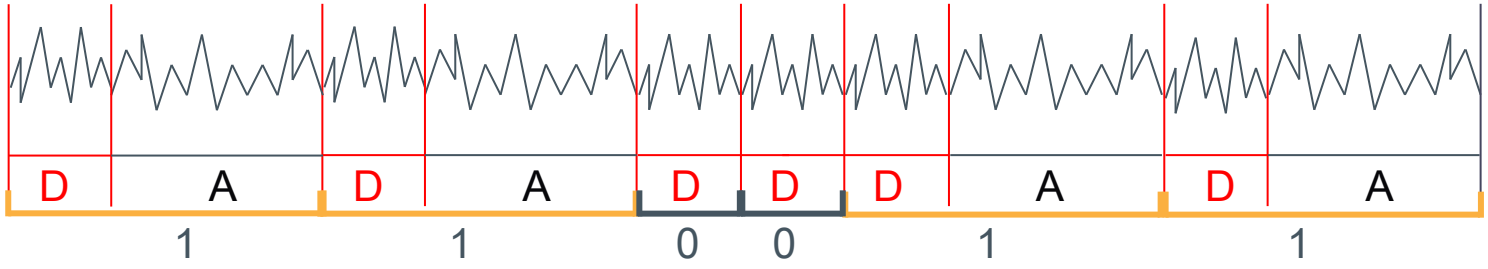
1 $P_0 \leftarrow \mathcal{O}$
2 $P_1 \leftarrow P$
3 **pour** $i \leftarrow n-1$ **a** 0 **faire**
4 $\quad P_0 \leftarrow [2]P_0$ — ECDBL
5 $\quad$ **si** $k_i = 1$ **alors**
6 $\quad\quad P_0 \leftarrow P_0 + P_1$ — ECADD
7 **retourner** P_0

Consommation de courant des opérations ECC

- **ECDBL**

- **ECADD**

Ex : $51P = (110011)_2 P$



Contre-mesures SPA

- Rendre ECADD et ECDBL indistinguables
  - Formules unifiées pour les courbes sous forme de Weierstrass simplifiée : très coûteuses
  - Utiliser des familles de courbes spécifiques (Jacobi, Hesse, Edwards, Huff) : pas utilisable dans notre cas

- Utiliser un algorithme de multiplication scalaire régulier
  - Doublement-et-toujours-addition (Coron 1999)
  - Doublement-et-addition atomique (Chevallier-Mames et al. 2004)
  - Chaînes d'addition Euclidiennes (Méloni 2007)
  - Echelle de Montgomery (Montgomery 1987, Brier et al. 2002)
  - Doublement-et-addition de Joye (Joye 2007)

Rendre ECADD et ECDBL indistinguables

Formules unifiées sur courbes Weierstrass simplifiée

- Avoir un même algorithme pour calculer ECADD et ECDBL
- Brier et al. 2004 :
  - Sur corps de grande caractérisque
  - 16M+3S en coord. proj stand
- Problèmes :
  - Coûteux
  - Attaques par canaux cachés ont été proposées

Familles de courbes

sur corps de large caractéristique

- Hesse
  - Point d'ordre 3
  - 12M (Farashahi et al. 2010)
- Jacobi
  - Point d'ordre 2
  - 10M+3S (Billet et al. 2003)
- Edwards
  - Point d'ordre 4 / Point d'ordre 2 (Edwards tordu)
  - 10M+1S (Bernstein et al. 2008)
- Huff
  - Point d'ordre 4
  - 12M (Joye et al. 2010)

Familles de courbes

sur corps de caractéristique 2

- Edwards
  - Toute courbe elliptique
  - 18M+2S+7D ou 21M+1S+4D (Bernstein et al. 2008)
- Huff
  - Toute courbe elliptique (generalized binary Huff curves)
  - 15M+3S+2D (Joye et al. 2010)
  - unifiée mais complète seulement pour certains sous-groupes de points (notamment ceux utilisés en crypto)
- Modèle de Wu et al. 2010 (http://eprint.iacr.org/2010/608)
  - Point d'ordre 4
  - 12M+4S+2D ou 13M+2D

Utiliser un algorithme de multiplication scalaire régulier

Doublement-et-toujours-addition (Coron 1999)

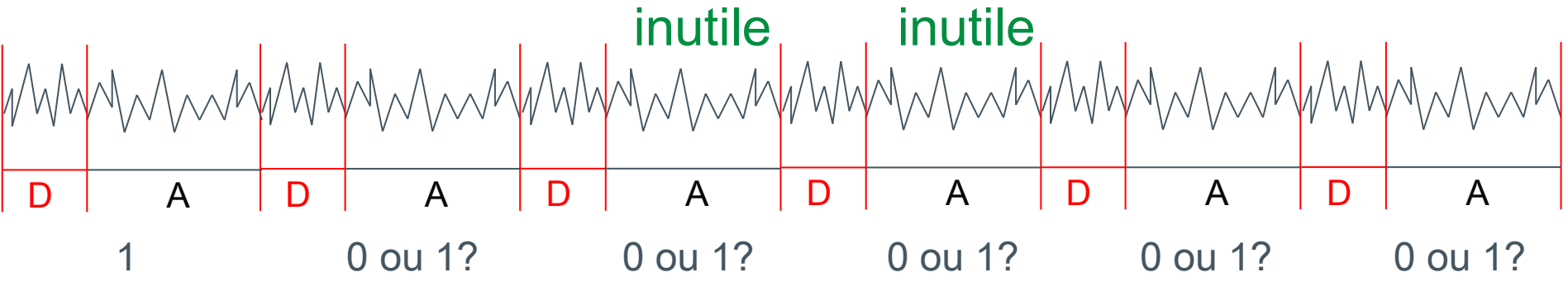
Algorithme 7: Doublement-et-toujours-addition

