D34: Méthodes de calcul efficaces et sécurisées

Nicolas Méloni

Master 2: 1er semestre
(2013/2014)

Arithmétique des courbes sur $\mathbb{F}_{2^n}$

Définition

Une courbe elliptique sur $\mathbb{F}_{2^n}$ est l'ensemble des solutions d'une équation de la forme:

$$y^2 + xy = x^3 + ax^2 + b,$$

plus un point à l'infini $\mathcal{O}$.

Loi de groupe

On muni la courbe de la loi de groupe suivante:

1. $P + \mathcal{O} = \mathcal{O} + P = P$ pour tout $P \in E(\mathbb{F}_{2^n})$.
2. Soit $P = (x, y) \in E(\mathbb{F}_{2^n})$, on définit $-P$ (ou $\overline{P}$) par: $-P = (x, x + y)$.
3. Soient $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$ deux points sur la courbe tels que $P_1 \neq -P_2$, alors $P_1 + P_2 = (x_3, y_3)$ avec:

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

où $\lambda = \frac{y_2+y_1}{x_2+x_1}$ si $P_1 \neq P_2$ et $\lambda = x_1 + \frac{y_1}{x_1}$ sinon.

Arithmétique des courbes sur $\mathbb{F}_{2^n}$

Cout des opérations

- Addition: $\mathtt{I} + 2\mathtt{M} + \mathtt{S}$
- Doublement: $\mathtt{I} + 2\mathtt{M} + \mathtt{S}$
- Multiplication de point: $\frac{3}{2}\mathtt{I} + 3\mathtt{M} + \frac{3}{2}\mathtt{S}$

Cout des opérations de bases

- $\mathtt{S} \simeq \frac{1}{8}\mathtt{M}$
- $\mathtt{I} \simeq 10\mathtt{M}$

- On ignore en général le cout des élévations au carré.

Doublements succéssifs

- On représente un point (x, y) par (x, λ) où $\lambda = x + \frac{y}{x}$.
- On calcule $[2](x, \lambda) = (x_2, \lambda_2)$ avec:

$$\begin{aligned} x_2 &= \lambda^2 + \lambda + a \\ \lambda_2 &= \lambda^2 + a + \frac{b}{x^4 + b} \end{aligned}$$

- Chaque doublements coute ainsi $\mathtt{I} + \mathtt{M}$.

Définition

- On représente un point $P = (x, y)$ par un triplet $(X : Y : Z)$ vérfiant:
$$(X : Y : Z) \sim \left(\frac{X}{Z}, \frac{Y}{Z^2}\right) \text{ si } Z \neq 0$$
$$(1 : 0 : 0) \sim \mathcal{O}$$
- L'équation de la courbe devient alors:
$$Y^2 + XYZ = X^3Z + aX^2Z^2 + bZ^4.$$
- Pour tout $\lambda \neq 0$, on a $(X : Y : Z) \sim (\lambda X : \lambda^2 Y : \lambda Z)$.
- $-(X : Y : Z) = (X : XZ + Y : Z)$.

Formules d'addition de points

- Soient $P_1 = (X_1 : Y_1 : Z_1)$, $P_2 = (X_2 : Y_2 : Z_2)$ et $P_3 = P_1 + P_2 = (X_3 : Y_3 : Z_3)$:

$$
\begin{aligned}
A &= Y_2 Z_1^2 + Y_1, & B &= X_2 Z_1 + X_1,\\
C &= Z_1 B, & D &= B^2(C + aZ_1^2),\\
E &= AC,\\
Z_3 &= C^2, & X_3 &= A^2 + D + E,\\
F &= X_3 + X_2 Z_3, & G &= X_3 + Y_2 Z_3,\\
Y_3 &= EF + Z_3 G,
\end{aligned}
$$

- Cout total: 14M

Formules de doublement de point

- Soient $P_1 = (X_1 : Y_1 : Z_1)$, $P_3 = [2]P_1$:

$$A = X_1 Z_1, \quad B = bZ_1^4,$$

$$\begin{aligned} X_3 &= X_1^4 + B, \\ Z_3 &= A^2, \\ Y_3 &= BZ_3 + X_3(aZ_3 + Y_1^2 + B) \end{aligned}$$

- Cout total: 4M

Formules de Montgomery

Formules d'addition de points

$$\begin{aligned} X_{m+n} &= (X_m Z_n)^2 + (X_n Z_m)^2, \\ Z_{m+n} &= Z_{m+n} X_{m-n} + X_m Z_n X_n Z_m. \end{aligned}$$

