N° d'ordre: 07 / PGS - 2004 / MT

Université des sciences et de la technologie Houari Boumediene

Faculté de mathématiques

Mémoire présenté pour l'obtention du diplôme de PGS en

Mathématiques

Spécialité: Cryptologie

Par:

MERABET BRAHIM

Thème

COURBES HYPERELLIPTIQUES ET CRYPTOGRAPHIE

Soutenu le 25 mars 2004 devant le jury composé de :

M. ZITOUNI	Président	professeur .USTHB
k. BETINA	Rapporteur	professeur .USTHB
M.ABID	Examinateur	M.de conférences .USTHB
H. HAMITI	Examinateur	DGSCT.

INTRODUCTION

Les courbes hyperelliptiques sont une classe spéciale de courbes algébriques, on peut les considérer comme étant une généralisation des courbes elliptiques. Il y a des courbes hyperelliptiques de différents genres $g \geq 1$. Une courbe hyperelliptique de genre $g = 1$ est une courbe elliptique. Les courbes elliptiques sont applicables dans des domaines importants tels que la théorie des codes, la génération des nombres pseudo aléatoires, les algorithmes de la théorie des nombres, et la cryptographie. Récemment, on a trouvé des applications des courbes hyperelliptiques en dehors de la géométrie algébrique, comme le test de primalité, la correction d'erreurs des codes, la factorisation d'entiers et la cryptographie à clef publique.

On s'intéresse dans ce travail aux crypto systèmes sur les courbes hyperelliptiques, on commence d'abord par une introduction à la théorie des courbes hyperelliptiques, les diviseurs sur ces courbes, leurs représentations polynomiales et leurs sommations, on passe ensuite à une description d'un crypto système sur le Jacobien d'une courbe hyperelliptique, on explique et on analyse les travaux de quelques chercheurs sur la performance et la possibilité de l'implémentation de ces crypto systèmes, comme l'étude de la complexité des calculs dans le corps où la courbe est définie pour différentes opérations sur les diviseurs, et une comparaison des temps d'exécution d'un algorithme de signature numérique en utilisant des courbes de différents genres, définies sur différent corps. On passe par suite aux attaques possibles de ces crypto systèmes, on distingue trois types d'attaques, on parle aussi de quelques algorithmes d'attaque ainsi l'implémentation de quelques-uns. Dans le dernier chapitre, on donne deux types de courbes convenables à l'utilisation en cryptographie et la méthode de leurs constructions, pour éviter ces attaques.

SOMMAIRE

Chapitre 1 : INTODUCTION ELEMENTAIRE AUX COURBES HYPERELLIPTIQUES

On donnera dans le premier et le deuxième chapitres des notions de base sur les courbes hyperelliptiques, les résultats seront donnés sans démonstrations, le lecteur intéressé pourra consulter [MHZ96] ou [Kob98].

1.1- DEFINITIONS DE BASE :

Dans tout ce qui suit K désigne un corps et $\overline{K}$ sa clôture algébrique,

$K[u]$: l'ensemble des polynômes à une indéterminée u,

$K[u, v]$: l'ensemble des polynômes à deux indéterminées u et v.

Définition 1.1 (*courbe hyperelliptique*) **:**

Une courbe hyperelliptique C de genre g sur K ($g \geq 1$) est l'ensemble des points (u, v) de $K \times K$ vérifiant une équation de la forme :

$$C : v^2 + h(u)v = f(u) \ \ dans \ \ K[u,v] \tag{1.1}$$

où :

$h(u)$ est un polynôme sur K de degré au plus g,

$f(u)$ est un polynôme sur K de degré $2g+1$, et il n'existe pas de solutions (u, v) dans $\overline{K} \times \overline{K}$ satisfaisant simultanément:

$$v^2 + h(u)v = f(u) \tag{1.2}$$

et les équations aux dérivées partielles :

$$2v + h(u) = 0 \tag{1.3}$$

et

$$h'(u)v - f'(u) = 0 \tag{1.4}$$

Un point singulier de C est une solution (u,v) qui satisfait simultanément les trois équations précédentes.

Une courbe hyperelliptique ne contient pas de points singuliers.

Lemme 1.2:

Soit *C* une courbe hyper elliptique sur *K* définie par l'équation (1.1) on a:

(*i*) si $h(u)=0$ alors la caractéristique du corps *K* est non nulle,

(*ii*) si caract $(K)\neq 2$, le changement de variable (u, v) en $(u, v-h(u)/2)$ transforme l'équation de *C* en $v^2=f(u)$, avec $deg^u f=2g+1$.

(*iii*)si *C* a une équation de la forme (1.1) avec h(u)=0 et caract $(K)\neq 2$, alors *C* est une courbe hyper elliptique si et seulement si $f(u)$ n'a pas de racine double dans *K*.

Définition 1.3 : (*points rationnelles, points à l'infini, points finis*)
Soit *L* une extension du corps *K*. L'ensemble des points *L*-rationnels sur *C,* noté $C(L)$ est l'ensemble des points $P=(x,y)$de LxL satisfaisant l'équation (1.1) et le point à l'infini noté ∞. L'ensemble des points $C(\overline{K})$ sera noté *C*. Les points de *C* autre que ∞ sont appelés points finis.

Définition 1.4 : (*opposé, points spécial et ordinaire*)
Soit $P=(x,y)$ un point fini sur *C*. L'opposé de *P* c'est le point $\widetilde{P}=(x,-y-h(x))$, (on remarque que le point $\widetilde{P}$ est aussi dans *C*). L'opposé du point ∞ est égal à lui-même. Si $P=\widetilde{P}$ on dit que *P* est un point spécial, sinon il est appelé point ordinaire.

Exemple 1.5 : (*courbe hyper elliptique sur le corps des réels*)
Les exemples suivants sont des courbes hyper elliptiques sur le corps des nombres réels avec un genre $g=2$ et $h(u)=0$.

1. $C_1 : v^2=u^5+u^4+4u^3+4u^2+3u+3=(u+1)(u^2+1)(u^2+3)$

2. $C_2 : v^2=u^5+u^4-u^2-u=u(u-1)(u+1)(u^2+u+1)$

3. $C_3 : v^2=u^5-5u^3+4u=u(u-1)(u+1)(u-2)(u+2)$

Exemple 1.6 : (*courbe hyper elliptique sur le corps des entiers rationnels modulo 7*)

On considère la courbe C : $v^2+uv=u^5+5u^4+6u^2+u+3$ sur $\boldsymbol{Z}_7$

Ici , $h(u)=u$, $f(u)=u^5+5u^4+6u^2+u+3$ et $g=2$.

On peut vérifier que C n'a pas de points singuliers, et donc c'est une courbe hyperelliptique. Les points $\boldsymbol{Z}_7$ –rationnels sur C sont :

$C(\boldsymbol{Z}_7) = \{\infty,(1,1),(1,5),(2,2),(2,3),(5,3),(5,6),(6,4)\}$

Le point (6,4) est un point spécial.

Exemple 1.7 : (*courbe hyper elliptique sur le corps* $\boldsymbol{F}_2^5$)

On considère le corps fini $\boldsymbol{F}_2^5=\boldsymbol{F}_2[x]/(x^5+x^2+1)$, et soit α une racine du polynôme primitif x^5+x^2+1 dans $\boldsymbol{F}_2^5$, le tableau suivant donne les puissances de α :

n	α^n	n	α^n	n	α^n
0	1	11	$\alpha^2+\alpha+1$	22	$\alpha^4+\alpha^2+1$
1	α	12	$\alpha^3+\alpha^2+\alpha$	23	$\alpha^3+\alpha^2+\alpha+1$
2	α^2	13	$\alpha^4+\alpha^3+\alpha^2$	24	$\alpha^4+\alpha^3+\alpha^2+\alpha$
3	α^3	14	$\alpha^4+\alpha^3+\alpha^2+1$	25	$\alpha^4+\alpha^3+1$
4	α^4	15	$\alpha^4+\alpha^3+\alpha^2+\alpha+1$	26	$\alpha^4+\alpha^2+\alpha+1$
5	α^2+1	16	$\alpha^4+\alpha^3+\alpha+1$	27	$\alpha^3+\alpha+1$
6	$\alpha^3+\alpha$	17	$\alpha^4+\alpha+1$	28	$\alpha^4+\alpha^2+\alpha$
7	$\alpha^4+\alpha^2$	18	$\alpha+1$	29	α^3+1
8	$\alpha^3+\alpha^2+1$	19	$\alpha^2+\alpha$	30	$\alpha^4+\alpha$
9	$\alpha^4+\alpha^3+\alpha$	20	$\alpha^3+\alpha^2$	31	1
10	α^4+1	21	$\alpha^4+\alpha^3$		

Tableau 1 : les puissances de α dans le corps fini $\boldsymbol{F}_2^5=\boldsymbol{F}_2[x]/(x^5+x^2+1)$

Considérons la courbe $C : v^2+(u^2+u)v=u^5+u^3+1$ de genre $g=2$ sur le corps fini $\boldsymbol{F}_2^5$.

Ici, $h(u)=u^2+u$ et $f(u)=u^5+u^3+1$. On peut vérifier que C n'a pas de points singuliers (autre que ∞), et donc C est une courbe hyperelliptique.

Les points $\boldsymbol{F}_2^5$ –rationnels finis sur *C* sont :

$(0,1)$ $(1,1)$ (α^5,α^{15}) (α^5,α^{27}) (α^7,α^4) (α^7,α^{25}) (α^9,α^{27}) (α^9,α^{30})
$(\alpha^{10},\alpha^{23})$ $(\alpha^{10},\alpha^{30})$ (α^{14},α^8) $(\alpha^{14},\alpha^{19})$ $(\alpha^{15},0)$ (α^{15},α^8) $(\alpha^{18},\alpha^{23})$ $(\alpha^{18},\alpha^{29})$
(α^{19},α^2) $(\alpha^{19},\alpha^{28})$ $(\alpha^{20},\alpha^{15})$ $(\alpha^{20},\alpha^{29})$ $(\alpha^{23},0)$ (α^{23},α^4) (α^{25},α) $(\alpha^{25},\alpha^{14})$
$(\alpha^{27},0)$ (α^{27},α^2) (α^{28},α^7) $(\alpha^{28},\alpha^{16})$ $(\alpha^{29},0)$ (α^{29},α) $(\alpha^{30},0)$ $(\alpha^{30},\alpha^{16})$

Les points *(0,1)* et *(1,1)* sont des points spéciaux.

1.2- FONCTIONS POLYNOMIALES ET RATIONNELLES :

On introduit dans cette partie des propriétés de base des fonctions polynomiales et rationnelles vues comme fonctions définies sur une courbe hyperelliptique.

Définition 1.8 :

Soit *G(u,v)* un polynôme dans $\overline{K}[u,v]$, on peut considérer *G(u,v)* comme étant une fonction sur la courbe *C* (en d'autres termes, comme un élément de l'anneau quotient :

$$\overline{K}[C]=\overline{K}[u,v]/(v^2+h(u)v-f(u))$$

avec $(v^2+h(u)v-f(u))$ l'idéal dans *K[u,v]* engendré par le polynôme $v^2+h(u)v-f(u)$.

Un élément de $\overline{K}[C]$ est appelé fonction polynomiale sur *C*.

Remarquons que pour chaque polynôme *G(u,v)* de $\overline{K}[C]$, on peut remplacer chaque occurrence de v^2 par $f(u) - h(u)v$, pour obtenir une représentation de la forme :

$G(u,v)=a(u)-b(u)v$, avec $a(u)$, $b(u)$ sont dans $\overline{K}[u]$.

Il est facile de voir que cette représentation est unique

Définition 1.9 : (*conjugué*)

Soit $G(u,v)=a(u)-b(u)v$ une fonction polynomiale sur $\overline{K}[C]$. Le conjugué de *G (u,v)* est la fonction polynomiale définie par :

$$\overline{G}(u,v)=a(u)+b(u)(h(u)+v)\ .$$

Définition 1.10 : (*norme*)

Soit $G(u,v)=a(u)-b(u)v$ une fonction polynomiale sur $\overline{K}[C]$. La norme de $G(u,v)$ est la fonction polynomiale définie par :

$$N(G)=G\overline{G}$$

La fonction norme sera utile pour les transformations d'une fonction à deux variables en une fonction à une variable.