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$
Sorties : $[k]P \in E$

1 $P_0 \leftarrow \mathcal{O}$
2 $P_1 \leftarrow P$
3 **pour** $i \leftarrow n-1$ **a** 0 **faire**
4 $P_0 \leftarrow [2]P_0$ **ECDBL**
5 $P_1 \leftarrow P_0 + P$ **ECADD**
6 $P_0 \leftarrow P_{k_i}$
7 **retourner** P_0

Ex :

$$51P = (110011)_2 P$$



Doublement-et-addition atomique (Chevallier-Mames et al. 2004)

- Principe : construire ECADD et ECDBL en répétant un motif fixé d'opérations du corps de base afin de rendre ECADD et ECDBL indistinguables
- Résistant aux SPA
- Problème : beaucoup d'opérations inutiles sont ajoutées
- Faiblesse : sensible aux FA

Chaînes d'addition Euclidiennes (Méloni 2007)

- Algorithme de multiplication scalaire + Addition co-Z
  - Représenter le scalaire de manière différente (EAC)

- Ex : Soit k=34, g=19 premier à k
  EAC(34,19) = (1,2,3,4,7,11,15,19,34)
  34-19=15 (big step)
  19-15=4 (small)
  15-4 =11 (small)
  11-4 =7 (big)
  7-4 =3 (big)
  4-3 =1 (small)
  3-1 =2
  2-1 =1
  1-1 =0

Chaînes d'addition Euclidiennes (2)

- Avantages :
  - Permet d'utiliser la formule d'addition co-Z pour un algorithme de multiplication scalaire
  - Algorithme qui est résistant aux attaques SPA (on n'utilise pas de ECDBL)

- Inconvénients :
  - Convertir un entier dans une représentation EAC courte n'est pas facile
  - Performances de l'algorithme de multiplication scalaire sont donc impactées

Familles d'attaques par canaux cachés

- **Attaques par analyse simple de courant (SPA)**
  L'attaquant observe la consommation de courant du composant lors d'un calcul cryptographique et retrouve la clé secrète

- **Attaques par analyse différentielle de courant (DPA)**
  L'attaquant observe plusieurs courbes de consommation et retrouve le secret à l'aide d'outils statistiques

- **Attaques par injection de faute (FA)**
  L'attaquant utilise des résultats de calculs corrects, des résultats faux dus à une injection de faute et l'endroit précis où la faute a été effectuée pour retrouver le secret

DPA sur ECC

- Soit k un scalaire fixé, soit P un point / message qu'un attaquant peut fixer
  - Ex : EC ElGamal, EC Diffie-Hellman, …
- La consommation de courant d'un algorithme comme doublement-et-toujours-addition **semble** identique, mais il existe de faibles différences
- Ces différences sont liées à la valeur des bits de la représentation de P et du scalaire k

Contre-mesures DPA

- Contre-mesures indépendantes du choix de l'algorithme de multiplication scalaire

- Techniques principales :
  1. Modifier le scalaire
     - **Randomisation,**
     - **Découpage aléatoire, …**
  2. Modifier le point de base
     - **Randomisation des coordonnées projectives,**
     - **Isomorphismes de courbes aléatoires,**
     - **Isomorphismes de corps aléatoires, …**

Découpage aléatoire du scalaire

- Découpage ‘additif’, Clavier et al. 2001 :

$$kP = (k - r)P + rP$$

- Découpage ‘multiplicatif’, Trichina et al. 2002 :

$$kP = (kr^{-1})(rP)$$

- Découpage ‘euclidien’, Ciet et al. 2003 :

$$kP = (k \bmod r)P + \lfloor k / r \rfloor (rP)$$

Randomisation des coordonnées projectives (Coron 1999)

- Soit $P = (X, Y, Z)$ en coordonnées projectives jacobiennes
- On a $(X, Y, Z) \cong (\lambda^2 X, \lambda^3 Y, \lambda Z)$ pour $\lambda \in \mathrm{K}^*$
- On peut modifier la représentation de P de manière aléatoire

Familles d'attaques par canaux cachés

- **Attaques par analyse simple de courant (SPA)**
  L'attaquant observe la consommation de courant du composant lors d'un calcul cryptographique et retrouve la clé secrète

- **Attaques par analyse différentielle de courant (DPA)**
  L'attaquant observe plusieurs courbes de consommation et retrouve le secret à l'aide d'outils statistiques

- **Attaques par injection de faute (FA)**
  L'attaquant utilise des résultats de calculs corrects, des résultats faux dus à une injection de faute et l'endroit précis où la faute a été effectuée pour retrouver le secret