Formules de doublement

$$\begin{aligned} X_{2n} &= (X_n^4 + b Z_n^4) = (X_n^2 + \sqrt{b} Z_n^2)^2, \\ Z_{2n} &= (X_n Z_n)^2. \end{aligned}$$

Cout des opérations

- Addition: $4\mathtt{M} + 1\mathtt{S}$
- Doublement: $2\mathtt{M} + 3\mathtt{S}$

Division de point

Idée Générale

- On cherche à calculer l'image réciproque du morphisme de doublement:

$$\begin{aligned} [2] \quad : E(\mathbb{F}_{2^n}) &\longrightarrow E(\mathbb{F}_{2^n}) \\ P &\longmapsto [2]P. \end{aligned}$$

- Probleme: ce morphisme n'est pas injectif.
- On cherche donc de plus des courbes dites de torsion minimale, pour lesquelles chaque image a au plus deux antécédents.

Division de point

On considère $P = (x, y)$ et $Q = (u, v)$ tel que $(u, v) = [2](x, y)$ (i.e. $P = [\frac{1}{2}]Q$).

- On considère la λ représentation de P et Q $((x, \lambda_P) = (x, x + \frac{y}{x}))$.
- On obtient alors les équations:

$$\begin{aligned} \lambda_P^2 + \lambda_P &= a + u \qquad (i) \\ v &= x^2 + u(\lambda_P + 1) \end{aligned}$$

- Il existe alors deux points solutions de ce système d'équations: P et $P + T_2$ où T_2 est l'unique point de 2-torsion ($[2]T_2 = \mathcal{O}$).
- Pour pouvoir les distinguer, il faut être capable de calculer la bonne solution λ_P de (i) (l'autre étant $\lambda_P + 1$) et pour cela il faut introduire la notion de trace.

Division de point

Définition

- On définit l'application `trace` comme suit:

$$\begin{aligned} \texttt{Tr} \quad : \mathbb{F}_{2^n} &\longrightarrow \mathbb{F}_{2^n} \\ c &\longmapsto \sum_{i=0}^{n-1} c^{2^i}. \end{aligned}$$

Division de point

Propriétés

Soient $c, d \in \mathbb{F}_{2^n}$.

- $\mathtt{Tr}(c^2) = \mathtt{Tr}(c)^2$ (et donc $\mathtt{Tr}(c) \in \mathbb{F}_2$)
- $\mathtt{Tr}(c+d) = \mathtt{Tr}(c) + \mathtt{Tr}(d)$
- Si $(x, y) \in E(\mathbb{F}_{2^n})$ alors $\mathtt{Tr}(x) = \mathtt{Tr}(a)$

Théorème

- Soient $P = (x, y)$ et $Q = (u, v)$ tels que $Q = [2]P$ et $\hat{\lambda}$ une solution de $\lambda^2 + \lambda = u + a$. Posons $t = v + u\hat{\lambda}$. Si $\mathtt{Tr}(a) = 1$ alors $\lambda_P = \hat{\lambda}$ si et seulement si $\mathtt{Tr}(t) = 0$.

Division de point

Algorithm 1 Division de point

Require: $Q = (u, \lambda_Q)$

Ensure: $P = (x, \lambda_P) = [\frac{1}{2}]Q$

1: Calculer une solution $\hat{\lambda}$ de $\lambda^2 + \lambda = u + a$.
2: $t \leftarrow u(u + \lambda_Q + \hat{\lambda})$
3: **if** $\mathtt{Tr}(t) = 0$ **then**
4: $\quad \lambda_P \leftarrow \hat{\lambda}, x \leftarrow \sqrt{t + u}$
5: **else**
6: $\quad \lambda_P \leftarrow \hat{\lambda} + 1, x \leftarrow \sqrt{t}$
7: **end if**
8: **return** (x, λ_P)

Division et addition

- À l'aide de l'algorithme précédent, il est alors possible de concevoir un algorithme de multiplication de point à partir de la proposition suivante:

Proposition

- Soient k un entier de l bits et s un entier impair, alors il existe un rationnel de la forme $\sum_{i=0}^{l-1} \frac{c_i}{2^i}, \quad c_i \in \{0,1\}$ tel que

$$k \equiv \sum_{i=0}^{l-1} \frac{c_i}{2^i} \mod s$$

où $\frac{1}{2^i} \mod s$ représente l'inverse de 2^i modulo s.

