Université Mohammed Premier
Faculté Pluridisciplinaire de Nador
Département de Mathématiques
Nador

Troisième Année Universitaire
Semestre 6
Filière : SMA

Arithmétique 2

Notes de Cours de Cryptographie

Professeur : Taoufik Serraj

Année universitaire : 2020-2021
Version : 1

Table des matières

Introduction à la cryptographie moderne

La cryptologie est la science de la sécurisation des données. Située à la frontière des mathématiques, de l'informatique et de la physique, elle est divisée en deux branches, la cryptographie et la cryptanalyse. La cryptographie s'intéresse à la conception et l'étude des mécanismes répondant aux besoins de la sécurité de l'information, également à l'amélioration de leurs performances, alors que la cryptanalyse analyse les faiblesses des systèmes cryptographiques en proposant et en étudiant les différentes attaques possibles.

Aperçu historique

Cette science moderne remonte à l'antiquité. En effet, dans l'ancienne Égypte, les scriptes utilisaient des symboles hiéroglyphiques que pouvaient, eux seuls, déchiffrer. Les Grecs et les Romains ont eux aussi utilisé la cryptographie à des fins militaires et diplomatiques. Le processus de chiffrement/déchiffrement était basé sur une rotation de l'alphabet. Cette méthode est connue sous le nom de chiffrement de César. Jusqu'à la fin du XIXème siècle, la cryptologie s'est attachée à résoudre des problèmes liés aux substitutions et aux transpositions mono et poly-alphabétiques, et s'est limitée aux sphères militaires et diplomatiques.

En 1883, dans son article intitulé « La cryptographie militaire », Auguste Kerchhoffs a mis en place certains principes d'une nouvelle cryptographie. Il a mentionné que la sécurité d'un système de chiffrement doit reposer sur la robustesse de la clé et non pas sur le fait que ce système est inconnu de l'ennemi. La période entre les deux guerres mondiales a vu l'avènement d'un âge technique de la cryptographie. Des machines électro-mécaniques, comme Hagelin ou Enigma ont été mises au point et les états-majors les adoptent pour sécuriser les échanges des informations militaires. A l'issue des conflits à cette époque, des travaux sur la théorie de l'information de Claude Shannon et d'Alan Turing sur l'algorithmique et la conception des calculateurs électroniques permettaient de grandes avancées dans le domaine de la cryptologie. Jusqu'aux débuts des années 1970, les méthodes cryptographiques étaient uniquement symétriques, la même clé secrète est utilisée à la fois pour le chiffrement et le déchiffrement. Ce qui pose des problèmes au niveau de l'échange de ces clés secrètes. Une nouvelle cryptographie est née en 1976 avec l'introduction du concept de la clé publique par Whitfield Diffie et Martin Hellman. Leur idée part du principe que seule l'opération de déchiffrement qui doit être protégée par une clé gardée secrète, et le chiffrement peut parfaitement être exécuté à l'aide d'une clé connue publiquement, à condition qu'il soit théoriquement difficile d'en déduire la valeur de la clé secrète. La première mise en œuvre effective de ce principe vient du MIT en 1978, avec la publication par Ronald Rivest, Adi Shamir et Leonard Adleman d'un procédé de chiffrement qui connaît un rapide succès et reçoit le nom RSA.

La sécurité de la plupart des systèmes cryptographiques repose sur la difficulté de résoudre certains problèmes mathématiques. La sécurité du système RSA est basée sur le problème de la factorisation des grands nombres entiers. En 1985, Taher el Gamal a proposé un système cryptographique portant son nom. La sécurité de ce système est basée sur la difficulté du problème du logarithme discret (DLP) dans le groupe multiplicatif d'un corps fini. Dans la même année, Neal Koblitz et Victor Miller ont introduit de façon indépendante un autre système cryptographique issu de la difficulté du DLP dans le groupe des points rationnels d'une courbe elliptique définie sur un corps fini. Quelques années plus tard, Johannes Buchmann et H. C. Williams ont proposé un protocole d'échange de clés de type Diffie-Hellman en utilisant les corps quadratiques imaginaires en 1988. La sécurité de ce protocole repose sur la difficulté du DLP dans le groupe de classes d'un corps quadratique imaginaire. Malheureusement, le système de Buchmann-