Doublement-et-toujours-addition

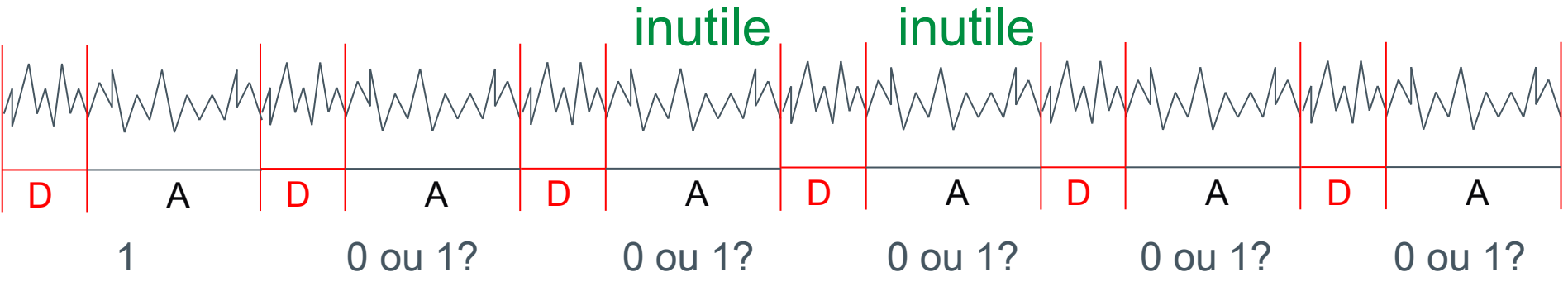
Algorithme 7: Doublement-et-toujours-addition

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$
Sorties : $[k]P \in E$

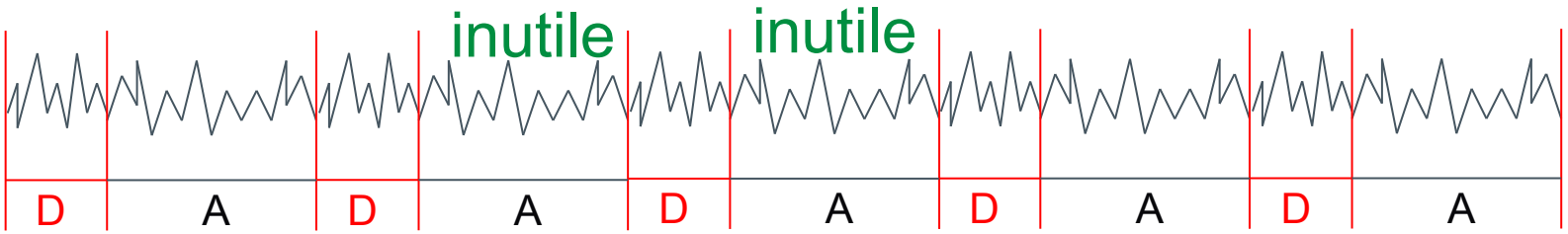
1 $P_0 \leftarrow \mathcal{O}$
2 $P_1 \leftarrow P$
3 **pour** $i \leftarrow n-1$ **a** 0 **faire**
4 $\quad P_0 \leftarrow [2]P_0$ **ECDBL**
5 $\quad P_1 \leftarrow P_0 + P$ **ECADD**
6 $\quad P_0 \leftarrow P_{k_i}$
7 **retourner** P_0

Ex :