Algorithm 2 `Halving-and-Add`

Require: $P \in E(\mathbb{F}_{2^n}), k = \sum_{i=0}^{l-1} \frac{c_i}{2^i}$

Ensure: $\sum_{i=0}^{l-1} \left[\frac{c_i}{2^i}\right] P = [k]P$

1: $Q \leftarrow P$
2: **for** $i = (l-2) \ldots 0$ **do**
3: $\quad Q \leftarrow [\frac{1}{2}]Q$
4: $\quad$ **if** $c_i = 1$ **then**
5: $\quad\quad Q \leftarrow Q + P$
6: $\quad$ **end if**
7: **end for**
8: **return** Q

Calcul de la trace

- Soit $c = \sum_{i=0}^{n-1} c_i X^i \in \mathbb{F}_{2^n}$. Alors la linéarité de la fonction $\mathtt{Tr}$ permet d'obtenir:

$$\mathtt{Tr}(c) = \mathtt{Tr}(\sum_{i=0}^{n-1} c_i X^i) = \sum_{i=0}^{n-1} c_i \mathtt{Tr}(X^i).$$

- On peut alors précalculer les valeurs de $\mathtt{Tr}(X^i)$ et ainsi calculer $\mathtt{Tr}(c)$ efficacement.

Semi trace (half-trace)

Définition

- Pour n impair, on définit l'application `half-trace` comme suit:

$$\begin{aligned} \mathtt{H} \quad : \mathbb{F}_{2^n} &\longrightarrow \mathbb{F}_{2^n} \\ c &\longmapsto \sum_{i=0}^{\frac{n-1}{2}} c^{2^{2i}}. \end{aligned}$$

- $H(c+d) = H(c) + H(d)$ pour tout $c, d \in \mathbb{F}_{2^n}$
- $H(c)$ est une solution de $x^2 + x = c + \mathtt{Tr}(c)$
- $H(c) = H(c^2) + c + \mathtt{Tr}(c)$

Résolution d'équation de la forme $x^2 + x = c$

- On se place dans le cas où $\mathtt{Tr}(c) = 0$, il suffit alors de calculer $\mathtt{H}(c)$.
- On remarque que

$$\mathtt{H}(c) = \mathtt{H}(\sum_{i=0}^{n-1} c_i X^i) = \sum_{i=0}^{n-1} c_i \mathtt{H}(X^i).$$

- De plus $\mathtt{H}(X^{2i}) = \mathtt{H}(X^i) + X^i + \mathtt{Tr}(X^{2i})$.
- On peut ainsi se permettre de ne stocker que les $\mathtt{H}(X^i)$ pour i impair.

Algorithm 3 `Solution de` $x^2 + x = c$

Require: $c \sum_{i=0}^{n-1} c_i X^i$ avec $\mathtt{Tr}(c) = 0$
Ensure: Une solution s de $x^2 + x = c$

1: Précalculs: $\mathtt{H}(X^i)$ pour $1 \leq i \leq n-2$ impair
2: $s \leftarrow 0$
3: **for** $i = (n-1)/2 \ldots 1$ **do**
4: **if** $c_{2i} = 1$ **then**
5: $c \leftarrow c + X^i, s \leftarrow s + X^i$
6: **end if**
7: **end for**
8: $s \leftarrow s + \sum_{i=1}^{(n-1)/2} c_{2i-1} \mathtt{H}(X^{2i-1})$
9: **return** Q

Racine carrée dans $\mathbb{F}_{2^n}$

- Pour calculer $\sqrt{c}, c \in \mathbb{F}_{2^n}$, on commence par remarquer que $c^{2^n} = c$ i.e. $\sqrt{c} = c^{2^{n-1}}$.
- Par linéarité de l'élévation au carré au obtient:

$$\left(\sum_{i=0}^{n-1} c_i X^i\right)^{2^{n-1}} = \sum_{i=0}^{n-1} c_i (X^{2^{n-1}})^i.$$

- $$\sum_{i \text{ pair}} c_i X^{\frac{i}{2}} + \sqrt{X} \sum_{i \text{ impair}} c_i X^{\frac{i-1}{2}}$$

Opérations dans $\mathbb{F}_{2^n}$

Opération	coût
Opération sur $\mathbb{F}_{2^n}$	
Multiplication	1 M
Inversion	10 M
Élévation au carré	0.1 M
Racine carrée	0.52 M
Résolution de $x^2 + x = c$	0.67 M
Opération sur la courbe	
addition de point	I + 2 M
doublement de point	I + 2 M
division de point	I + 1.19 M

TableCout relatifs des opérations sur $\mathbb{F}_{2^n}$

Courbe de Koblitz

Définition

- Une courbe de Koblitz est une courbe elliptique telle que $a, b \in \mathbb{F}_2$.