Williams restait une construction théorique, il n'avait pas d'applications commerciales à cause du coût élevé de l'exponentiation des idéaux dans les groupes de classes des corps quadratiques imaginaires à partir de certaines tailles de discriminants. En 1997, Peter Shor a proposé un algorithme permettant de résoudre tous les problèmes mathématiques cités ci-dessus en un temps polynômial dans le cas où les calculs sont effectués par un ordinateur quantique. Par conséquent, et dans l'hypothèse de la conception d'un véritable ordinateur quantique dans le futur, tous les systèmes cryptographiques largement utilisés actuellement seront cassés. Pour pallier aux attaques quantiques, les chercheurs en cryptologie étudiaient d'autres problèmes mathématiques qui pourraient résister aux calculs quantiques. En effet, ils ont proposé les systèmes de McEliece et son équivalent dû à Niederreiter dont la sécurité est basée sur des problèmes de la théorie des codes correcteurs, des systèmes cryptographiques basés sur des problèmes issus de la théorie des réseaux euclidiens, et bien d'autres. Parmi les systèmes cryptographiques post-quantiques récemment standardisés on trouve le système NTRU. Ce système a été proposé en 1996 par Jeffrey Hoffstein, Jill Pipher et Joseph H. Silverman. La sécurité du NTRU repose sur le problème du plus court vecteur dans un réseau euclidien.

Cryptographie classique Vs cryptographie moderne

Durant des siècles, le terme cryptographie était synonyme de l'art de la création/résolution des codes secrets, la cryptographie était plus un art qu'une science. Ceci est historiquement exact, mais ne rend pas compte de l'ampleur actuelle du domaine ou de ses fondements scientifiques actuels. De nos jours, la cryptographie englobe bien plus que la confidentialité des communications : elle traite des mécanismes pour garantir l'intégrité, des techniques d'échange de clés secrètes, des protocoles d'authentification des utilisateurs, des enchères, des actions boursières, les élections électroniques ou encore les crypto-monnaies, etc. Sans tenter de fournir une caractérisation complète, nous dirions que la cryptographie moderne implique l'étude des techniques mathématiques pour sécuriser les informations numériques, les systèmes d'informations et les calculs distribués contre les attaques adverses.

Historiquement, les schémas cryptographiques ont été conçus de manière « ad hoc » et évalués en fonction de la complexité, la créativité et l'intelligence de leurs constructions. Un schéma serait analysé pour voir si des attaques pourraient être trouvées ; si c'est le cas, le système serait «corrigé» pour contrecarrer cette attaque, et le processus serait répété. Bien que l'on puisse s'accorder sur le fait que certains systèmes n'étaient pas sûrs, il n'y avait pas de consensus sur les exigences qu'un système «sécurisé» devrait satisfaire, et aucun moyen de prouver qu'un système spécifique était sécurisé.

Au cours des dernières décennies, la cryptographie est devenue, de plus en plus, une science. Les schémas sont désormais développés et analysés de manière plus systématique, le but ultime étant de donner une preuve rigoureuse qu'une construction donnée est sécurisée. Afin d'élaborer de telles preuves, nous avons d'abord besoin de définitions formelles qui définissent exactement ce que signifie «sécurisé» ; ces définitions sont utiles et intéressantes en elles-mêmes. En fait, la plupart des preuves cryptographiques reposent sur des hypothèses actuellement non prouvées sur la difficulté algorithmique de certains problèmes mathématiques (problème de factorisation des entiers, problème de logarithme discret, ...) ; toute hypothèse de ce type doit être explicite et énoncée avec précision. Les définitions, les hypothèses et les preuves distinguent la cryptographie moderne de la cryptographie classique ; nous résumons les trois principes de la cryptographie modernes en :

- Principe 1 : Définitions formelles.
- Principe 2 : Hypothèses précises.
- Principe 3 : Preuves de sécurité.