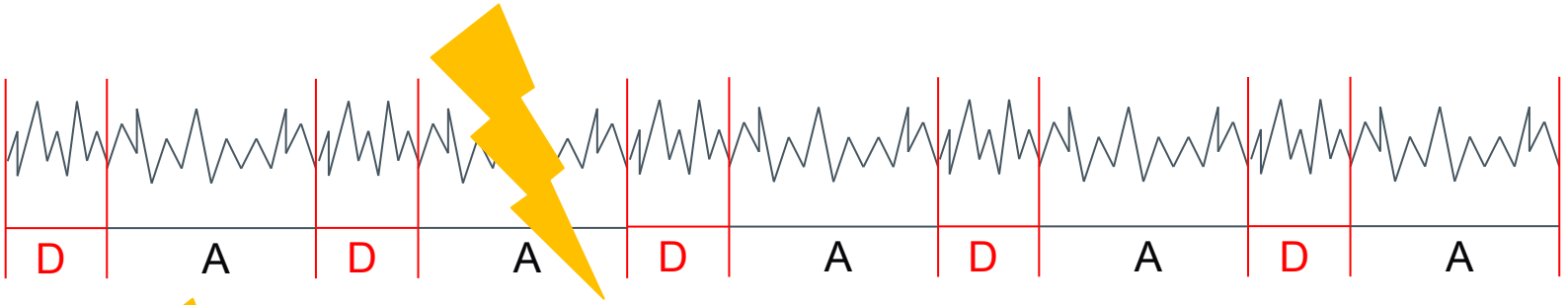
$$51P = (110011)_2 P$$



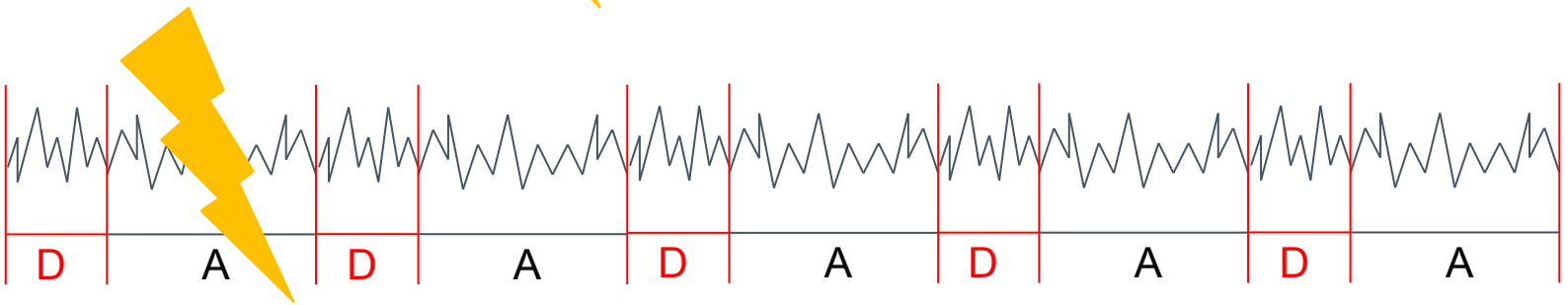
Résistant aux SPA mais pas aux FA



$= 51P$



$= 51P$



$\neq 51P$

Contre-mesures FA

- Vérifier que le point, résultat de la multiplication, est *valide* :
  - P n'est pas le point à l'infini
  - Les coordonnées de P sont bien des éléments du corps de base
  - P satisfait l'équation de la courbe
  - Vérifier $nP = \infty$ avec n le cardinal du sous-groupe premier utilisé

- Faire en sorte que tous les résultats intermédiaires soient utilisés pour le calcul du résultat final
  - faute injectée → résultat faux dans tous les cas
  - dépend de l'algorithme choisi

Echelle de Montgomery

Algorithme

Algorithme 9: Échelle de Montgomery

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$

Sorties : $[k]P \in E$

1 $P_0 \leftarrow \mathcal{O}$

2 $P_1 \leftarrow P$

3 **pour** $i \leftarrow n-1$ **a** 0 **faire**

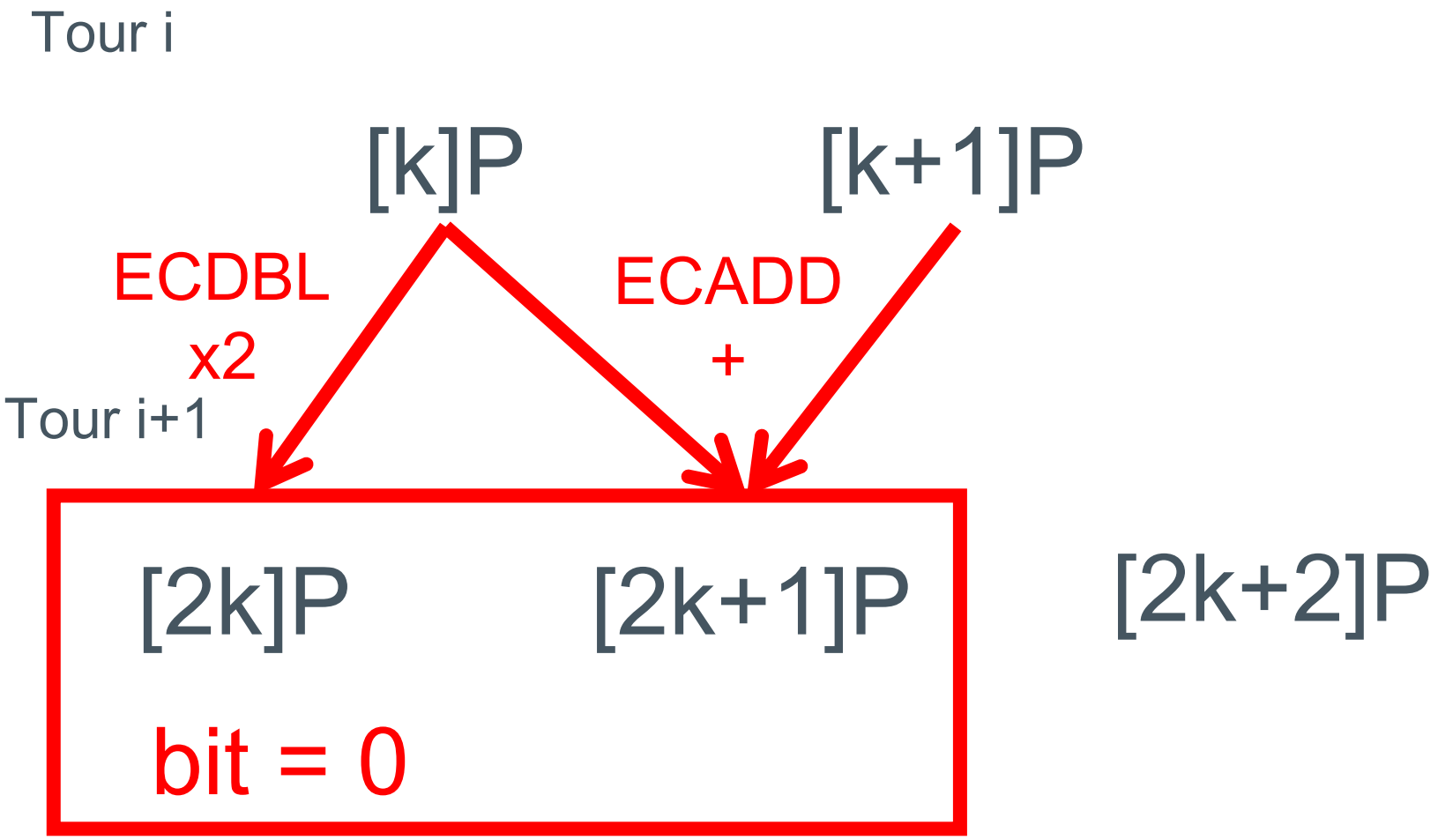
4 $\quad P_{\bar{k_i}} \leftarrow P_{\bar{k_i}} + P_{k_i}$