Lemme 1.11 : (*propriétés de la norme*)

Soient G et H deux fonctions polynomiales de $\overline{K}[C]$, alors :

(i) $N(G)$ est un polynôme dans $\overline{K}[u]$.

(ii) $N(\overline{G})=N(G)$,et

(iii) $N(GH)=N(G)N(H)$

Définition 1.12 : (*corps des fonctions, fonctions rationnelles*)

Le corps des fonctions $\overline{K}(C)$de C sur $\overline{K}$ est le corps des fractions de $\overline{K}[C]$; les éléments de $\overline{K}(C)$ sont appelés fonctions rationnelles sur C.

Remarquons que $\overline{K}[C]$est un sous anneau de $\overline{K}(C)$, i.e : chaque fonction polynomiale est aussi une fonction rationnelle.

Définition 1.13 : (*valeur d'une fonction rationnelle en un point fini*)

Soit R une fonction rationnelle, et soit P un point fini de C, on dit que R est définie au point P s'il existe des fonctions polynomiales G, H dans $\overline{K}[C]$ telles que $R=G/H$ et $H(P)$ non nulle. Si R est définie au point P, la valeur de R en P est $R(P)=G(P)/H(P)$.

Il est facile de voir que la valeur de $R(P)$ est bien définie (elle ne dépend pas du choix de G et H.

La définition suivante exprime la notion de degré d'une fonction polynomiale.

Définition 1.14 : (*degré d'une fonction polynomiale*)

Soit $G(u,v)=a(u)-b(u)v$ une fonction polynomiale dans $\overline{K}[C]$. Le degré de G est défini par :

$$\deg(G)=\max[2\deg_u(a),2g+1+2\deg_u(b)]$$

(g étant le genre de la courbe C).

Lemme 1.15 : (*Propriétés de degré*)

Soit G, H deux fonctions polynomiales de $\overline{K}[C]$

(i) $\deg(G)=\deg_u(N(G))$

(ii) $deg(GH)=deg(G)+deg(H)$

(iii) $\deg(G)=\deg(\overline{G})$

Définition 1.16: (*valeur d'une fonction rationnelle à l'infini)*

Soit $R=G/H$ une fonction rationnelle de $\overline{K}(C)$.

(i) si $\deg(G)<\deg(H)$ on dit que la valeur de R au point ∞ est nulle : $R(\infty)=0$,

(ii) si $\deg(G)>\deg(H)$ on dit que R n'est pas définie au point ∞,

(iii) si $\deg(G)=\deg(H)$ on dit que la valeur de R au point ∞ est définie comme étant le quotient des termes dominants de G et de H, (les termes de plus haut degré).

1.3- ZEROS ET POLES :

Ce paragraphe introduit la notion d'uniformisante, et l'ordre d'une fonction rationnelle aux zéros et aux pôles.

Définition 1.17 : (*zéro, pôle*)

Soit *R* une fonction rationnelle non nulle de $\overline{K}(C)$, et soit *P* un point de la courbe *C*. Si *R(P)=0*, on dit que *R* a un zéro au point *P ;* si *R* n'est pas définie au point *P,* on dit que *R* a un pôle au point *P*, on écrit dans ce cas $R(P)=\infty$.

Lemme 1.18 :

Soit *G* une fonction polynomiale et *P* un point de *C*. Si *G(P)=0* alors $\overline{G}(\widetilde{P})=0$. ($\widetilde{P}$ étant l'opposé de *P*).

Les trois lemmes suivants seront utilisés dans le théorème 1.22 qui prouve l'existence d'uniformisante.

Lemme 1.19 :

Soit *P=(x,y)* un point de *C* . Supposons que *G(u)=a(u)-b(u)v* a un zéro au point *P* et que *x* n'est pas une racine de *a(u)* et de *b(u)*.Alors

$\overline{G}(P)=0$ si et seulement si *P* est un point spécial.

Lemme 1.20 :

Soit *P=(x,y)* un point ordinaire de *C*, et soit *G(u)=a(u)-b(u)v* une fonction polynomiale. Supposons que *G(P)=0* et que *x* n'est pas une racine de *a(u)* et de *b(u)*. Alors *G* peut s'écrire sous la forme :

$G=(u-x)^sS$, où *s* est la plus grande puissance de *(u-x)* qui divise *N(G)*, et *S* une fonction rationnelle n'ayant ni zéro ni pôle au point *P*.

Lemme 1.21 :

Soit *P=(x,y)* un point spécial de *C*. Alors *(u-x)* peut s'écrire sous la forme :

$(u-x)=(v-y)^2S(u,v)$, où *S(u,v)* est une fonction rationnelle n'ayant ni zéro ni pôle au point *P*.

Théorème 1.22 (*existence de l'uniformisante*)

Soit *P* un point de la courbe *C*, alors il existe une fonction rationnelle *U* de $\overline{K}(C)$ vérifiant *U(P)=0* et telle que l'on ait la propriété suivante :
Pour chaque fonction rationnelle non nulle *G*, il existe un entier *d* et une fonction rationnelle *S* n'ayant ni zéro ni pôle en *P* tels que :
$G=U^dS;$ de plus l'entier *d* ne dépend pas du choix de la fonction *U*.
La fonction *U* est appelée uniformisante au point *P*.

La notion d'uniformisante sera utilisée pour définir l'ordre d'une fonction rationnelle en un point *P*.

Définition 1.23 : (*définition usuelle de l'ordre d'une fonction rationnelle en un point*)
Soit *G* une fonction polynomiale non nulle et *P* un point de la courbe *C*. Soit $U \in \overline{K}(C)$ une uniformisante au point *P*, ($G=U^dS$), $S(P)\neq 0,\infty$. L'ordre de *G* au point *P* est défini par $ord_P(G)=d$.

Lemme 1.24 :
Soient G_1, G_2 deux fonctions polynomiales et *P* un point de la courbe *C*, et soit $ord_p(G_1)=r_1$ et $ord_p(G_2)=r_2$, alors :

(i) $ord_p(G_1G_2)=ord_P(G_1)+ ord_p(G_2)$,

(ii) supposons que $G_1\neq -G_2$:
si $r_1\neq r_2$ alors $ord(G_1+G_2)=min(r_1,r_2)$
si $r_1=r_2$ alors $ord(G_1+G_2)\geq \min(r_1,r_2)$.

Définition 1.25 : (*définition alternative de l'ordre d'une fonction polynomiale en un point)*
Soient $G=a(u)-b(u)v$ une fonction polynomiale et *P* un point de *C*, l'ordre de *G* au point *P,* noté $ord_p(G)$, est défini comme suit :

(i) Si *P=(x,y)* est un point fini, soit *r* la plus grande puissance de *(u-x)* qui divise *a(u)* et *b(u)*, et on écrit :

$$G(u,v)=(u-x)^r(a_0(u)-b_0(u)v)$$

si $a_0(x)-b_0(x)y\neq 0$, on pose $s=0$, sinon soit s la plus grande puissance de $(u-x)$ qui divise $N(a_0(u)-b_0(u)v)= a_0^2+a_0b_0h-b_0^2f$.

Si P est un point ordinaire alors on pose $ord_p(G)=r+s$, si P est un point spécial alors on pose $ord_p(G)=2r+s$.

(ii) Si $P=\infty$ alors

$$ord_p(G)=-\max\left[2\deg_u(a),2g+1+2\deg_u(b)\right]$$

Lemme 1.26 :

Les définitions 1.23 et 1.25 sont équivalentes.

Le lemme suivant est une généralisation du lemme 1.18

Lemme 1.27 :

Soient G une fonction polynomiale et un point de C, alors :

$$ord_P(G)=ord_{\bar{P}}(\overline{G})$$

Théorème 1.28 :

Soit G une fonction polynomiale non nulle, alors G a un nombre fini de zéros et de pôles ; en plus :

$$\sum_{P\in C} ord_P(G)=0$$

Définition 1.29 : (*ordre d'une fonction rationnelle en un point*)

Soit $R=G/R$ une fonction rationnelle sur et P un point de la courbe C, l'ordre de R au point P est défini par :

$$ord_P(R)=ord_P(G)-ord_P(H)$$

Il facile de vérifier que $ord_P(R)$ ne dépend pas du choix des fonctions G et H, et que le lemme 1.24 et le théorème 1.28 restent vrais pour une fonction rationnelle non nulle.

Chapitre 2 : *DIVISEURS SUR UNE COURBE HYPERELLPTIQUE*

2.1- DIVISEURS :

Dans ce chapitre on présente des propriétés de base des diviseurs et on introduit la notion du Jacobien d'une courbe hyperelliptique.

Définition 2.1 : (*diviseur, degré, ordre*)

Un diviseur est une somme formelle des points de la courbe C :

$$D=\sum_{P\in C} m_P P\ ,\ m_P\in \boldsymbol{Z}$$

avec seulement un nombre fini de nombres m_P non nulles.

Le degré de D noté $degD$ est l'entier $\sum_{P\in C} m_P$;

L'ordre de D au point P est l'entier m_P, on écrit : $ord_P(D)= m_P$.

L'ensemble des diviseurs noté $\boldsymbol{D}$, forme un groupe additif grâce à l'addition définie par:

$$\sum_{P\in C} m_P P+\sum_{P\in C} n_P P=\sum_{P\in C}(m_P+n_P)P\ .$$

L'ensemble des diviseurs de degré 0, noté $\boldsymbol{D}^0$, est un sous-groupe de $\boldsymbol{D}$.

Définition 2.2 : (*pgcd de diviseurs*)

Soient $D_1=\sum_{P\in C} m_P P\ ,\ D_2=\sum_{P\in C} n_P P$ deux diviseurs ; le plus grand diviseur commun de D_1 et de D_2 est défini par:

$$p\gcd(D_1,D_2)=\sum_{P\in C}\min(m_P,n_P)P-\left(\sum_{P\in C}\min(m_P,n_P)\right)\infty$$

(notons que $pgcd(D_1,D_2) \in \boldsymbol{D}^0$).

Définition 2.3 : (*diviseurs d'une fonction rationnelle*)

Soit R une fonction rationnelle non nulle, le diviseur de R est défini par :

$$div(R)=\sum_{P\in C} ord_P(R)P \ .$$

Notons que si $R=G/H$ alors :

$$div(R)=div(G)-div(H)$$

Le théorème 1.28 implique que le diviseur d'une fonction rationnelle est aussi une somme formelle fini et il est de degré 0.

Exemple 2.4 :

- Si $P=(x,y)$ est un point ordinaire de C alors $div(u-x)=P+\widetilde{P}-2\infty$,
- Si $P=(x,y)$ est un point spécial de C alors $div(u-x)=2P-2\infty$.
- soient la courbe $C=\boldsymbol{R}$ (l'ensemble des nombres réel), et $R(x)=G(x)/H(x)$ une fonction sur C, avec : $G(x)=(x-1)(x+2)^2$ et $H(x)=(x+1)(x-3)^4$, alors :

$div(R)=2(-2,0)-(-1,0)+(1,0)-4(3,0)$.

Lemme 2.5 :

Soit G une fonction polynomiale et soit $div(G)=\sum_{P\in C} m_P P$, alors

$div(\overline{G})=\sum_{P\in C} m_P \widetilde{P}$.

Définition 2.6 : (*diviseur principal, Jacobien*)

Un diviseur $D\in \boldsymbol{D}^0$ est appelé un diviseur principal s'il est le diviseur d'une fonction rationnelle non nulle R.

L'ensemble des diviseurs principaux noté $\boldsymbol{P}$ est un sous-groupe de $\boldsymbol{D}^0$.

Le groupe quotient $\boldsymbol{J}= \boldsymbol{D}^0/\boldsymbol{P}$ est le jacobien de la courbe C.