Ainsi, pour prouver la sécurité d'un protocole cryptographique on doit avoir :

- Une description précise du protocole cryptographique.
- Une description précise de la classe des attaquants.
- Une description précise du modèle de sécurité et les objectifs à atteindre.
- Une description précise des conditions de réussir une attaque.
- Une preuve de sécurité qu'aucun attaquant dans une classe spécifiée ne réussit à casser ce protocole sous les conditions de succès prédéfinies dans le modèle de sécurité.

Alors, il faut montrer qu'aucun attaquant n'a d'avantage «raisonnable» pour casser le protocole cryptographique en un temps «raisonnable». Une approche commune consiste à fonder les preuves de sécurité sur des expériences théoriques, appelées «jeux». On dit qu'un attaquant gagne un jeu s'il parvient à casser la propriété de sécurité qui correspond à ce jeu.

Chapitre 1

Préliminaires Mathématiques

La sécurité de la plupart des primitives cryptographiques repose sur la difficulté de résoudre certains problèmes mathématiques. L'étude des propriétés mathématiques de ces problèmes est primordiale afin d'assurer la sécurité et l'efficacité d'un protocole cryptographique.

1.1 Arithmétique Modulaire

1.1.1 Rappels

Soient a, b, $N \in \mathbb{Z}$ avec $N > 1$. On note $[a \mod N]$ le reste de la division Euclidienne de a par N. L'application $a \mapsto [a \mod N]$ est appelée réduction modulo N. On dit que a et b sont congrus modulo N et on écrit $a = b \mod N$ si $[a \mod N] = [b \mod N]$, et on a :

$a = b \mod N$ si et seulement si $N|(a - b)$. On montre que $a = [b \mod N]$ implique $a = b \mod N$ mais la réciproque est fausse en général.

Exemple :

$36 = 21 \mod 15$ mais $36 \neq [21 \mod 15] = 6$.

La congruence modulo N est une relation d'équivalence compatible avec l'addition, la soustraction et la multiplication modulo N. Si $a = a' \mod N$ et $b = b' \mod N$ alors $(a + b) = (a' + b') \mod N$ et $ab = a'b' \mod N$. Par conséquent, on peut effectuer une réduction suivée d'une addition/multiplicatif au lieu de faire une addition/multiplication suiviée d'une réduction, cela permet de simplifier les calculs.

Exemple :

Calculons $[1093028.190301] \mod 100$. Puisque $[1093028 = 28 \mod 100]$ et $[190301 = 1 \mod 100]$, alors :

$$1093028.190301 = [1093028 = 28 \mod 100][190301 = 1 \mod 100] = 28.1 = 28 \mod 100.$$

Evidemment, l'autre méthode consiste à calculer le produit 1093028.190301, puis réduire le résultat modulo 100, ce qui est moins efficace en particulier lors de l'utilisation des grands nombres.

Proposition 1. *Soit b, $N \in \mathbb{N}^*$, b est inversible modulo N si et seulement si* $\gcd(b, N) = 1$.

Exemple :

Soint $b = 11$, $N = 17$. Alors, $(-3).11 + 2.17 = 1$ (en utilisant l'algorithme d'Euclide étendu). Donc, $14 = [-3 \mod 17]$ est l'inverse de 11 modulo 17. On peut vérifier que $14.11 = 1 \mod 17$.