5 $\quad P_{k_i} \leftarrow [2]P_{k_i}$

6 **retourner** P_0

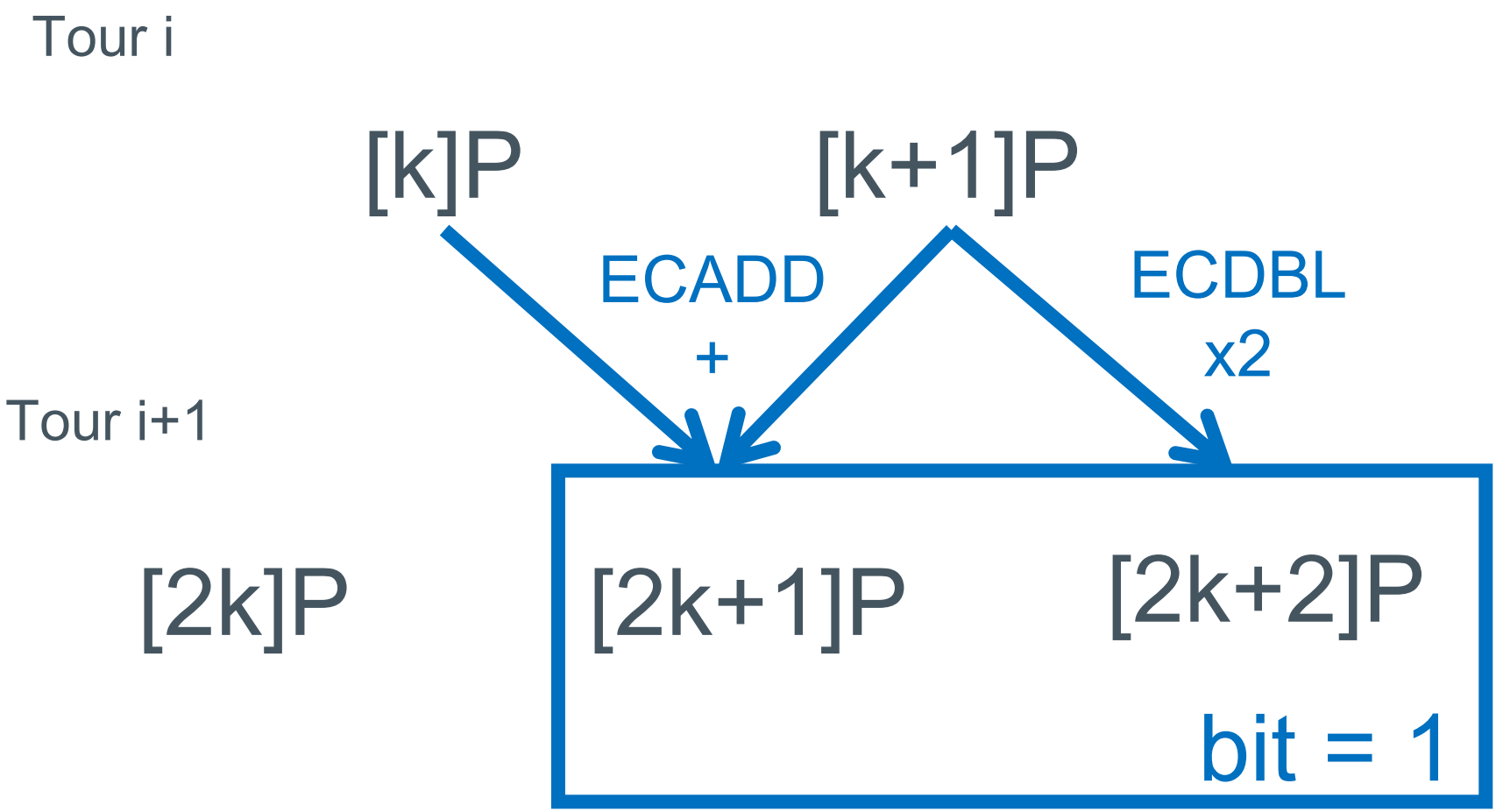
Echelle de Montgomery

Principe (1)



Echelle de Montgomery

Principe (2)