Si $D_1,D_2 \in \boldsymbol{D}^0$ on écrit $D_1 \sim D_2$ si $D_1 - D_2 \in \boldsymbol{P}$, on dit dans ce cas que D_1 et D_2 sont équivalents.

Définition 2.7 : (*support d'un diviseur*)

Soit $D=\sum_{P\in C} m_P P$ un diviseur, le support de *D* est l'ensemble :

$supp(D)=\{P\in C\ ;\ m_P\neq 0\}$.

Définition 2.8 :(*diviseurs semi-réduit*)

Un diviseur semi-réduit est un diviseur de la forme $D=\sum m_i P_i-\left(\sum m_i\right)\infty$, où $m_i\geq 0$ pour tout *i,* les P_i sont des points finis telles que, si $P_i\in supp(D)$ alors $\tilde{P}_i\notin supp(D)$,sauf si $P_i=\tilde{P}_i$ dans ce cas $m_i=1$.

Lemme 2.9 :

Pour tout diviseur $D\in \boldsymbol{D}^0$ il existe un diviseur semi-réduit D_1 ($D_1\in \boldsymbol{D}^0$) tel que $D\sim D_1$.

2.2- REPRESENTATION DES DIVISEURS SEMI-REDUITS:

Il n'est pas facile de travailler avec les diviseurs au point de vue de l'implémentation, pour cette raison on va introduire dans cette section une représentation polynomiale des diviseurs semi-réduits.

Théorème 2.10 :

Soit $D=\sum m_i P_i-\left(\sum m_i\right)\infty$ un diviseur semi-réduit, où $P_i=(x_i,y_i)$. Soit $a(u)=\prod(u-x_i)^{m_i}$ alors, il existe un unique polynôme *b(u)* satisfaisant :

1) $deg_u b<deg_u a$,
2) $b(x_i)=y_i$ pour tout *i* tels que $m_i\neq 0$,
3) *a(u)* divise $(b(u)^2+b(u)h(u)-f(u))$,
   et on a $D=pgcd(diva(u),div(b(u)-v))$.

Notation :

Le diviseur $D=\sum m_i P_i-\left(\sum m_i P_i\right)\infty$ représenté par la pair de polynômes *(a(u), b(u))*

Sera noté *div(a,b)*.

Le diviseur nul noté par *div(1,0).*

Lemme 2.11 :

Soient *a(u)* et *b(u)* deux fonctions polynomiales sur *C* telles que $deg_u b < deg_u a$.

Si *a* divise (b^2+bh-f) alors *div(a,b)* est semi-réduit.

2.3- DIVISEURS REDUITS :

On définira dans cette section la notion des diviseurs réduits, et on verra que chaque classe du groupe quotient ***J*** a exactement un diviseur réduit.

On peut donc identifier chaque classe avec son diviseur réduit.

Définition 2.12 :(*diviseur réduit*)

Soit $D=\sum m_i P_i-\left(\sum m_i\right)\infty$ un diviseur semi-réduit, si $\sum m_i \le g$ (g étant le genre de C) alors D est appelé diviseur réduit.

Définition 2.13 :(*norme d'un diviseur*)

Soit $D=\sum_{P\in C} m_P P$ un diviseur ; la norme de D, notée $|D|$, est définie par :

$$|D|=\sum_{P\in C-\{\infty\}}|m_P| .$$

Notons que pour un diviseur $D \in \boldsymbol{D}^0$, on peut construire un diviseur semi-réduit D_1 tel que $D \sim D_1$ et $|D_1| \le |D|$.(voir pour cela la preuve du lemme 2.9 dans [MHZ96]).

Lemme 2.14 :

Soit R une fonction rationnelle non nulle. Si R n'a pas de pôles finis, alors R est une fonction polynomiale.

Théorème 2.15 :

Pour tout diviseur $D \in \boldsymbol{D}^0$ il existe un unique diviseur réduit D_1 tel que $D \sim D_1$.

2.4- SOMMATION DES DIVISEURS REDUITS:

Soit *C* une courbe hyperelliptique de genre *g* sur un corps fini *K*, et soit ***J*** le jacobien de *C*. Soit *P=(x,y)* un point de *C,* et soit σ un automorphisme de *K* dans $\overline{K}$, alors $P^{\sigma}=(x^{\sigma},y^{\sigma})=(\sigma(x),\sigma(y))$ est aussi un point de *C*.

Définition 2.16 : (*corps de définition d'un diviseur*)
On dit qu'un diviseur $D=\sum m_P P$ est défini sur *K* si $D^{\sigma}=\sum m_P P^{\sigma}$ est égale à *D* pour tout automorphisme σ de *K* dans $\overline{K}$.

Remarquons que si *D* est défini sur *K*, cela ne veut pas dire que chaque point du support de *D* est *K*-ratinnel. Un diviseur principal est défini sur *K* si et seulement si il est le diviseur d'une fonction rationnelle à coefficients dans *K*.
L'ensemble ***J(K)*** des classes des diviseurs dans ***J*** ayant un représentant défini sur *K* est un sous-groupe de ***J***.
Chaque élément de ***J(K)*** a une représentation unique comme un diviseur réduit *div(a,b)*,où *a* et *b* sont dans *K[u],* $deg_u a \leq g$, $deg_u b < deg_u a$, et donc ***J(K)*** est en fait un groupe abélien fini.

Cette partie présente un algorithme pour sommer des éléments dans ce groupe, cette opération comprend deux étapes :
-la composition de deux diviseurs (algorithme 1)
-la réduction du diviseur obtenu (algorithme 2)

Soient $D_1=div(a_1,b_1)$ et $D_2=div(a_2,b_2)$ deux diviseurs réduits définis sur *K*, (a_1, a_2 b_1,b_2 sont dans *K[u])*. L'algorithme 1 permet de trouver un diviseur semi-réduit $D=div(a,b)$ avec *a* et *b* dans *K[u]* tel que $D \sim D_1+D_2$. L'algorithme 2 permet de réduire le diviseur *D* à un diviseur D'.

Algorithme 1 :

Entré : diviseurs réduits $D_1=div(a_1,b_1)$ et $D_2=div(a_2,b_2)$ définis sur K .

Sortie : un diviseur semi-réduit $D=div(a,b)$ défini sur K tel que $D \sim D_1+D_2$.

1. Utiliser l'algorithme d'Euclide étendu pour trouver d_1, e_1, e_2 dans $K[u]$ vérifiant:

$$d_1=pgcd(a_1,a_2) \text{ et } d_1=e_1a_1+e_2a_2.$$

2. Utiliser l'algorithme d'Euclide étendu pour trouver d, c_1, c_2 dans $K[u]$ vérifiant:

$$d=pgcd(d_1,b_1+b_2+h) \text{ et } d=c_1 d_1+c_2 (b_1+b_2+h).$$

3. Soit $s_1=c_1e_1$, $s_2=c_1e_2$, et $s_3=c_2$, et donc :

$$d=s_1a_1+s_2a_2+s_3((b_1+b_2+h) \qquad (2.1)$$

4. Posons:

$$a=a_1a_2/d^2 \qquad (2.2)$$

et

$$b=\frac{s_1a_1b_2+s_2a_2b_1+s_3(b_1b_2+f)}{d} \ mod\ a \qquad (2.3)$$

Remarque:

L'opération serait plus facile si on veut doubler un diviseur (autrement dit si on veut calculer $2D$), on pose donc $a=a_1=a_2$ et $b=b_1=b_2$ dans l'algorithme 1.

L'intérêt de doubler un diviseur est de calculer efficacement $mD=D+D+\ldots+D$ (m fois). Cette opération s'appelle *la multiplication scalaire,* elle serait utile en cryptographie.

Théorème 2.17 :

Soient $D_1=div(a_1,b_1)$ et $D_2=div(a_2,b_2)$ deux diviseurs semi-réduits. Si a et b sont définis comme dans (2.2) et (2.3), alors $D=div(a,b)$ est un diviseur semi-réduit et $D \sim D_1+D_2$.

Exemple 2.18 :(*sommation de deux diviseurs*)

Considérons la courbe hyper elliptique $C : v^2+(u^2+u)v=u^5+u^3+1$ de genre $g=2$ sur le corps fini $\boldsymbol{F}_2^5$ (voir exemple 1.7).

$P=(\alpha^{30},0)$ est un point ordinaire dans $C(\boldsymbol{F}_2^5)$ et l'opposé de P est $\tilde{P}=(\alpha^{30},\alpha^{16})$.

$Q_1=(0,1)$ et $Q_2=(1,1)$ sont des points spéciaux dans $C(\boldsymbol{F}_2^5)$.

Les exemples suivants sont des exemples de calcul du diviseur semi-réduit $D=div(a,b)=D_1+D_2$, pour des diviseurs réduits D_1 et D_2 (voir l'algorithme 1).

(i) Soient $D_1=P+Q_1-2\infty$ et $D_2=\tilde{P}+Q_2-2\infty$ deux diviseurs réduits ; alors

$D_1=div(a_1,b_1)$ où $a_1=u(u+\alpha^{30}), b_1=\alpha u+1$ et $D_2=div(a_2,b_2)$ où $a_2=(u+1)(u+\alpha^{30})$, $b_2=\alpha^{23}u+\alpha^{12}$.

1. $d_1=pgcd(a_1,a_2)=u+\alpha^{30}; d_1=a_1+a_2.$
2. $d=pgcd(d_1,b_1+b_2+h)=u+\alpha^{30}; d=1.d_1+0.(b_1+b_2+h).$
3. $d=a_1+a_2+0.(b_1+b_2+h).$
4. Posons $a=a_1a_2/d^2=u(u+1)=u^2+u$ et

$$b=\frac{1.a_1b_2+1.a_2b_1+0.(b_1b_2+f)}{d} \bmod a$$

$$\equiv 1 \pmod a$$

donc :

$$div(a)=2Q_1+2Q_2-4\infty$$

$$div(b-v)=Q_1+Q_2+\sum_{i=1}^{3}P_i-5\infty \text{ où } P_i\neq Q_1,Q_2$$

$$div(a,b)=Q_1+Q_2-2\infty$$

(ii) Soient $D_1=P+Q_1-2\infty$ et $D_2=Q_1+Q_2-2\infty$; alors

$D_1=div(a_1,b_1)$ où $a_1=u(u+\alpha^{30}), b_1=\alpha u+1$ et $D_2=div(a_2,b_2)$ où $a_2=u(u+1)$, $b_2=1$.

1. $d_1=pgcd(a_1,a_2)=u; d_1=\alpha^{14}a_1+\alpha^{14}a_2.$
2. $d=pgcd(d_1,b_1+b_2+h)=u; d=1.u+0.(b_1+b_2+h).$
3. $d=\alpha^{14}a_1+\alpha^{14}a_2+0.(b_1+b_2+h)$
4. $a=(u+\alpha^{30})(u+1); b\equiv \alpha^{14}u+\alpha^{13} (mod\ a)$.

donc :

$$div(a)=2Q_2+P+\tilde{P}-4\infty$$

$$div(b-v)=P+Q_2+\sum_{i=1}^{3}P_i-5\infty \text{ où } P_i\neq P,\tilde{P},Q_2 .$$

$$div(a,b)=P+Q_2-2\infty .$$

(iii) Soient $D_1=P+Q_1-2\infty$ et $D_2=P+Q_2-2\infty$; alors

$D_1=div(a_1,b_1)$ où $a_1=u(u+\alpha^{30}), b_1=\alpha u+1$ et $D_2=div(a_2,b_2)$ où,

$a_2=(u+1)(u+\alpha^{30})$, $b_2=\alpha^{14}u+\alpha^{13}$.