Théorème 1. *Soit $\mathbb{G}$ un groupe fini d'ordre m. Alors :*

$$\forall g \in \mathbb{G},\ g^m = 1.$$

Corollaire 1. *Soit $\mathbb{G}$ un groupe fini d'ordre $m > 1$. Pour tout $g \in \mathbb{G}$ et pour tout entier x, on a :*

$$g^x = g^{[x \mod m]}.$$

Corollaire 2. *Soit $\mathbb{G}$ un groupe fini d'ordre $m > 1$. Soit $e > 0$ un entier, on définit la fonction : $f_e : \mathbb{G} \longrightarrow \mathbb{G}$ par $f_e(g) = g^e$. Si $\gcd(e, m) = 1$, alors f_e est une permutation (bijection). De plus, si $d = e^{-1} \mod m$, donc f_d est l'inverse de f_e.*

1.1.2 Les groupes $\mathbb{Z}_N$ et $\mathbb{Z}_N^*$

Soit N un entier > 1. L'ensemble $\{0, \cdots, N-1\}$ muni de l'addition modulo N est un groupe abélien d'ordre N, on note ce groupe $\mathbb{Z}_N$. Un élément $b \in \mathbb{Z}_N$ est inversible pour la multiplication modulo N vérifie $\gcd(b, N) = 1$, ce qui permet de donner la définition suivante :

$$\mathbb{Z}_N^* = \{b \in \{0, \cdots, N-1\} | \gcd(b, N) = 1\}.$$

On montre que $\mathbb{Z}_N^*$ est aussi un groupe abélien pour la multiplication modulo N. On définit $\phi(N) =| \mathbb{Z}_N^* |$ l'ordre du groupe $\mathbb{Z}_N^*$ (ϕ est appelée la fonction phi d'Euler).

Le théorème suivant permet de calculer la fonction ϕ.

Théorème 2. *Soit $N = \prod_i p_i^{e_i}$ où les p_i sont des nombres premiers différents, et les e_i sont des entiers ≥ 1. Alors :*

$$\phi(N) = \prod_i p_i^{(e_i - 1)}(p_i - 1).$$

Exemple :
Prenons $N = 15 = 5.3$, alors $\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$ et $| \mathbb{Z}_{15}^* | = 8 = 4.2 = \phi(15)$.

Corollaire 3. *Soit $N > 1$ un entier et $a \in \mathbb{Z}_N^*$, alors :*

$$a^{\phi(N)} = 1 \mod N.$$

Corollaire 4. *Fixons un entier $N > 1$, pour un autre entier $e > 0$, on définit la fonction : $f_e : \mathbb{Z}_N^* \longrightarrow \mathbb{Z}_N^*$ par $f_e(x) = x^e \mod N$.*

Si $\gcd(\phi(N), e) = 1$, alors f_e est une permutation (bijection). De plus, si $d = e^{-1} \mod \phi(N)$, donc f_d est l'inverse de f_e.

1.1.3 Isomorphismes et théorème des restes chinois

Dans la suite, nous énonçons le théorème des restes chinois dans le cas particulier où $N = p.q$

Théorème 3 (Théorème des restes chinois (CRT))**.** *Soient $N = p.q$ où $p, q > 1$ et $\gcd(p, q) = 1$. Alors :*

$$\mathbb{Z}_N \simeq \mathbb{Z}_p \times \mathbb{Z}_q \text{ et } \mathbb{Z}_N^* \simeq \mathbb{Z}_p^* \times \mathbb{Z}_q^*.$$

De plus, l'application $f : \mathbb{Z}_N \longrightarrow \mathbb{Z}_p \times \mathbb{Z}_q$ définit par : $f(x) = ([x \mod p], [x \mod q])$ est un isomorphisme de $\mathbb{Z}_N$ dans $\mathbb{Z}_p \times \mathbb{Z}_q$, et la restriction de f à $\mathbb{Z}_N^*$ est un isomorphisme de $\mathbb{Z}_N^*$ dans $\mathbb{Z}_p^* \times \mathbb{Z}_q^*$.

Les groupes isomorphes donnent différents représentations d'une même structure algébrique. Pourtant, le choix de telle ou telle représentation peut affecter significativement l'efficacité calculatoire des opérations du groupe.

Soient $(\mathbb{G}, \circ_{\mathbb{G}})$ et $(\mathbb{H}, \circ_{\mathbb{H}})$ deux groupes et f un isomorphisme de $\mathbb{G}$ dans $\mathbb{H}$, où f et f^{-1} peuvent être calculées de façon efficace.