Doublement-et-addition de Joye

Algorithme 10: Doublement-et-addition de Joye

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$

Sorties : $[k]P \in E$

1 $P_0 \leftarrow \mathcal{O}$

2 $P_1 \leftarrow P$

3 **pour** $i \leftarrow 0$ **a** $n-1$ **faire**

4 $\quad P_{\bar{k_i}} \leftarrow [2]P_{\bar{k_i}}$

5 $\quad P_{\bar{k_i}} \leftarrow P_{\bar{k_i}} + P_{k_i}$

6 **retourner** P_0

Equivalent de l'échelle de Montgomery de « droite à gauche »

Proposition

(Venelli et al. Geocrypt 2009)

- Combiner l'échelle de Montgomery avec l'addition co-Z (ZADDU) de Méloni pour obtenir un algorithme résistant et efficace

- Problème :
  - L'échelle de Montgomery effectue un ECDBL à chaque tour
  - Le tour suivant, si on utilise ZADDU, il faut que les coordonnées Z des deux points soient égales
  - Modifier la sortie de ECDBL en conséquence est beaucoup trop coûteux

- Solution : supprimer ECDBL, n'utiliser que des additions

Remplacer ECDBL

- Un tour de l'échelle de Montgomery :

$$P_{\bar{k}_i} \leftarrow P_{\bar{k}_i} + P_{k_i} \quad \text{et} \quad P_{k_i} \leftarrow [2]P_{k_i}$$

- On peut remplacer ECDBL par : $(P_{k_i} + P_{\bar{k}_i}) + (P_{k_i} - P_{\bar{k}_i})$

- Soit T une variable temporaire. On obtient le tour de boucle :

$$T \leftarrow P_{k_i} - P_{\bar{k}_i}$$
$$P_{\bar{k}_i} \leftarrow P_{\bar{k}_i} + P_{k_i} \quad \text{ECADD (1)}$$
$$P_{k_i} \leftarrow P_{\bar{k}_i} + T \quad \text{ECADD (2)}$$