1. $d_1=pgcd(a_1,a_2)=(u+\alpha^{30}); d_1=1.a_1+1.a_2.$
2. $d=pgcd(d_1,b_1+b_2+h)=1.$
3. $d=(\alpha^{15}u+\alpha^{4})a_1+(\alpha^{15}u+\alpha^{4})a_2+\alpha^{15}.(b_1+b_2+h)$
4. $a=u(u+1)(u+\alpha^{30})^2; b\equiv \alpha^{17}u^3+\alpha^2u+1 (mod\ a)$.

donc :

$$div(a)=2P+2\tilde{P}+2Q_1-8\infty$$

$$div(b-v)=2P+Q_1+Q_2+\sum_{i=1}^{2}P_i-6\infty \text{ où } P_i\neq P,\tilde{P},Q_1,Q_2 .$$

$$div(a,b)=2P+Q_1+Q_2-4\infty .$$

Algorithme 2 :

Entré : un diviseur semi-réduit $D=div(a,b)$ défini sur K.

Sortie : l'unique diviseur réduit $D'=div(a',b')$ tel que $D' \sim D$.

1.Posons

$$a' = (f - bh - b_2)/a \tag{2.4}$$

et

$$b' = (-h - b) \bmod a' \tag{2.5}$$

2. si $deg_u a' > g$ alors poser $a = a'$ et aller à l'étape 1.

3. soit c le coefficient de a', posons $a' = c^{-1}a'$

4.sortie (a', b').

Théorème 2.19 :

Soit $D = div(a,b)$ un diviseur semi-réduit; alors le diviseur $D' = div(a', b')$ défini par l'algorithme 2 est réduit et $D' \sim D$.

Notons que tous les calculs dans les algorithmes 1 et 2 se font dans le corps K lui-même (et pas dans une extension de K). Dans l'algorithme 1, si $deg_u a_1 \leq g$ et $deg_u a_2 \leq g$, alors $deg_u a \leq 2g$. Dans ce cas l'algorithme 2 comprend au plus $\lceil g/2 \rceil + 1$ itérations pour l'étape 1.

Exemple 2.20 :

Considérons la courbe hyper elliptique C : $v^2 + (u^2 + u)v = u^5 + u^3 + 1$ de genre $g = 2$ (voir l'exemple 1.7). Considérons le diviseur semi-réduit $D = (0,1) + (1,1) + (\alpha^5, \alpha^{15}) - 3\infty$. Alors $D = div(a,b)$, avec:

$$a(u) = u(u+1)(u+\alpha^5) = u^3 + \alpha^2 u^2 + \alpha^5 u$$

et

$$b(u) = \alpha^{17} u^2 + \alpha^{17} u + 1$$

L'algorithme 2 nous donne :

$$a'(u) = u^2 + \alpha^{15} u + \alpha^{26},$$

$$b'(u) = \alpha^{23} u + \alpha^{21}.$$

Ainsi $D \sim div(a',b') = (\alpha^{28},\alpha^{7}) + (\alpha^{29},0) - 2\infty$.

2.5- ANALOGIE : (avec l'addition dans le groupe quotient ***Z/pZ***)

Pour mieux comprendre la sommation des diviseurs, on présente une analogie avec la sommation dans un groupe plus connu, c'est le groupe ***Z/pZ***, il est aussi un groupe quotient comme le Jacobien. Prenons par exemple le groupe $\boldsymbol{Z_7}$, on présente les éléments de ce groupe par les classes d'équivalence. Soient $\overline{0},\overline{1},\overline{2},\overline{3},\overline{4},\overline{5},\overline{6}$ les représentants uniques des classes ; ainsi $\overline{0}=\{...,-14,-7,0,7,14,...\}, \overline{1}=\{...,-13,-6,1,8,...\},\ldots, \overline{6}=\{...,-8,-1,6,13,20,...\}$.

Le diviseur principal dans le Jacobien est analogue aux représentants qu'on a choisi. Pour sommer deux éléments de $\boldsymbol{Z_7}$, on passe par deux étapes : la somme de deux éléments et la réduction du résultat modulo *7*, on a de même deux étapes pour l'addition dans le Jacobien. Par exemple pour sommer 3+5, le résultat de cette addition est 8, on réduit par suite *8 = 1 mod 7*, on obtient ainsi la classe d'équivalence $\overline{1}$.

2.6- COMPTAGE DES POINTS D'UNE COURBE HYPERELLIPTIQUE :

2.6.1- La fonction Zeta :

La fonction Zeta d'une courbe hyperelliptique est un outil de base dans les calculs.

Soit *J(C)* le Jacobien d'une courbe hyperelliptique *C* défini sur un corps $\boldsymbol{F}_q$, et soit $\boldsymbol{F}_q^r$ une extension de degré *r* de $\boldsymbol{F}_q$, on note N_r l'ordre du groupe fini abélien *J(C)* et M_r le nombre de points ($\boldsymbol{F}_q^r$ –points) dans *C*, ($M_r = \#C(F_{q^r}), r \geq 1$).

Définition 2.21:

La fonction Zeta de la courbe *C* est la série :

$$Z_C(T) = exp\left(\sum_{r\geq 1} M_r T^r / r\right) \qquad (2.6)$$

où : *exp* désigne la fonction exponentielle.

Théorème 2.22 :(*propriétés de la fonction Zeta*)

Soit C une courbe hyper elliptique de genre g défini sur F_q, et soit $Z_C(T)$ la fonction Zeta de C .

1) $Z_C(T)$ est une fonction rationnelle de la forme :

$$Z_C(T)=\frac{P(T)}{(1-T)(1-qT)} \tag{2.7}$$

où : $P(T)$ est un polynôme de degré $2g$ à coefficients entiers de la forme :

$$P(T)=1+a_1T+...+a_{g-2}T^{g-2}+a_{g-1}T^{g-1}+a_gT^g+qa_{g-1}T^{g+1}+q^2a_{g-2}T^{g+2}+...+q^{g-1}a_1T^{2g-1}+q^gT^{2g}.$$

2) $P(T)$ se factorise comme suit:

$$P(T)=\prod_{i=1}^{g}(1-\alpha_iT)(1-\overline{\alpha}_iT), \tag{2.8}$$

où chaque α_i est un nombre complexe de module $\sqrt{q}$, et $\overline{\alpha}_i$ est le conjugué complexe de α_i .

3) $N_r=\#J(F_{q^r})$ est donné par :

$$N_r=\prod_{i=1}^{g}\left|1-\alpha_i^r\right|^2 \tag{2.9}$$

où : | | dénote le module d'un nombre complexe. En particulier, $N_1=P(1)$. On déduit du théorème 2.22 que pour calculer N_r, pour r arbitraire, on aura besoin de :

(i) déterminer les coefficients $a_1,a_2,\ldots,a_g$ de $P(T)$, donc de déterminer $P(T)$;

(ii) factoriser $P(T)$ et déterminer ainsi α_i ;

(iii) calculer N_r par l'équation (2.9)

notons qu'en multipliant les deux membres de l'équation (2.7) par *(1-T)(1-qT)*, on obtient

$$P(T)=(1-T)(1-qT)Z_C(T)$$

Prenons le logarithme des deux membres, utilisons (2.6), et dérivons par rapport à *T* pour obtenir :

$$\frac{P'(T)}{P(T)}=\sum_{r\geq 0}(M_{r+1}-1-q^{r+1})T^r$$

En utilisant cette dernière équation on voit que les *g* premières valeurs M_1, M_2, …, M_g seraient suffisantes pour déterminer les coefficients $a_1,a_2,\ldots,a_g$ et ainsi N_r pour tout *r*.

La procédure suivante détermine N_r dans le cas $g=2$:

1) calculer M_1 et M_2 par la recherche exhaustive.
2) Les coefficients de $Z_C(T)$ sont donnés par :

$$a_1=M_1-1-q \text{ et } a_2=(M_2-1-q^2+a_1^2)/2$$

3) résoudre l'équation quadratique pour obtenir deux solutions η_1 *et* η_2.
4) résoudre l'équation $X^2+\eta_1 X+q=0$ pour une solution α_1, et résoudre $X^2+\eta_2 X+q=0$ pour une solution α_2.
5) alors $N_r=\left|1-\alpha_1^r\right|^2.\left|1-\alpha_2^r\right|^2$.

De théorème 2.22 on déduit le corollaire suivant.

Corollaire 2.23 :

Soit *C* une courbe hyper elliptique de genre *g* défini sur F_q, et soit $N_r=\#J(F_{q^r})$.

Alors :

$$(q^{r/2}-1)^{2g} \leq N_r \leq (q^{r/2}+1)^{2g}.$$

Ainsi, $N_r \approx q^{rg}$.

Chapitre 3 : CRYPTO SYSTEMES SUR LES COURBES HYPERELLIPTIQUES

Les crypto systèmes utilisant les courbes hyperelliptiques sont des crypto systèmes à clef publique, pour illustrer ces crypto systèmes on aura besoin de quelques notions de base de la cryptographie à clef publique.

3.1 Cryptographie à clef publique :

Avant la fin des années soixante dix, tous les messages cryptographiques utilisés étaient à clef secrète (cryptographie à clef secrète), les systèmes utilisant ces protocoles cryptographiques sont appelés *systèmes symétriques*. L'algorithme de chiffrement et de déchiffrement pour ces systèmes sont identiques, donc si une personne a une information sur la façon de chiffrer un message, il aura automatiquement une certaine information sur le déchiffrement de ce message. Par conséquent, si deux personnes veulent communiquer, ils doivent se rencontrer.

En 1976, Diffie et Hellman ont trouvé une solution pour cette difficulté par la découverte d'un nouveau protocole de communication secrète, c'est la *cryptographie à clef publique.* L'idée est d'utiliser une fonction à sens unique (avec trappe).

3.1.1 Fonctions à sens unique :

Définition 3.1 : (*fonction à sens unique avec trappe*)

Une fonction à sens unique avec trappe présente la propriété d'être facile à calculer, et difficile à inverser si l'on ne dispose pas d'information supplémentaire (secrète). C'est cette information supplémentaire qui constitue la trappe du système.

Le terme ‘‘difficile’’ signifie qu’aucun algorithme connu d’inversion de la fonction n’est assez rapide et économique pour valoir la peine d’être appliqué. Un exemple d’une fonction à sens unique c’est le *problème du logarithme discret* **DLP**.

3.1.2 Problème du logarithme discret :

Définition 3.2 : (*problème du logarithme discret*)

Soit G un groupe commutatif, le *problème du logarithme discret* dans le groupe cyclique $G = \langle g \rangle$ est le suivant : étant donné y dans G, trouver un entier x tel que $g^x = y$ ($xg = y$ si l’opération dans le groupe G est additif).

Remarquons que pour un x donné, on peut facilement calculer g^x, mais que le problème inverse, trouver x connaissant g et y (c’est à dire trouver le logarithme discret de x dans la base g, $log_g x$) est apparemment très difficile.

Définition 3.3 : (**ECDLP**)

Soit E une courbe elliptique sur un corps K et Q un point de E, le problème du logarithme discret dans E de base Q est le problème suivant : étant donné un point P de E, trouver un entier x tel que $P = xQ$ si un tel x existe.

Définition 3.4 : (**HCDLP**)

Soit $J_C(K)$ le Jacobien d’une courbe hyperelliptique C sur un corps K et D_1 un diviseur réduit, le problème du logarithme discret dans $J_C(K)$ de base D_1 c’est le problème, étant donné un diviseur D_2 de E, trouver un entier x tel que $D_2 = xD_1$ si un tel x existe.

3.1.3 Protocole d’échange de clefs de Diffie – Hellman :

Diffie – Hellman (1976) ont décrit un système d’échange de clefs secrètes dont la sécurité repose sur la difficulté du problème du logarithme discret **DLP.**

Supposons que deux personnes Alice et Bob veulent communiquer secrètement, ils peuvent échanger leurs clefs dans un canal public pour avoir une clef secrète commune, de la façon suivante :

S'étant fixé un nombre premier *p* et un générateur du groupe multiplicatif F_p^*

Alice et Bob choisissent k_A et k_B, entiers compris entre *1* et *p-2*, Alice calcule $x(A)=g^{k_A}$, et Bob calcule $x(B)=g^{k_B}$. Puis *x(A)* et *x(B)* sont échangés (publiés dans un annuaire), alors que k_A et k_B sont gardés secrets. Alors $k_{AB}=g^{k_A k_B}$ constitue une clef secrète de Alice et Bob, en effet :

Alice peut calculer

$$(x(B))^{k_A}=\left(g^{k_B}\right)^{k_A}=g^{k_B k_A}=k_{AB}$$

et Bob peut faire de même. Un intercepteur ne peut calculer k_{AB} sans connaître k_A ou k_B , (il faut trouver un logarithme discret).