Maintenant, pour $g_1, g_2 \in \mathbb{G}$ on peut calculer $g = g_1 \circ g_2$ par deux méthodes : soit caluler g directement en appliquant la loi de $\mathbb{G}$, ou bien par :

— Calculer $h_1 = f(g_1)$ et $h_2 = f(g_2)$.
— Calculer $h = h_1 \circ_{\mathbb{H}} h_2$.
— Calculer $g = f^{-1}(h)$.

Ce qui précède s'étend de manière naturelle lorsque nous voulons calculer plusieurs opérations du groupe $\mathbb{G}$ (par exemple, pour calculer g^x pour un entier x). La meilleure méthode dépend de l'efficacité relative du calcul de l'opération dans chaque groupe, ainsi que de l'efficacité du calcul de f et f^{-1}.

Exemple :

Pour $N = 35 = 5.7$, on cherche à calculer $[18^{25} \mod 35]$. On a la correspondance suivante :

$$18 \leftrightarrow ([18 \mod 5], [18 \mod 7]) = (3, 4)$$

$$[18^{25} \mod 35] \leftrightarrow (3, 4)^{25} = ([3^{25} \mod 5], [4^{25} \mod 7]).$$

Or $\mid \mathbb{Z}_5^* \mid = 4$ et $\mid \mathbb{Z}_7^* \mid = 6$, alors $3^{25} = 3^{[25 \mod 4]} = 3^1 = 3 \mod 5$, et $4^{25} = 4^{[25 \mod 6]} = 4^1 = 4 \mod 7$. Par la suite, $([3^{25} \mod 5], [4^{25} \mod 7]) = (3, 4) \leftrightarrow 18$, donc $[18^{25} \mod 35] = 18$.

1.2 Courbes Elliptiques sur les Corps Finis

Les courbes elliptiques jouent des rôles très importants dans des domaines centraux des mathématiques. En particulier, dans la théorie des nombres, l'algèbre et la géométrie. L'étude des courbes elliptiques remonte au XIX$^{\text{ème}}$ siècle, et durant ces longues années, elles avaient de nombreuses applications théoriques et pratiques. Par exemple, les courbes elliptiques ont joué un rôle fondamental dans la fameuse preuve de Andrew Wiles du grand théorème de Fermat, achevée en 1994. Les premiers liens entre les courbes elliptiques et la cryptographie sont apparus en 1984 lorsque Hendrik Lenstra a découvert son algorithme de factorisation des nombres entiers en se basant sur les courbes elliptiques. Une année plus tard, Neal Koblitz et Victor Miller ont séparément suggéré l'utilisation des courbes algébriques pour construire des systèmes cryptographiques. Cette proposition présentait une variante des cyptosystèmes à clés publiques proposés à l'époque comme le RSA ou le cyptosystème d'EL Gamal et le protocole d'échange de clés de Diffie-Hellman. Plus récemment, d'autres applications cryptographiques des courbes elliptiques utilisant les propriétés des couplages ont été proposées. Par exemples, le schéma tripartite d'échange de clés de Joux en 2000, le schéma de chiffrement basé sur l'identité de Boneh et Franklin ou les schémas de signatures courtes en 2001.

1.2.1 Définition et propriétés des courbes elliptiques

Dans un contexte algébrique plus général, une courbe elliptique est une courbe algébrique (variété projective de dimension 1) non singulière de genre 1. Étant donné que toute courbe elliptique admet une équation de Weierstraß, et que les courbes elliptiques proposées par les organisations internationales de sécurité sont représentées sous cette forme, dans le cadre de ces notes de cours on se contentera de la définition suivante :

Définition 1. Soit $\mathbb{K}$ un corps et $\overline{\mathbb{K}}$ sa clôture algébrique, une courbe elliptique E sur $\mathbb{K}$, notée $E/\mathbb{K}$, est l'ensemble des points $(x, y) \in \overline{\mathbb{K}}^2$ vérifiant l'équation de Weierstraß