Echelle de Montgomery modifiée

Algorithme

Algorithme 11: Échelle de Montgomery avec additions

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$

Sorties : $[k]P \in E$

1 $P_0 \leftarrow \mathcal{O}$

2 $P_1 \leftarrow P$

3 **pour** $i \leftarrow n-1$ **a** 0 **faire**

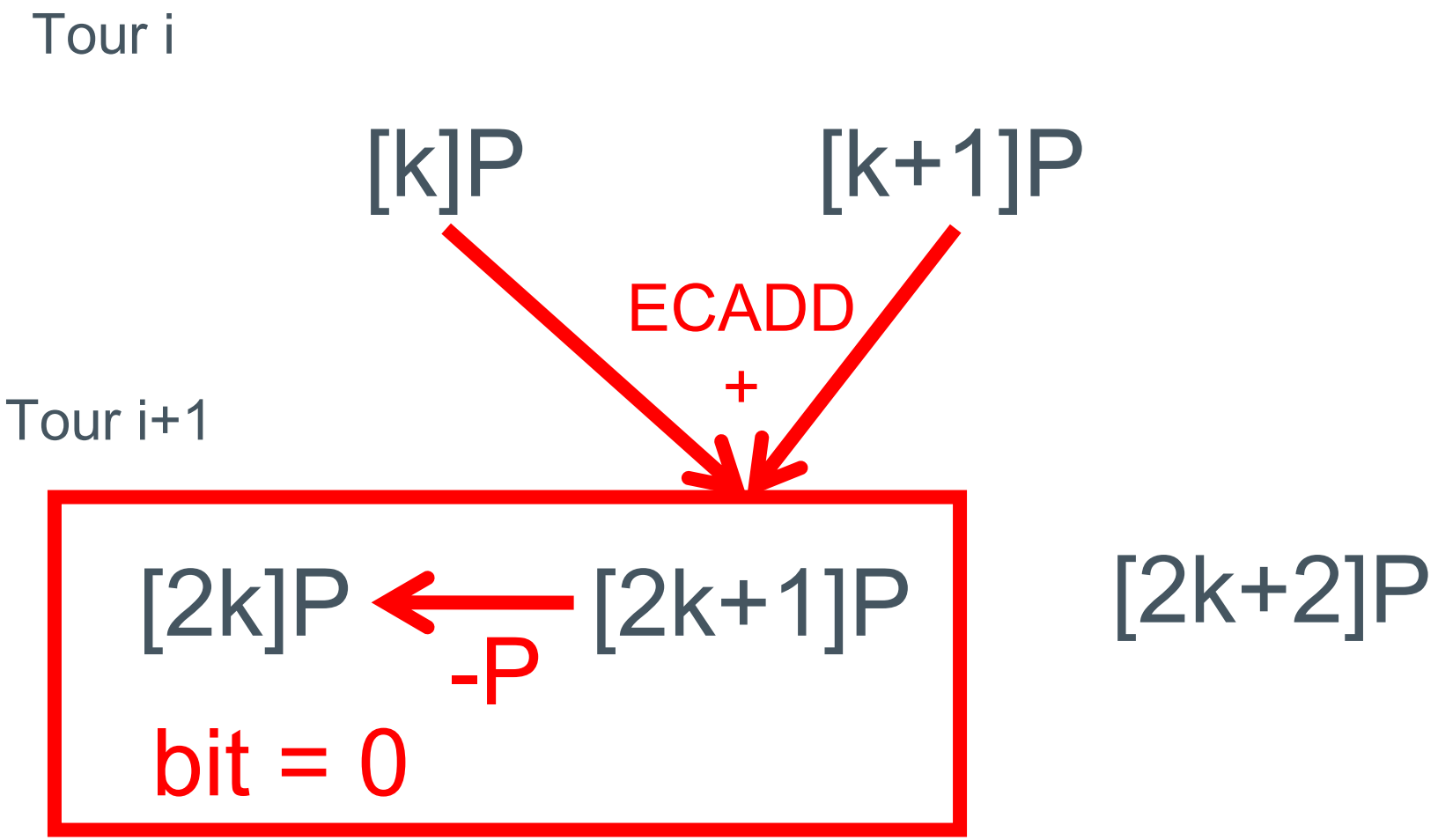
4 $\quad P_0 \leftarrow P_0 + P_1$

5 $\quad P_1 \leftarrow P_0 + (-1)^{\bar{k}_i} P$

6 **retourner** P_1

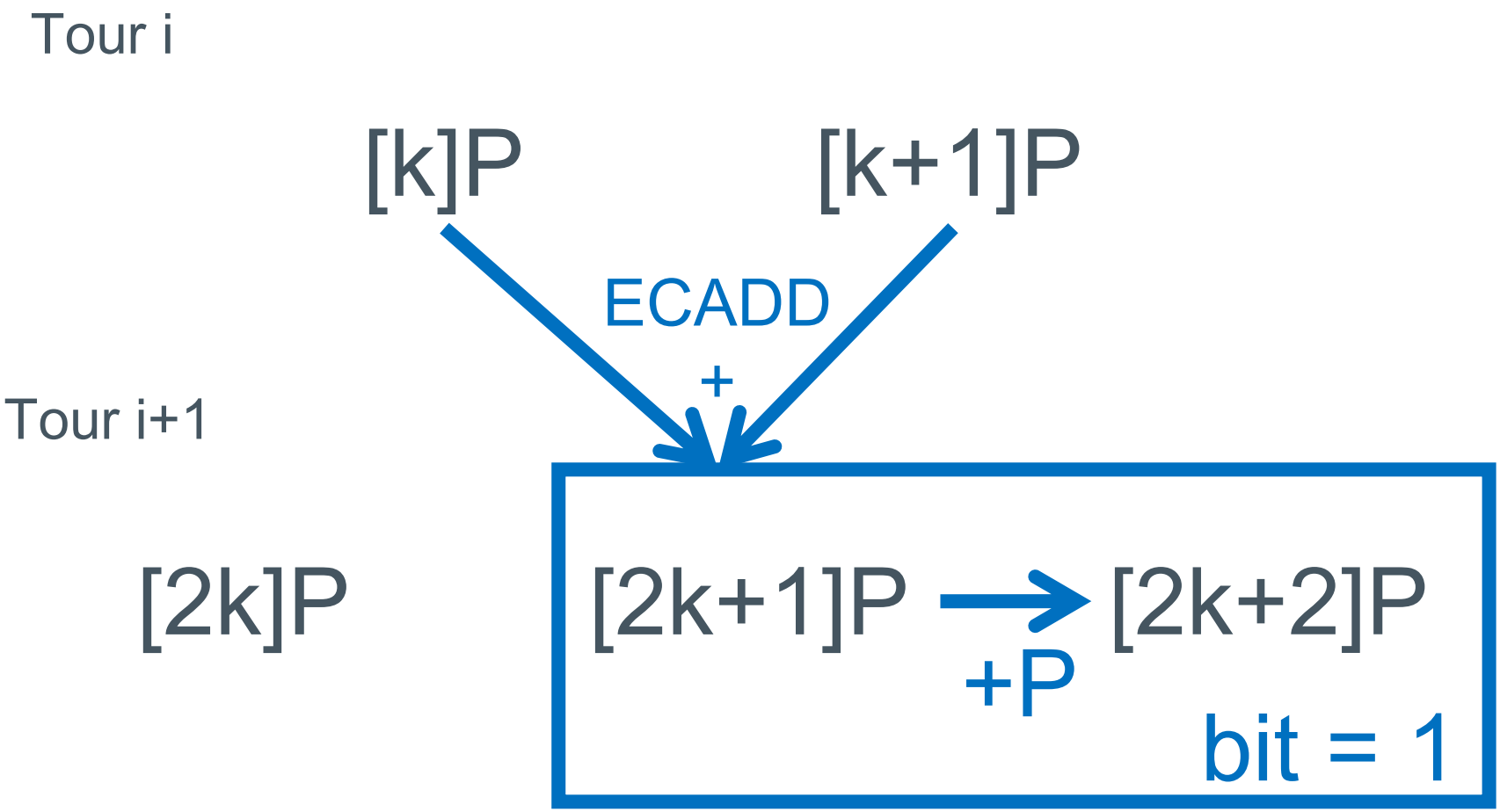
Echelle de Montgomery modifiée

Principe (1)



Echelle de Montgomery modifiée

Principe (2)



Modifier l'addition co-Z

- Problème : il faut un point P avec la coordonnée Z correcte à chaque tour
- Solution : modifier l'addition ZADDU afin de calculer en même temps la soustraction de deux points tel que

$$ZADDC \rightarrow (\tilde{P}_1, P_1 + P_2, P_1 - P_2)$$
$$\text{avec } Z_{\tilde{P}_1} = Z_{P_1+P_2} = Z_{P_1-P_2}$$