Remarque 3.5 :

Un expéditeur A peut s'assurer que personne ne peut envoyer un message à son ami B (le récepteur), et l'expéditeur B à son tour peut s'assurer que le message reçu est à l'origine de son ami A.Ils utilisent pour cet effet le schéma de la signature numérique **DSA**, c'est un algorithme de chiffrement à clef publique.

Remarque 3.6 :

On peut signer puis chiffrer un message, ou bien on chiffre d'abord puis on signe ce message.

Il existe un protocole de signature numérique sur les courbes elliptiques **ECDSA** et sur les courbes hyperelliptique **HCDSA**, consulter [Kob98] pour ces deux protocoles.

Les deux diagrammes suivants résument les deux systèmes de chiffrement, chiffrement sans et avec signature, où *f* désigne la fonction de chiffrement d'inverse *g*.

Utilisateur quelconque	B
	Désire recevoir des messages confidentiels -Choisit K et K' -Publie K
Désire envoyer un message confidentiel M à B -Prend connaissance de K -Chiffre M en calculant $C=f_K(M)$ -Envoie *C* à B	
	-Reçoit C -Déchiffre C en calculant $g_{K'}(C)=M$

Tableau 3.1 : chiffrement à clef publique

A	B
Désir signer des messages confidentiels	Désire recevoir des messages confidentiels dont il puisse vérifier la signature
Choisit K(A), K'(A)	Choisit K(B), K'(B)
Publie K(A)	Publie K(B)
Désire envoyer un message confidentiel et signé M à B -signe M en calculant $M_s=g_{K'(A)}(M)$ -prend connaissance de *K(B)* -chiffre M_s en calculant $C=f_{K(B)}(M_s)$ -envoie *C* à *B*	
	-Reçoit *C* -Déchiffre *C* en calculant $g_{K'(B)}(C)=M_s$ -désire vérifier la signature de *A* -prend connaissance de *K(A)* -vérifie la signature en calculant $f_{K(A)}(M_s)=M$

Tableau 3.2 : chiffrement et signature à clef publique.

3.2 crypto systèmes sur les courbes hyperelliptiques HECC :

3.2.1 Historique :

Les crypto systèmes sur les courbes elliptiques **ECC** (se sont des courbes hyperelliptique de genre 1) sont proposé en 1985 indépendamment par Victor Miller et par Neal Koblitz, ces crypto systèmes ont les avantages suivants :

1-Une meilleur flexibilité du choix de groupe, (pour chaque nombre $q=p^n$, p premier, il y a un seul groupe multiplicatif F_q^*, mais on peut définir plusieurs courbes elliptiques $E(F_q)$ sur F_q, et spécialement :

2- L'absence d'algorithmes sous-exponentielle pour casser le système (trouver une solution du ECDLP) si *E* est convenablement choisie.

Quelques années après l'invention des crypto systèmes sur les courbes elliptiques ECC, Menezes, Okamoto, et Vanstone [MOV 93] trouvaient une réduction du problème du logarithme discret sur les courbes elliptiques à un problème du logarithme discret dans une extension F_{q^k}. Pour que cette réduction soit faisable il faut que *k* soit petit, dans la caractéristique 2 du corps, les courbes elliptiques pour lesquelles *k* est petit sont super singulières (voir Koblitz [Kob98] page 125 pour la définition de ces courbes), mais la grande majorité des courbes elliptiques sont non super singulières, pour ces dernières la réduction de Menezes, Okamoto, et Vanstone n'est jamais sous exponentielle (voir Balasubramanian et Koblitz [BK98]).

3.2.2 Description des crypto systèmes sur les courbes elliptiques ECC et sur les courbes hyperelliptiques HECC:

Ils utilisent une idée d'El Gamal [ElG85], soit *E* une courbe elliptique définie sur un corps fini *F,* et soit *Q* un point de *E* d'ordre *h*. Le destinataire Bob choisit un entier secret k_B dans [*0, h-1*] et diffuse la valeur du point $B=k_BQ$.
L'expéditeur Alice peut alors chiffrer un message $M = (x_M, y_M)$ de $F_q \times F_q$ de la manière suivante : (noter que le point *M* peut ne pas appartenir à *E*)
Elle choisit un entier secret k_A dans [*0, h-1*] et elle calcule $G= k_AQ$, elle transmet à Bob ensuite le pair de points $(G, C) = (k_A Q, M+ k_A (k_B Q))$.
A la réception Alice peut retrouver le message clair *M* en multipliant la première composante de la paire de points *(G, C)* par sa clef privée k_B et retrancher le résultat obtenu du deuxième composante.
Remarquons que bien que *G* et *B* soient connus, un intercepteur ne peut calculer ni k_BG ni k_AB (pour retrouver *M*), car il est quasiment impossible de connaître k_A et k_B grâce à la difficulté du problème du logarithme discret (ECDLP) dans *E*. Ce protocole se généralise au cas des courbes hyperelliptiques en remplaçant les

points d'une courbe elliptique *E* par les diviseurs du Jacobien *J* d'une courbe hyperelliptique *C*, pour plus de détail le lecteur pourra consulter [Kob89].

3.3 Implémentation des crypto système sur les courbes hyperelliptiques :

Il y a quelques articles qui parlent de l'implémentation des HECC sur le plan théorique, et d'autres dont les auteurs ont implémenté le crypto système.

Dans [Eng99] Andreas Enge a fait une analyse théorique des calculs dans les courbes hyperelliptiques. La partie de base de son article est consacrée à l'étude et l'analyse de la complexité moyenne de l'arithmétique dans le Jacobien sur quelques corps finis. L'auteur a trouvé un nombre moyen d'opérations pour calculer le plus grand diviseur commun des polynômes sur un corps fini en utilisant l'algorithme d'Euclide étendu. Il a utilisé ses résultats pour calculer la complexité de l'addition et de doublement des diviseurs du groupe Jacobien. Les deux tableaux 3.3 et 3.4 indiquent ses résultats.

	multiplications	*inversions*
$p\neq 2$, *g pair*	$17g^2+5g-7+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+3+\frac{1}{q}O(g^2)$
$p\neq 2$, *g impair*	$17g^2+6g-4+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+\frac{7}{2}+\frac{1}{q}O(g^2)$
$p=2$, *g pair*	$14g^2+6g-6+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+2+\frac{1}{q}O(g^2)$
$p=2$, *g impair*	$14g^2+7g-3+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+\frac{5}{2}+\frac{1}{q}O(g^2)$

Tableau 3.3 : Nombre d'opérations dans le corps pour additionner des diviseurs [Eng99].

	multiplications	*inversions*
$p\neq 2$, *g pair*	$16g^2+7g-6+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+2+\frac{1}{q}O(g^2)$
$p\neq 2$, *g impair*	$16g^2+8g-3+\frac{1}{q}O(g^3)$	$\frac{3}{2}g+\frac{5}{2}+\frac{1}{q}O(g^2)$
$p=2$, $h=1$, *g pair*	$7g^2+3g-3+\frac{1}{q}O(g^3)$	$\frac{1}{2}g+2+\frac{1}{q}O(g^2)$
$p=2$, $h=1$, *g impair*	$7g^2+4g+\frac{1}{q}O(g^3)$	$\frac{1}{2}g+\frac{5}{2}+\frac{1}{q}O(g^2)$
$p=2$, $h=x$, *g pair*	$11g^2+4g-3+\frac{1}{q}O(g^3)$	$\frac{1}{2}g+3+\frac{1}{q}O(g^2)$
$p=2$, $h=x$, *g impair*	$11g^2+5g+\frac{1}{q}O(g^3)$	$\frac{1}{2}g+\frac{7}{2}+\frac{1}{q}O(g^2)$

Tableau 3.4 : Nombre d'opérations dans le corps pour doubler des diviseurs [Eng99].

Le premier tableau indique le nombre d'opérations dans le corps nécessaires pour additionner deux diviseurs distincts. Le second tableau indique le nombre d'opérations dans le corps nécessaires pour doubler deux diviseurs. Dans les deux tableaux, il distingue les genres pairs et impairs et la caractéristique égale à 2 et la caractéristique différente de 2. Il ajoute que si la complexité d'opérations du corps est constante ou croissante avec *log q*, alors le plus petit genre possible est convenable au système. Si la complexité d'opérations du corps croit avec $(logq)^2$, un genre plus grand est recommandé.

Dans [Sma99] quelques aspects des crypto systèmes sur les courbes hyperelliptiques ont été discutés. En particulier l'auteur analyse l'implémentation de la loi de groupe du Jacobien et comment trouver des courbes convenables à la cryptographie. L'article présente aussi une comparaison pratique entre la performance des schémas de signature sur les courbes elliptiques ECDSA et les schémas de signature sur les courbes hyperelliptiques HCDSA. Il a implémenté la loi du groupe du Jacobien pour des genres quelconques sur F_{2^n} et sur F_p ,où p est premier. Il décide de choisir les valeurs de p et n telles que p et 2^n sont plus petits que 2^{32}. Ce choix assure que l'arithmétique de base dans le processeur se fait convenablement entre deux mots (du langage). Il a trouvé pour ce choix du corps le temps de calcul pour certains HCDSA de différents genres obtenus sur un *pentium pro 334 MHz*, en utilisant un compilateur de *Microsoft visual* C^{++}, il a aussi estimé le temps pour des courbes elliptique approximativement de même ordre, les résultats sont dans le tableau 3.5.

courbe	corps	signature	vérification
HCDSA g=5	$F_{2^{31}}$	18 ms	71 ms
HCDSA g=6	$F_{2^{31}}$	26 ms	98 ms
HCDSA g=7	$F_{2^{31}}$	40 ms	156 ms
ECDSA	$F_{2^{161}}$	4 ms	19 ms
ECDSA	F_p	3 ms	17 ms

Tableau 3.5 : Le temps de calcul pour HCDSA et ECDSA [Sma 99].

Il est important de noter que l'implémentation des courbes elliptiques n'utilise pas de représentations spéciale de corps tels que l'utilisation de la structure des sous corps. Il est à remarquer que le temps nécessaire pour signer ou vérifier la signature dans le cas du Jacobien d'une courbe hyperelliptique est quatre fois plus que le temps dans le cas des courbes elliptiques. Vu la difficulté de trouver une courbe hyperelliptique convenable à la cryptographie et la non performance des algorithmes des courbes hyperelliptiques par rapport à ceux des courbes elliptiques, il voit qu'il n'est pas intéressant d'utiliser les courbes hyperelliptiques.

Yasuyuki Sakai, Kouichi Sakurai et Hirokazu Ishizuka [SSI98] ont recherché sur le problème du logarithme discret dans le jacobien d'une HEC, ils ont clarifié les avantages pratiques des HECC comparés aux ECC et au crypto système RSA (dont la sécurité repose sur la difficulté de la factorisation des grands entiers) . Ils ont insisté sur les courbes définis sur un corps de caractéristique 2 ayant des genres *g=3* et *11*. Ils ont discuté aussi l'implémentation de tels crypto systèmes ils ont donné seulement des résultats théoriques à la fin de leur article.