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{1.1}$$

avec les coefficients a_1, a_2, a_3, a_4, $a_6 \in \mathbb{K}$, tels que $\Delta \neq 0$ où Δ, le discriminant de la courbe, est défini comme suit :

$$\begin{cases} d_2 = a_1^2 + 4a_2 \\ d_4 = 2a_4 + a_1a_3 \\ d_6 = a_3^2 + 4a_6 \\ d_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2 \\ \Delta = -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6 \end{cases}.$$

L'ensemble $E(\mathbb{K})$ défini par

$$E(\mathbb{K}) = \left\{(x, y) \in \mathbb{K}^2 \,:\, y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\right\} \cup \{\mathcal{O}\}, \tag{1.2}$$

où $\mathcal{O}$ est le point à l'infini, est appelé l'ensemble des points rationnels de la courbe $E/\mathbb{K}$. Les coordonnées (x, y) sont appelées coordonnées affines.

On définit une loi d'addition notée "+" sur $E(\mathbb{K})$ comme suit : si $P = (x_P, y_P)$ et $Q = (x_Q, y_Q)$ deux points de $E(\mathbb{K})$ tels que, $P \neq \mathcal{O}$ et $Q \notin \{\mathcal{O}, -P\}$, alors le point $R = (x_R, y_R) = P + Q$ est donné par les formules suivantes :

$$\begin{cases} x_R = \lambda^2 + a_1\lambda - a_2 - x_P - x_Q \\ y_R = \lambda(x_P - x_R) - y_P - a_1 x_R - a_3 \end{cases} \quad \text{où} \quad \lambda = \begin{cases} \frac{y_P - y_Q}{x_P - x_Q} & \text{si} \quad P \neq Q \\ \frac{3x_P^2 + 2a_2x_1 + a_4 - a_1y_P}{2y_P + a_1x_P + a_3} & \text{si} \quad P = Q \end{cases} . \tag{1.3}$$

L'opposé du point P est le point $-P = (x_P, -y_P - a_1x_P - a_3)$ et l'élément neutre de ce groupe est le point $\mathcal{O}$.

Théorème 4. *$(E(\mathbb{K}), +)$ est un groupe additif abélien.*

Définition 2. Soit $E/\mathbb{K}$ une courbe elliptique définie par $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$, les deux applications : $x \mapsto u^2x' + r$ et $y \mapsto u^3y' + u^2sx' + t$, avec $(u, r, s, t) \in \mathbb{K}^* \times \mathbb{K}^3$ sont inversibles et transforment la courbe E en la courbe

$$E' : y^2 + a'_1xy + a'_3y = x^3 + a'_2x^2 + a'_4x + a'_6, \text{ où } a'_1, a'_2, a'_3, a'_4, a'_6 \in \mathbb{K}.$$

On dit que les deux courbes E et E' sont $\mathbb{K}$-isomorphes.

Si le corps $\mathbb{K}$ est de caractéristique différente de 2 et de 3, et en choisissant $u = 1$, $r = -\frac{a_1^2 + 4a_2}{12}$, $s = -\frac{a_1}{2}$, et $t = \frac{a_1^3 + 4a_1a_2 - 12a_3}{24}$, alors dans ce cas la courbe E est transformée en la courbe

$$E' : y^2 = x^3 + ax + b, \tag{1.4}$$

de discriminant $\Delta = -16(4a^3 + 27b^2)$, pour certains $a, b \in \mathbb{K}$. On associe à une telle courbe l'élément $j(E') = 1728\frac{4a^3}{4a^3 + 27b^2}$, appelé le j-invariant. L'équation 1.4 est appelée l'équation de Weierstraß réduite.

Remarque 1. L'équation de Weierstraß réduite est la forme la plus utilisée pour représenter les courbes elliptiques pour des applications cryptographiques.

Définition 3. Soient E et $\tilde{E}$ deux courbes elliptiques définies sur un corps $\mathbb{K}$. Alors, la courbe $\tilde{E}$ est dite tordue de degré d de la courbe E si les deux courbes E et $\tilde{E}$ sont $\mathbb{L}$-isomorphes, telle que $\mathbb{L}$ est une extension minimale de degré d de $\mathbb{K}$.