- Il existe exactement 2 courbes de Koblitz:

$$E_0 : y^2 + xy = x^3 + 1 \text{ et } E_1 : y^2 + xy = x^3 + x^2 + 1.$$

- Il est possible de remplacer les doublements par une application beaucoup moins couteuse, le morphisme de Frobenius.

Morphisme de Frobenius

Soit $E(\mathbb{F}_{2^n})$ une courbe elliptique le morphisme de Frobenius est l'application suivante:

$$\begin{aligned} \tau : \quad E(\mathbb{F}_{2^n}) &\longrightarrow E(\mathbb{F}_{2^n}) \\ (x, y) &\longmapsto (x^2, y^2) \end{aligned}$$

Propriété

- Soient $a \in \mathbb{F}_2$ et $E_a : y^2 + xy = x^3 + ax^2 + 1$ une courbe de Koblitz. Le morphisme de Frobenius vérifie alors:

$$\forall n \in \mathbb{N}, \forall P \in E_a(\mathbb{F}_{2^n}), \quad \tau^2(P) - [\mu]\tau(P) + [2]P = \mathcal{O},$$

où $\mu = (-1)^{1-a}$

- On peut alors voir τ comme la racine complexe du polynome $X^2 - \mu X + 2$.

Courbes de Koblitz

- On considère maintenant l'anneau $\mathbb{Z}[\tau]$ dont les éléments sont de la forme

$$z = \sum_{i=0}^{l-1} u_i \tau^i$$

- Il est alors possible d'étendre le morphisme multiplication par un scalaire à $\mathbb{Z}[\tau]$ en définissant

$$\forall z \in \mathbb{Z}[\tau], [z]P = \sum_{i=0}^{l-1} [u_i]\tau^i(P).$$

Représentation τ-adique

- Soit $z \in \mathbb{Z}[\tau]$, la division par 2 de k permet d'écrire: $z = u + 2 \times u'$.
- Ainsi on peut écrire:

$$\begin{aligned} z &= z_0 + \tau z_1 \\ &= u_0 + 2 \times u'_0 + \tau z_1 \\ &= u_0 + u'_0 \mu \tau - u'_0 \tau^2 + \tau z_1 \\ &= u_0 + \tau(u'_0 \mu + z_1 - u'_0 \tau) \\ &= u_0 + \tau(z_2 + \tau z_3) \end{aligned}$$

- On peut donc répéter l'opération avec $z_2 + \tau z_3$.

Courbes de Koblitz

- On définit une norme sur $\mathbb{Z}[\tau]$ par:

$$\begin{array}{rccc} N: & \mathbb{Z}[\tau] & \longrightarrow & \mathbb{N} \\ & z_0 + \tau z_1 & \longrightarrow & z_0^2 + \mu z_0 z_1 + 2 z_1^2. \end{array}$$

- On peut vérifier que $N(z_0 + \tau z_1) > N(z_2 + \tau z_3)$, se qui assure que le procédé précédent termine bien.

Algorithm 4 Représentation τNAF

Require: $z = z_0 + z_1\tau \in \mathbb{Z}[\tau]$
Ensure: $z = (r_{l-1} \dots r_0)_{\tau\mathtt{NAF}} = \sum_{i=0}^{l-1} r_i \tau^i$

$i = 0$
while $|z_0| + |z_1| \neq 0$ **do**
  if $z_0 \equiv 1 \mod 2$ **then**
    $r \leftarrow 2 - ((z_0 - 2z_1) \mod 4)$
    $z_0 \leftarrow z_0 - r$
  else $r \leftarrow 0$
  end if
  $r_i \leftarrow r$
  $(z_0, z_1) \leftarrow (z_1 + \mu z_0/2, -z_0/2)$
  $i \leftarrow i + 1$
end while
return $(r_{l-1} \dots r_0)$

Algorithm 5 Multiplication de point τNAF

Require: $P \in E(\mathbb{F}_{2^n}), k = (r_{l-1} \dots 0)_{\tau\text{NAF}}$
Ensure: $[k]P$

1: $Q \leftarrow \mathcal{O}$
2: **for** $i = (l-1) \dots 0$ **do**
3: $\quad \tau(Q)$
4: $\quad$ **if** $r_i = 1$ **then** $Q \leftarrow Q + P$
5: $\quad$ **else if** $r_i = -1$ **then** $Q \leftarrow Q - P$
6: $\quad$ **end if**
7: **end for**
8: **return** Q

Courbes de Koblitz

Performances

- La représentaion τNAF est en général de longueur $2\log(k)$ et de densité $1/3$.
- Il est possible de la ramener à $\log(k)$ termes.
- Cout: $\tau + (\mathtt{I} + 2\mathtt{M})/3$ par bit d'exposant.