Les auteurs de l'article [SS98] sont intéressés au problème du logarithme discret sur les courbes hyperelliptiques HCDLP définis sur des corps de caractéristiques petites 2, 3, 5, et 7. Ils ont implémenté des HECC sur des corps du type F_{2^n} en software sur un micro ordinateur Alpha 467 MHz, et sur un Pentium-II 300 MHz. Ils ont implémenté le temps de calcul dans les deux machines pour l'addition, le doublement et la multiplication scalaire en utilisant des courbes offrant une même sécurité que le crypto système RSA-1024, (tableau 3.6), ils ont fait de même avec RSA-2048, tableau 3.7. Dans l'appendice de leur article, ils ont cité des courbes hyper elliptiques convenables à l'utilisation en cryptographie.

g	$J(v^2+v=f(u);\ F_{2^n})$		*addition (ms)*		*doublement (ms)*		*multiplication scalaire (ms)*	
	F_{2^n}	*f(u)*	Alpha	P-II	Alpha	P-II	Alpha	P-II
3	$F_{2^{59}}$	u^7	0.54	67.6	0.26	34.1	83.3	$1.17.10^4$
4	$F_{2^{41}}$	$u^9+u^7+u^3+1$	0.55	67.2	0.26	33.3	96.6	$1.09.10^4$
5	$F_{2^{41}}$	$u^{11}+u^5+u+1$	0.88	109	0.48	58.7	183	$2.36.10^4$
6	$F_{2^{29}}$	$u^{13}+u^{11}+u^7+u^3+1$	0.83	2.68	0.44	1.45	159	476

Table 3.6 : Temps de calculs pour Jacobien ayant une même sécurité que RSA-1024 [SS98]

g	*J*	*C*	*Taille de*	*addition*	*doublement*	*multiplication scalaire*
			P_{max}	*(ms)*	*(ms)*	*(ms)*
3	$J(C; F_{2^{89}})$	$v^2+v=u^7$	246-bit	85.3	42.8	$2.57.\ 10^4$
3	$J(C; F_{2^{113}})$	$v^2+v=u^7$	310-bit	118	58.9	$3.79.\ 10^4$
11	$J(C; F_{2^{47}})$	$v^2+v=u^{23}$	310-bit	5.04	3.13	$1.74.\ 10^3$

Table 3.7 : Temps de calculs pour Jacobien ayant une même sécurité que RSA-2048 [SS98].

Uwe Krieger a implémenté dans une partie de sa thèse en 1997 une signature basé sur les courbes hyperelliptiques [Kri97]. Il a implémenté ensuite des versions des HECC et des ECC. Comme conclusion, il a comparé les temps nécessaires pour signer par RSA, ECC et HECC, il constate enfin que les ECC et aussi les HECC sont mieux que RSA du point de vue vitesse d'exécution.

Crypto système	Temps de calcul
RSA avec 1024 bit	0.53 sec
ECC	0.11 sec
HECC de genre g=2	0.84 sec

Table 3.8 : Comparaison de trois crypto systèmes [Kri97].

3.4 -Attaques des crypto systèmes sur les courbes hyperelliptiques :

L'attaque d'un crypto système sur les courbes hyperelliptiques revient à la résolution du problème du logarithme discret dans le Jacobien HCDLP, (dans le groupe formé des points d'une courbe elliptique ECDLP).

Le problème du logarithme discret dans un groupe multiplicatif F_q^* d'un corps fini bien choisi est difficile. Pohlig et Hellman [PH78] observaient qu'une instance du DLP dans un groupe *G* cyclique d'ordre *n* peut être réduit facilement en une instance du DLP dans un sous-groupe de *G* d'ordre premier. Le meilleur algorithme connu pour résoudre le DLP est celui de Pollard rho [Pol78], cet algorithme a une complexité estimée à $\sqrt{(\pi n/2)}$ opérations dans ce groupe. Il peut être facilement implémenté dans un network de *r* processeurs rapides. Pour avoir une résistance maximum aux algorithmes de Pohlig Hellman et de Pollard rho, l'ordre *n* du groupe doit avoir un facteur premier très grand (plus grand que 2^{160}). Les algorithmes pour le DLP meilleurs que la méthode de Pollard sont connus pour une classe de groupes restreint, entre autre le Jacobien des courbes hyperelliptiques de grand genre. On présente dans ce travail trois différentes méthodes pour résoudre le problème du logarithme discret dans le groupe du Jacobien d'une courbe hyperelliptique.

3.4.1- Attaque de Fray et Rack :

Fray et Rack [FR94] ont décrit en 1994 dans une publication un algorithme pour résoudre le problème du logarithme discret dans le Jacobien $J(F_q)$ défini sur un corps fini F_q. Cet algorithme réduit facilement le DLP dans le Jacobien $J(F_q)$ au DLP dans le groupe multiplicatif d'une extension F_{q^k} de F_q. Le degré de cette extension *k* est le plus petit entier pour lequel $\#J_C(F_q)$ (ou bien le plus grand facteur de $\#J_C(F_q)$) divise q^k-1. On prévoit pour la plus part des courbes définies sur F_q que *k* est grand, mais pour quelques courbes *C*, *k* est effectivement petit et par conséquent la réduction induit un algorithme sous exponentiel pour résoudre le DLP dans $J_C(F_q)$. Par exemple, si *C* est une courbe elliptique super singulière défini sur un corps fini, alors $k \in \{1,2,3,4,5,6\}$.

3.4.2 - Attaque par descente de Weil :

Fray [Fr01] le premier qui a proposé l'utilisation de la descente de Weil pour réduire le ECDLP dans une courbe elliptique sur un corps fini $GF(q^n)$ au problème DLP dans la variété Jacobienne d'une courbe algébrique de genre grand sur un sous-corps propre $GF(q)$ de $GF(q^n)$. Soient l et n deux entiers positifs, $q=2^l$. Considérons la courbe elliptique super singulière définie sur $K=GF(q^n)$ par l'équation :

$$E: y^2 + xy = x^3 + ax^2 + b,\ a \in K, b \in K^*.$$

On suppose que $\#E(K) = d\,r$, où d est petit (par exemple $d=2$ ou $d=4$) et r est un nombre premier. Ainsi $r \approx q^n$. Soit $b_i = b^{q^i}$, on définit :

$$m(b) = dim_{F_2}\left(Span_{F_2}\left\{(1, b_0^{1/2}), (1, b_1^{1/2}), \ldots, (1, b_{n-1}^{1/2})\right\}\right)$$

Supposons maintenant que soit n impair, ou $m(b)=n$, ou $Tr_{K/F2}(a)=0$. Gaudry, Hess et Smart [GHS02] ont trouvé comment utiliser la descente de Weil pour réduire le problème ECDLP dans le sous-groupe d'ordre r de $E(K)$ au problème du logarithme discret DLP dans un sous-groupe d'ordre r du Jacobien $J_C(F_q)$ d'une courbe hyperelliptique C de genre g définie sur F_q. On construit d'abord la restriction de Weil $W_{K/F2}$ des scalaires de E, qui est une variété abélienne de dimension n sur F_2. Alors $W_{K/F2}$ est intersecté avec $n-1$ hyperplans pour obtenir éventuellement la courbe hyperelliptique C. L'algorithme de réduction avec le meilleur algorithme de résolution du HCDLP sont appelé *attaque GHS* du ECDLP. Le genre de C est $g=2^{m-1}$ ou $2^{m-1}-1$, où $m=m(b)$.

Pour que l'attaque GHS puisse avoir un succès, le DLP dans $J_C(F_q)$ est solvable en un temps moins que celui de l'algorithme de Pollard pour résoudre le ECDLP. En général $m \approx n$, donc $g \approx 2^{n-1}$ et $\#J_C(F_q) \approx q^{2^{n-1}}$ et l'attaque GHS est infaisable. L'attaque GHS est intéressante seulement si m est petit, disons $m \approx log_2\, n$, donc $g \approx n$ et $\# J_C(F_q) \approx q^n$.

Les auteurs de [MQ01] ont analysé la formule qui donne $m(b)$, et il a été observé que l'attaque GHS est impossible pour les courbes elliptiques sur $GF(2^n)$ où $n \in [160,600]$ est premier, et elle est faisable pour quelques courbes elliptiques

sur $GF(2^n)$ où n est composé, consulter pour un exemple [JMS01] où une instance du ECDLP définie sur $GF(2^{124})$ a été résolue par une réduction à une instance du HCDLP de genre 31 sur $GF(2^4)$, et l'utilisation de l'algorithme de Enge-Gaudry des HEC de petit genre.

3.4.3 - Attaque par calculs d'indice :

Les algorithmes des calculs d'indice ont été appliqués à des problèmes de cryptographie intéressants tels que le problème du logarithme discret dans des corps finis et des classes des corps de nombres. Les mêmes idées peuvent être appliquées aux calculs du DLP dans le Jacobien d'une courbe hyperelliptique de genre g définie sur un corps fini F_q. Si le genre g est grand avec q, on peut trouver des algorithmes sous exponentiels en $n=q^g$ qui sont dans $O(L_n[c])$,où :

$$L_n[c] = exp\left((c+o(1))\sqrt{lognloglogn}\right)$$

avec c >0 réel, *log* : logarithme de base 2.

La notion des diviseurs *t-faible* joue un rôle fondamental dans les algorithmes des calculs d'indice.

Définition 3.7 :

Un diviseur $D=div(a,b)$, pour lequel a est un polynôme irréductible sur F_q est dit *t-faible* s'il se décompose en $D=\sum_{i=1}^{L} e_i\, div(a_i,b_i)$, avec $e_i \in Z$ et $max\{\deg a_i\} \leq t$.

Pour déterminer la décomposition de $D=div(a,b)$ en diviseurs premiers, on factorise a en des facteurs a_i irréductibles sur F_q : $a = a_1^{e_1} a_2^{e_2} ... a_L^{e_L}$ et soit $b_i = b \bmod a_i$ alors $D=\sum_{i=1}^{L} e_i\, div(a_i,b_i)$.

Il y a deux stratégies principales pour résoudre le HCDLP par calculs d'indice. Soient $D_1 \in J_C(F_q)$ et $D_2 \in \langle D_1 \rangle =$ le groupe engendré par D_1, on cherche $l = Log_{D_1} D_2$.

Stratégie 1 :

On utilise dans cette stratégie l'idée de base de l'algorithme de Hafner et Mc Curley [HM89] pour l'étude du DL dans les corps de nombres quadratiques

imaginaires. La méthode procède comme suit :

Soit $S= \{ P_1,P_2,\ldots,P_n\}$ la base de facteurs constituée par toutes les décompositions des diviseurs premiers ramifiés $P_i=div(a_i,b_i)$.

Trouver m diviseurs principaux *t-faible* ($m>n$), chacun d'eux satisfait une relation de la forme : $\sum e_j P_j \sim 0$.

Si S engendre $J_C(F_q)$, alors l'application $\phi : Z^n \to J_C(F_q)$, où $\phi(e_1,\ldots,e_n)=\sum e_j P_j$ est un homomorphisme de groupe surjectif et $\phi : Z^n / ker\,\phi \cong J_C(F_q)$. Chaque relation induit un élément $\vec{e}_i=(e_{i1},e_{i2},\ldots,e_{in}) \in ker\,\phi$, et si l' ensemble de m relations engendre un système générateur complet de $ker\,\phi$ alors

$$J_C(F_q) \cong Z/d_1Z \oplus Z/d_2Z \oplus \ldots \oplus Z/d_nZ$$

où $(d_1,d_2,\ldots,d_n)$ sont les éléments diagonaux de la forme normale de Smith (SNF) de la relation matricielle $A=(\vec{e}_1,\vec{e}_2,\ldots,\vec{e}_n)$ (les relations sont écrit comme colonne de A). Les générateurs X_i de chaque sous-groupe cyclique Z/d_iZ peut être retrouvé par les matrices de transformation uni modulaire $P=(P_{ij})$ et $Q=(Q_{ij})$ telles que $P^{-1}AQ = SNF(A)$ et $X_i = \sum_{j=1}^{n} P_{ji}P_j$.

Trouver les représentation de D_1 et de D_2 dans la somme directe $Z/d_1Z \oplus \ldots \oplus Z/d_nZ$. Si D_1 et D_2 peuvent se factoriser dans S comme $D_1 \sim \sum \alpha_i P_i$ et $D_2 \sim \sum \beta_i P_i$ alors $D_1 \sim \sum \alpha_i' X_i$ et $D_2 \sim \sum \beta_i' X_i$, où $(\alpha_1',\alpha_2',\ldots,\alpha_n')=P^{-1}(\alpha_1,\alpha_2,\ldots,\alpha_n)^T$ et $(\beta_1',\ldots,\beta_n')=P^{-1}(\beta_1,\ldots,\beta_n)^T$.