1.2.2 Multiplication par un scalaire

Définition 4. Soit $n \in \mathbb{N}^*$, la multiplication d'un point P dans la courbe elliptique E par le scalaire n est l'opération qui consiste à additionner n fois le point P à lui même, qu'on la note $[n]P$ (ou $[n]_EP$), cette opération est définie par :

$$[n]P = \underbrace{P + \cdots + P}_{n \text{ fois}} .$$

La multiplication d'un point P de la courbe elliptique par un scalaire n est l'opération la plus importante dans la cryptographie basée sur les courbes elliptiques, son temps d'exécution domine le temps nécessaire pour l'exécution du protocole cryptographique tout entier. La multiplication par un scalaire est analogue, en quelques sortes, à l'exponentiation modulaire utilisée dans le RSA. Plusieurs méthodes existent pour effectuer cette opération, le choix d'une méthode dépend du niveau de sécurité demandé, ainsi que des capacités de calcul et de mémoire des dispositifs cryptographiques où les primitives cryptographiques seront implantées. Dans la suite, nous rappelons les méthodes les plus utilisées pour faire la multiplication par un scalaire.

Méthodes irrégulières pour la multiplication par un scalaire :

La méthode doublement-et-addition : l'algorithme 1 illustre la méthode "doublement-et-addition". Dans cet algorithme, une opération de doublement est nécessaire à chaque itération, alors qu'une opération d'addition n'est requise que lorsque le bit correspondant du scalaire n est égal à 1. Le coût de cette méthode

Algorithme 1 : La méthode doublement et addition

Entrées : P, $n = (n_{l-1}, n_{l-2}, \cdots, n_0)_2$.
Sorties : $[n]P$.

$Q \leftarrow \mathcal{O}$
pour i ***de*** $l-1$ ***à*** 0 **faire**
  $Q \leftarrow [2]Q$
  si $n_i = 1$ **alors**
    $Q \leftarrow Q + P$
  fin
fin
Retourner Q

est : $\frac{l}{2}$ additions et l doublements [**?**].

La méthode de la forme non-adjacente du scalaire traitée par fenêtres : en tenant compte du fait que le calcul de l'opposé d'un point P dans une courbe elliptique est négligeable, il est possible de réduire le coût de la multiplication scalaire en réduisant le nombre des additions.

Algorithme 2 : La méthode NAF

Entrées : P, $n = (n_{l-1}, n_{l-2}, \cdots, n_0)_{\text{NAF}}$.
Sorties : $[n]P$.

$Q \leftarrow \mathcal{O}$
pour i ***de*** $l-1$ ***à*** 0 **faire**
  $Q \leftarrow [2]Q$
  si $n_i = 1$ **alors**
    $Q \leftarrow Q + P$
  sinon
    si $n_i = -1$ **alors**
      $Q \leftarrow Q - P$
    fin
  fin
fin
Retourner Q

Cette réduction peut être obtenue en représentant le scalaire n sous d'autres formes. Par exemple, la forme non adjacente (NAF) est utilisée pour réduire le nombre des "1" dans la représentation binaire de n. Le coût de cette méthode est : $\frac{l}{3}$ additions et l doublements.

Si on dispose d'un équipement ayant un espace mémoire suffisant, la méthode précédente peut être généralisée par l'algorithme 3. Dans cette nouvelle méthode, le scalaire n est réécrit en utilisant un ensemble de chiffres dans $D = \{-2^{w-1}, \cdots, 2^{w-1}\}$, ce qui revient à le découper en fenêtres de longueur fixe w. Cette méthode nécessite un pré-calcul des points $[i]P$ où $i \in \{1, 3, 5, \cdots, 2^{w-1} - 1\}$, posons $\varphi(w) = \frac{4}{3} - \frac{(-1)^w}{3.2^{w-2}}$,

le coût de cette méthode est : $\frac{l}{w+\varphi(w)} + \frac{2^w - (-1)^w}{3} - 1$ additions et $l + 1$ doublements.