Complexités

	GF(p)	GF(2^n)
ZADDU	5M+2S	7M+2S
ZADDC	6M+3S	11M+2S

Proposition d'algorithme 1

Algorithme 12: Échelle de Montgomery avec ZADDC

Entrées : $P \in E$ et $k = (k_{n-1} \dots k_1 k_0)_2$
Sorties : $[k]P \in E$

1 $P_0 \leftarrow [2]P$
2 $P_1 \leftarrow P$
// On suppose $Z_{P_0} = Z_{P_1}$
3 **pour** $i \leftarrow n-2$ **a** 0 **faire**
4 $\quad (Q_0, Q_1, Q_2) \leftarrow \texttt{ZADDC}(P_{k_i}, P_{\bar{k}_i})$
5 $\quad P_{\bar{k}_i} \leftarrow Q_1$ /* $P_{\bar{k}_i} \leftarrow (P_0 + P_1)$ */
6 $\quad P_{k_i} \leftarrow Q_2$ /* $P_{k_i} \leftarrow (P_{k_i} - P_{\bar{k}_i}) = \pm P$ */
7 $\quad (Q_0, Q_1, Q_2) \leftarrow \texttt{ZADDC}(P_{\bar{k}_i}, P_{k_i})$
8 $\quad P_{k_i} \leftarrow Q_0$ /* $P_{k_i} \leftarrow \tilde{P}_{\bar{k}_i}$ */
9 $\quad P_{\bar{k}_i} \leftarrow Q_1$ /* $P_{\bar{k}_i} \leftarrow P_0 + P_1$ */
10 **retourner** P_1

Un travail équivalent à cette proposition a été publié par Goundar et al. dans CHES 2010 uniquement pour GF(p).

Proposition d'algorithme 2

- Uniquement sur un corps de grande caractéristique
- On peut **ne pas** calculer la coordonnée Z tout au long de l'algorithme
- La coordonnée Z du point final est recalculée lors du dernier tour de l'algorithme pour un coût de 5M+1I
- Les complexités des algorithmes ZADDU et ZADDC deviennent :
  - ZADDUwoZ = 4M+2S
  - ZADDCwoZ = 5M+3S

Recalculer Z au dernier tour (1)

- A chaque tour, on calcule un $\tilde{P} = (\tilde{X}, \tilde{Y}, \tilde{Z})$
- On sauvegarde le $P = (X, Y, Z)$ de départ
- On peut retrouver facilement

$$\tilde{Z} = \frac{Z\tilde{X}\tilde{Y}}{XY}$$

Recalculer Z au dernier tour (2)

// Dernier tour

$(Q_0, Q_1, Q_2) \leftarrow \texttt{ZADDCwoZ}(P_0, P_1)$

$P_{\bar{k}_i} \leftarrow Q_1$

$P_{k_i} \leftarrow Q_2$

// Calcul de Z_{final}

$Z_{final} \leftarrow X_{P_1} \cdot Y_{P_{save}}$

$Z_{final} \leftarrow (Z_{final})^{-1}$

$Z_{final} \leftarrow Z_{final} \cdot Y_{P_1}$

$Z_{final} \leftarrow Z_{final} \cdot X_{P_{save}}$

$Z_{final} \leftarrow Z_{final} \cdot Z_{P_{save}}$

$Z_{final} \leftarrow (Z_{final} \cdot (X_{P_1} - X_{P_0}))$

$(Q_0, Q_1, Q_2) \leftarrow \texttt{ZADDCwoZ}(P_{\bar{k}_i}, P_{k_i})$

$P_{k_i} \leftarrow Q_0$

$P_{\bar{k}_i} \leftarrow Q_1$

$P_1 \leftarrow [X_{P_1}, Y_{P_1}, Z_{final}]$

Comparaison de performances sur GF(2^n)

Algorithmes	Complexités (pour un bit de scalaire)
ML Basique	16M+9S ≈ 25M
ML X-only	6M+5S ≈ 11M
D&A + Edwards	27M+3S ≈ 30M
D&A + Huff	22,5M+4,5S ≈ 27M
ML+ZADDC+ZADDU (Prop. 1)	**18M+4S ≈ 22M**

Comparaison de performances sur GF(p)

Algorithmes	Complexités (pour un bit de scalaire)
ML Basique	12M+13S ≈ 25M
ML X-only	9M+7S≈ 16M
ML+ZADDU+ZADDC (Prop. 1)	**11M+5S ≈ 16M**
ML+ZADDUwoZ+ZADDCwoZ (Prop. 2)	**9M+5S ≈ 14M**

Conclusion

- Sur GF(p), notre proposition 2 permet d'obtenir l'algorithme de multiplication scalaire le plus efficace en considérant ce niveau de sécurité et pour toute courbe elliptique.

- Sur GF(2^n), notre proposition est loin derrière ML X-only en terme de performances.

- Challenge : trouver une alternative à ML X-only sur GF(2^n) qui offre de meilleures performances pour le même niveau de sécurité pour toutes courbes elliptiques.

Merci de votre attention

Contact : avenelli@insidefr.com