Finalement, étant donné les représentations de $D_1=(\alpha_1',\alpha_2',\ldots,\alpha_n')$ et $D_2=(\beta_1',\beta_2',\ldots,\beta_n')$ dans la somme directe $Z/d_1Z \oplus Z/d_2Z \oplus \ldots \oplus Z/d_nZ$, on peut résoudre le DLP en utilisant le théorème des restes de Chinois généralisé pour trouver un entier l tel que les congruences $\alpha_1' \equiv \beta_i \pmod{d_i}$, $1 \le i \le n$ soient satisfaites simultanément.

Stratégie 2 :

Elle n'est applicable que si le cardinal de $J_C(F_q)$ est connu.

Soit $S=\{ P_1,P_2,\ldots,P_n\}$ la base de facteurs formée par toutes les décompositions des diviseurs premiers de degré $\le t$. On essaye de factoriser des diviseurs de la forme $\alpha D_1+\beta D_2$ sur S. Chaque diviseur *t-faible* satisfait une relation de la forme

$\alpha_i D_1+\beta_i D_2 \sim R_i=\sum_j e_{ij}P_j$. Quand *(n+1)* relations différentes sont obtenues, on applique l'algèbre linéaire modulo $\#J_C(F_q)$ pour trouver une combinaison linéaire non triviale $\sum_i \gamma_i \vec{e}_i=(0,0,...,0)$ qui implique que $\sum_{i=1}^{n} \gamma_i R_i=0$. Ainsi $\sum_i \gamma_i (\alpha_i D_1+\beta_i D_2)=0$ et $Log_{D_1} D_2=-\left(\sum_i \gamma_i \alpha_i\right) / \left(\sum_i \gamma_i \beta_i\right) mod \#J_C(F_q)$.

Remarquons que l'hypothèse que $\#J_C(F_q)$ soit connu nous permet d'éviter les calculs difficiles de la forme normale de Smith de la stratégie précédente.

3.4.4- Quelques algorithmes proposés *(qui suivent l'une des deux stratégies)* :

a. Algorithme ADH :

Adleman, De Marrais et Huang [ADH94] ont présenté le premier algorithme de calculs d'indice pour résoudre le HCDLP. Leur algorithme est pour *q* premier impair. Cette étude a été généralisée pour *q* arbitraire par Bauer [Bau99]. Cet algorithme ne suppose pas que $\#J_C(F_q)$ soit connu et suit la stratégie 1.

b. Algorithme de Flassenberg et Paulus :

Flassenberg et Paulus [FP99] sont les premiers à appeler la méthode de génération des relations de Hafner et Mc Curley [HC89] pour le HCDLP. Leur méthode est applicable à des courbes sur des corps de caractéristique impair, et elle suit la stratégie 1.

c. Algorithme de Bauer et Enge pour corps finis arbitraires :

L'algorithme ADH est indépendamment généralisé aux courbes hyperelliptiques sur des corps finis arbitraires par Bauer [Bau99]et par Enge [Eng01]. L'algorithme de Bauer a une même forme que l'algorithme ADH et ils ont le même temps de calcul sous les mêmes hypothèses. Comme l'algorithme de Flassenberg et Paulus, l'algorithme de Enge applique la méthode de Hafner et Mc Curley au HCDLP pour engendrer des relations et suivre la stratégie 1. Enge est le premier à illustrer la dépendance en temps de calcul entre *g* et *q*, il trouve que si $g \geq \eta \log q$, pour un nombre positif η, alors $n=O\left(L_{q^g}\left\lfloor \mu+1/\sqrt{\eta}\right\rfloor\right)$.

d. **Algorithme de Gaudry pour petit genre :**

Gaudry [Gau00] a présenté un algorithme de calculs d'indice utilisant le fait que $\#J(F_q)$ soit connu , et suivant la méthode de la stratégie 2. La base de facteurs S est formée de diviseurs premiers de degré 1.

Pour passer au petit genre, Gaudry a analysé son algorithme pour un genre g fixé et q variable, il trouve un temps de calcul de $O(g^3q^2log^2q+g^2g\ !qlog^2q)$, donc le HCDLP est plus facile à résoudre qu'un algorithme générique si $g>4$, et très récemment Thériault [The03] a amélioré l'algorithme de Gaudry pour petit genre.

3.4.5- Résultat expérimental :

Un bon algorithme proposé pour résoudre le ECDLP est la version parallèle de celui de Pollard rho (algorithme de Pollard installé dans un réseau de r processeurs). Certicom (Entreprise canadienne de cryptographie) a initialisé un ECC challenge en novembre 1997 pour encourager et stimuler les recherches sur les crypto systèmes sur les courbes elliptiques. Escott, Sager, Selkirk et Tsapakides [ESST99] dans leur implémentation en 1998 de l'algorithme parallèle de Polard rho, ils ont résolu une instance difficile du ECDLP sur une courbe elliptique définie sur un corps premier F_p, d'une clef de 97 bits. Ils ont utilisé pour cet effet 1200 machines installées dans 16 pays travaillaient pendant 53 jours.

En avril 2000, des chercheurs français de l'INRIA (institut national de recherche en informatique et automatique), aidés par près de 10.000 internautes, ont annoncé avoir casser un système sur ECDLP d'une clef de 109 bits. Il a fallu pour cela pendant 4 mois la participation de 9.500 ordinateurs fournis par 1.300 internautes enthousiastes dans une quarantaine de pays.

Chapitre 4 : CHOIX DES COURBES HYPERELLIPTIQUES CONVENABLES A L'UTILISATION EN CRYPTOGRAPHIE

Pour assurer la sécurité d'un crypto système basé sur le problème du logarithme discret dans un groupe abélien fini, il faut que l'ordre de ce groupe soit divisible par un nombre premier très grand. Le problème qui se pose dans le cas d'une courbe hyper elliptique est la difficulté de trouver l'ordre du Jacobien $J_C(F_q)$. Théoriquement, on peut calculer l'ordre du groupe $\# J_C(F_q)$ en un temps polynomial en utilisant des méthodes dues à Adleman, Huang et Pila [AH96] et [Pil96], leurs méthodes sont des généralisations de la méthode de Schoof [Sch85] qui est utilisée pour compter les points d'une courbe elliptique. La méthode due à (AHP) n'est pas encore implémentée, elle est facile à comprendre et apparemment difficile à implémenter. Un des problèmes dans le cas d'une courbe hyperelliptique est qu'il n'y a une analogie connue pour l'algorithme de Schoof à la méthode de Atkin et Alkie. Gaudry et Harley ont implémenté récemment une généralisation de l'algorithme de Schoof-Atkin-Elkie, ils ont pu avec cet algorithme, déterminer l'ordre du jacobien d'une courbe de genre 2 définie sur F_p où : $p = 10^{19}+51$ [GH00]. On constate ainsi que la difficulté du calcul de l'ordre du Jacobien d'une courbe n'est pas basée sur des problèmes mathématiques mais sur la difficulté de programmation d'un algorithme facile et efficace. Pour avoir une bonne courbe hyperelliptique à utiliser en cryptographie, il faut que l'ordre du Jacobien de cette courbe possède un diviseur premier r très grand (d'au moins 45 chiffres) et que r ne divise pas q^k-1 pour tout entier

k petit pour lequel le DLP dans $GF(q^k)$ est faisable ($1 \leq k \leq 2000 / log_2 q$) suffit, mais il n'est pas facile de trouver une courbe vérifiant cette propriété vu la difficulté de l'implémentation d'un algorithme qui calcule l'ordre du Jacobien d'une courbe hyperelliptique, néanmoins, on peut utiliser la multiplication complexe (CM) pour construire des courbes de genre 2 qui vérifie cette propriété [Wen02], ou bien utiliser des courbes de la forme $v^2 + v = u^n$ [Kob98].

4.1- Construction des courbes de genre 2 pour l'utilisation en cryptographie :

La construction d'une courbe hyperelliptique de genre 2 définie sur un corps premier $Fp,(p \neq 2)$ ou une petite extension d'un corps premier suit les idées principales de l'algorithme suivant,(consulter [Wen02] pour les détails de cet algorithme):

1. Fixer un CM-corps K et trouver un bon nombre premier p et un ordre possible n.
2. Chercher toutes les variétés abéliennes sur $\boldsymbol{C}$ principalement polarisée ayant une multiplication complexe par l'ordre maximal O_K.
3. Calculer les 10 constantes theta.
4. Calculer les invariants de Igusa j_1, j_2, j_3 avec les constantes theta et réduire ces invariants modulo p.
5. Calculer les invariants de Mestre Q_{ij} et H_{ijk} avec j_1, j_2, j_3.
6. Appliquer l'algorithme de Mestre pour obtenir l'équation de la courbe C.
7. Voir si l'ordre du groupe $\#J(K)$ est égal à n.

Le problème dans cette technique est qu'elle est applicable seulement dans le cas du genre 2.

Exemple 4.1 :(*d'une bonne courbe de genre 2*)

On utilise la multiplication complexe donnée par $Q\left(i\sqrt{3+\sqrt{7}}\right)$.

Considérons le CM-corps $Q\left(i\sqrt{3+\sqrt{7}}\right)$ ayant deux nombres de classes et 2 polarisations.

Un nombre premier convenable est donné par :

$$p = 580943314814642181310688596463593$$

ce qui va nous donner deux ordres possibles du groupe :

$n_1 =$337495135027824453733283789094149750817279621391373902756574895952
et
$n_2 =$337495135027824453733281165453395182011292833807497899521740278848

Notons que :

$n_1 = 16 \cdot q$, où q est un nombre premier.

La courbe correspondante à l'ordre n du groupe est donnée par :

$C : y^2 = x^5 +$ 474727596586211034284401845850785 x^4
+ 314748234596474418739550339957648 x^3
+ 314740766532984346191929527993409 x^2
+ 574397988361190658944043563780018 x
+ 546228693859470379418770593594687.

4.2- Construction des courbes de la forme $v^2+v=u^n$ pour l'utilisation en cryptographie :

Le but de construire des courbes de cette forme est la facilité du calcul de la fonction Zeta.

Soit $n = 2g+1$ un nombre premier impair, et soit $p \equiv 1 \pmod n$.

Considérons la courbe hyperelliptique

$$C : v^2 + v = u^n \text{ sur } F_p \qquad (4.1)$$

Soit $\xi = e^{2\pi i/n}$, et soit $\alpha \in F_p$, supposons que α n'est pas une puissance n$^{\text{ième}}$. Il existe une unique application multiplicative χ sur F_p^* telle que

$\chi(\alpha)=\xi$. On prolonge cette application à F_p en posant $\chi(0)=0$. La somme de Jacobi du caractère χ avec elle-même est définie par :

$$J(\chi,\chi)=\sum_{y\in F_p}\chi(y)\,\chi(1-y) \qquad (4.2)$$

Pour $1\le i\le n-1$ soient σ_i les automorphismes du corps $Q(\xi)$ telles que $\sigma_i(\xi)=\xi^i$. Alors par un argument de calcul simple on trouve le nombre M de points de la courbe (4.1) y compris le point à l'infini :

$$M=p+1+\sum_{i=1}^{n-1}\sigma_i\left(J(\chi,\chi)\right)$$

La fonction Zeta de cette courbe est donnée par :

$$Z(C/F_p,T)=\frac{\prod_{i=1}^{n-1}\left(1+\sigma_i\left(J(\chi,\chi)\right)T\right)}{(1-T)(1-pT)}$$

Le nombre N de points du Jacobien J de la courbe C est égale à la valeur 1 du numérateur de $Z(C/F_p,T)$:

$$N=\prod_{i=1}^{n-1}\left(\sigma_i(J(\chi,\chi)+1\right)$$

On peut procéder de façon analogue, pour calculer le nombre de points du Jacobien d'une courbe de la forme :

$$v^2+v+(1-\beta^i)/4=\beta^i\,\alpha^j u^n$$

où α,β ne sont pas des puissances n$^{\text{ième}}$, $i=0,1$ et $j=0,1,\ldots,n-1$.

Pour ces courbes le nombre $N_{i,j}$ des points du Jacobien est donné par :

$$N_{i,j}=\prod_{i=1}^{n-1}\left(\sigma_i(J(\chi,\chi)+(-1)^i\xi^j\right)$$

Pour $n=2g+1\ge 5$, si on choisit p comme étant un nombre de Mersenne généralisé de la forme $p=\frac{a^n-1}{a-1}$, on peut trouver rapidement $N_{i,j}$.

Exemple 4.2 :

Soit $n=5$ et $a\ge 10^{15}$, on peut trouver rapidement les cas suivants où $p=a^4+a^3+a^2+a+1$ est premier et l'ordre du Jacobien sur F_p est divisible par un grand nombre premier :

a= 100003, p=100013000640014200121,

$N_{0,1}$= 5 . 2000520059203862158324190070180683302981

a= 100018, p=100073019992433811151,

$N_{1,0}$= 100146093314071777867678004569575770133 41.

Exemple 4.3 : (*d'une courbe non vulnérable à la réduction de Fray et Rack)*
Considérons la courbe hyperelliptique C : $v^2+v=u^{383}+1$ sur F_2 de genre

$g=191$. On peut facilement calculer la fonction Zeta de cette courbe grâce à la forme spéciale de son équation. Le nombre de points du Jacobien de cette courbe est de la forme $N=3\ q_{58}$, où q_{58} est le nombre premier à 58 chiffres :

q_{58} =104618362256444679397263157049793709568656318343345253034 7.

Cette courbe n'est pas vulnérable à la réduction de Fray et Rack puisque q_{58} ne divise pas 2^k-1 pour $k \leq 2000$. Mais, puisque le genre de cette courbe est grand par rapport au logarithme du caractéristique du corps, il faut se méfier de l'utilisation de telles courbes, elles peuvent être vulnérables à l'attaque de Adleman, DeMarrais et Huang [ADH94], actuellement leurs algorithme est applicable seulement en caractéristique impaire, mais on peut développer à peu prés de la même façon un algorithme analogue pour la caractéristique $q=2$.

CONCLUSION

Avant la découverte de la cryptographie à clef publique à la fin des années soixante dix, la communication secrète utilisait la cryptographie à clef secrète ; les deux inconvénients essentiels de cette dernière sont que, si un intercepteur a une information sur l'algorithme de chiffrement, il aura automatiquement une information sur le déchiffrement, et que les deux personnes qui échangent les messages doivent se rencontrer pour convenir d'une clef secrète. La cryptographie à clef publique a trouvé une solution pour cette dernière difficulté; mais elle a un autre genre de problèmes, à savoir, une exécution très lente et un espace mémoire nécessaire pour l'exécution des algorithmes de chiffrement et de déchiffrement très important. Par exemple, un système symétrique avec une clef de 80 bits et un système RSA avec une clef de 1024 bits ont une même résistance aux attaques. Néanmoins, les crypto système à clef publique utilisant les courbes elliptiques et hyperelliptiques nous offrent une même sécurité que le système RSA en utilisant une clef beaucoup moins longue ; par exemple, une clef de 163 bits pour un système sur les courbes elliptiques offre une même sécurité qu'un système RSA avec une clef de 1024 bits. En outre de la vitesse de l'exécution et l'espace mémoire nécessaire pour l'exécution d'un algorithme cryptographique, un autre critère à respecter pour adopter un système cryptographique est la facilité d'implémentation des algorithmes de ce système. Les crypto systèmes sur les courbes hyperelliptiques ne sont pas malheureusement faciles à implémenter, à cause de l'addition et la multiplication par les scalaires des diviseurs du Jacobien et les diverses opérations sur le corps sur lequel la courbe est définie. Par contre, les crypto systèmes sur les courbes elliptiques sont beaucoup plus faciles à implémenter. L'autre inconvénient des crypto systèmes sur les courbes hyperelliptiques est la difficulté du choix d'une courbe pour avoir un système sûr. Finalement, vu les nombreuses attaques possibles des crypto systèmes sur les courbes

hyperelliptiques et tous les inconvénients précédents, on recommande d’utiliser les systèmes sur les courbes elliptiques.

Références :

[ADH94] L. Adleman, J. De Marrais and M. Huang, ‘‘*A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobians of large genus hyperelliptic curves over finite fields’’, Algorithmic number theory, LNCS* 877, (1994), 28-40.

[AH96] L.Adleman, M-D.Huang,*’’Counting rational points on curves and abelian varieties over finite fields’’, In ANTS-2: Algorithmic Number Theory, Springer-Verlag, LNCS* 1122 (1996), 1-16.

[Bau99] M. Bauer, ’’ *A subexponential algorithm for solving the discrete logarithm problem in the Jacobian of high genus over arbitrary finite fields’’,* preprint, 1999.

[BK98] Balasubramanian , N. Koblitz, ‘’*The improbability that an elliptic curve has subexponential discrete log problem under the Menazes-Okamoto-Vanstone algorithm’’* , J. cryptology 11 (1998), 141-145.

[ElG85] El Gamal, ‘’*A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Trans. Information Theory 31 (1985), 469-472.

[Eng01] A. Enge, *’’A general framework for subexponential discrete logarithm algorithms in group of unknown order’’, Finite Geometries, developments in mathematics* vol. 3, *Kluwer Academic Publishers, Dordrecht* 2001, 133-146.

[Eng99] Andreas Enge, ‘’*The extended Euclidean algorithm on polynomials, and the computational efficiency of hyperelliptic cryptosystems’’, Preprint November* 1999.

[ESST99] Escott, Sager,Selkirk and Tsapakides, ‘’*Attacking elliptic curve cryptosystems using the parallel Pollard rhomethod’’, CryptoBytes – The Technecal News letter of RSA Laboratoiries,* vol 4, n°2, (1999) 15-19.

[FP99] *M. Flassenberg and S. Paulus, ‘’Sieving in function fields’’, Experimental Mathematics,* 8 (1999), 339-349.

[Fr01] Fray, ‘’*applications of arithmetical geometry to cryptographic constructions ‘’, proceedings of the fifth international conference on finite fields and applications,* springer -Verlag, 2001,128-161.

[FR94] G.Fray and H.Rack, ‘’*A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves’’*, Mathematics of computation, 62 (1994), 865-874.

[Gau00] P. Gaudry, ‘’*An algorithm for solving the discrete log problem on hyperelliptic curves’’, Advances in Cryptology- EUROCRYP 2000, LNCS* 1807, 2000, 19-34.

[GH00] P. Gaudry and R. Harly, *’’Counting points on hyperelliptic curves over finite fields, ANTSIV* (2000), 313-332.

[GHS02] P. Gaudry, F. Hess and N. Smart, ‘’*constructive and destructive facets of Weil descent on elliptic curves’’ journal of cryptology,* 15(2002),19-46.

[HC89] J. Hafner and K. Mc Curley, ‘’*A rigorous subexponential algorithm for computation of class groups’’, Journal of the American Mathematical Society,* 2 (1989), 837-850.

[JMS01] M. Jacobson, A. Menezes and A. stein, *''Solving elliptic curve discrete logarithm problems using Weil descent'', journal of the Ramanujan Mathematical Society,* 16 (2001), 231-260.
[Kob89] N.Koblitz, ''Hyperelliptic cryptosystems'', journal of cryptology, 1 (1989), 139-150.
[Kob98] N.Koblitz, *''Algebraic aspects of cryptography''*, Algorithms and computation in mathematics. Vol 3,Springer Verlag 1998.
[Kri97]Uwe Krieger, *signture.c, February 1997. Diplomarbeit, Universitat Essen, Fachbereich 6 (Mathematik und informatik).*
-MHZ96] A.Menazes, Y.Hong, R.Zuccherato, *''An elementary introduction to hyper elliptic curves'',*Technical report CORR 96-19, Department of C&O, University of Waterloo, Ontario, Canada November 1996.
[MOV93] Menezes, Okamoto, Vanstone *''Reducing elliptic curve logarithms to logarithms in a finite field'',*IEEE Trans. Information Theory 39 (1993), 1639-1646.
[MQ01] A. Menazes and M. Qu, *'' Analyses of the Weil descent attack of Gaudry, Hess and Smart'', Topic in cryptology- CT- RSA 2001,LNCS* 2020 2001, 308-313.
[PH78] S. Pohlig et M. Hellman, *''An improved algorithm problem for computing logarithms over GF(p) and its cryptographic significance''*, IEEE Transactions on Information Theory, 24 (1978), 106-110.
[Pil96] Pila, *''Frobenius maps of abelian varieties and finding roots of unity in finite fields'', Math. Comp*., 55(1996),745-763.
[Pol78] J. Pollard, *''Monte carlo methods for index computation mod p''*, Mathematics of computation, 32 (1978), 918-924.
[Sch85] Schoof,*''Elliptic curves over finite fields and the computation of square roots mod p''. Math. Comp.,* 44 (1985),483-494.
[Sma99] Nigel P. Smart, *''On the performance of hyperellptic cryptosystems'', In Advances in cryptology – EUROCRYPT 99, volume* 1592, 165-175, BERLIN 1999, *Springer-verlag. Lecture Notes in Computer Science.*
[SS98] Y. Sakai, K. Sakurai, *''Design of hyperellptic cryptosystems in small characteristic and a software implementation over GF(2^n)''. In advances in cryptology ASIACRYPT'98,* 1514, 80-94 (1998), *Springer-verlag. Lecture Notes in Computer Science.*
[SSI98] Y. Sakai, K. Sakurai and H. Ishizuka, *''Secure hyperelliptic cryptosystems and their performance'', In Public Key Cryptography,* 1431, 164-181, Berlin (1998). *Springer-Verlag, Lecture Notes in Computer Science.*
[The03] N. Thériault, *''Index calculus attack for hyperelliptic curves over finite fields of small genus''*, Preprint, 2003.
[Wen02] A.Weng; *''Constructing hyperelliptic curves of genus 2 suitable for cryptography''*; Journal of the American Mathematical Society, Article electronically published on may 3, 2002.

QUELQUES NOTATIONS ET ABREVIATIONS UTILISES :

$K[u]$: L'ensemble des polynômes à une indéterminé u sur un corps K.
$K[u,v]$:L'ensemble des polynômes à deux indéterminés u et v sur un corps K.
g: le genre d'une courbe hyperelliptique C.
L : extension du corps K.
$C(L)$: l'ensemble des points P de LxL satisfaisant l'équation de la courbe C.
$C(\overline{K})$: noté C.
$\underline{F_q}=GF(q)$: corps fini à q éléments, ($q=p^n$, p premier).
$K[C]$: L'ensemble des fonctions polynomiales sur une courbe C.
$N(G)$: norme d'une fonction G définie sur une HEC.
$Ord_P(G)$: ordre d'une fonction G au point P.
D : l'ensemble des diviseurs sur une courbe hyperelliptique.
D^0 : l'ensemble des diviseurs de degré 0.
$supp(D)$: support d'un diviseur D.
J : Jacobien d'une HEC.
$Z_C(T)$: la fonction Zeta d'une HEC.
$\#J(F_q)=N_r$, $(q=p^r)$: cardinal du Jacobien d'une HEC définie sur $GF(p^r)$.

RSA : cryptosystème à clef publique RSA (Rivest, Shamir et Adleman).
HEC : courbe hyperelliptique.
HECC : cryptosystèmes sur les courbes hyperelliptiques.
ECC : cryptosystèmes sur les courbes elliptiques.
DLP : problème du logarithme discret.
HCDLP : problème du logarithme discret sur le jacobien d'une HEC.
ECDLP : problème du logarithme discret sur l'ensemble des points d'une courbe elliptique.
DSA : algorithme de signature numérique.
ECDSA : algorithme de signature numérique sur une courbe elliptique.
HCDSA : algorithme de signature numérique sur une courbe hyperelliptique.