Méthodes régulières pour la multiplication par un scalaire :

On peut facilement voir que les itérations des algorithmes précédents prennent des temps d'exécutions différents pour les différentes valeurs de n_i où $0 \leq i \leq l-1$. Ce fait peut être exploité par un attaquant qui, à partir d'une analyse simple peut déterminer le scalaire secret n. Pour résoudre ce problème, les calculs effectués ne doivent pas dépendre des données secrètes et les conditions doivent être évitées.

La méthode doublement et toujours addition : l'algorithme 4 proposé par Coron est un exemple des méthodes régulières pour faire la multiplication par un scalaire, dans ce cas le nombre des additions et

Algorithme 3 : La méthode w-NAF

Entrées : P, $n = (n_{l-1}, n_{l-2}, \cdots, n_0)_{w\text{-NAF}}$.
Sorties : $[n]P$.

Pré-calculs des $P_i = [i]P$ pour $i \in \{1, 3, 5, \cdots, 2^{w-1} - 1\}$
$Q \leftarrow \mathcal{O}$
pour i ***de*** $l-1$ ***à*** 0 **faire**
  $Q \leftarrow [2]Q$
  si $n_i \neq 0$ **alors**
    si $n_i > 0$ **alors**
      $Q \leftarrow Q + P_{n_i}$
    fin
  sinon
    $Q \leftarrow Q - P_{-n_i}$
  fin
fin
Retourner Q

des doublements est constant.

Algorithme 4 : La méthode doublement et toujours addition

Entrées : P, $n = (n_{l-1}, \cdots, n_1, 1)_2$.
Sorties : $[n]P$.

$Q_0 \leftarrow P$
$Q_1 \leftarrow P$
$Q_2 \leftarrow [2]P$
pour i ***de*** 1 ***à*** $l-1$ **faire**
  $Q_{1-n_i} \leftarrow Q_{1-n_i} + Q_2$
  $Q_2 \leftarrow [2]Q_2$
fin
Retourner Q_0

Le coût de cette méthode est : l additions et l doublements.

La méthode échelle de Montgomery : une idée due à Montgomery est basée sur le fait que la somme de deux points, dont la différence est connue, peut être calculée sans la coordonnée y de ces points. Cette méthode est une variante de la méthode doublement et toujours addition.

Le coût de cette méthode est : l additions et l doublements.

1.2.3 Courbes elliptiques sur les corps finis

Les courbes elliptiques utilisées dans la cryptographie sont des courbes définies sur les corps finis $\mathbb{F}_q$ à q éléments. Dans la suite de ce chapitre, on considère une courbe elliptique E définie sur $\mathbb{F}_q$, avec $q = p^m$ où p est un premier différent de 2 et 3, et $m \in \mathbb{N}^*$.

Arithmétique d'une courbe elliptique sur un corps fini :

Soit E une courbe définie par l'équation de Weierstraß réduite suivante :

$$y^2 = x^3 + ax + b,\ a, b \in \mathbb{F}_q,\ 4a^3 + 27b^2 \neq 0 \text{ dans } \mathbb{F}_q\ . \tag{1.5}$$

Dans ce cas, si on a $P = (x_P, y_P)$ et $Q = (x_Q, y_Q)$ deux points de $E(\mathbb{F}_q)$ tels que, $P \neq \mathcal{O}$ et $Q \notin \{\mathcal{O}, -P\}$, alors le point $R = (x_R, y_R) = P + Q$ est donné par les formules :

Algorithme 5 : La méthode échelle de Montgomery

Entrées : P, $n = (1, n_{l-2}, \cdots, n_0)_2$.
Sorties : $[n]P$.

$Q_0 \leftarrow P$
$Q_1 \leftarrow [2]P$
pour i ***de*** $l-2$ ***à*** 0 **faire**
  $Q_{1-n_i} \leftarrow Q_0 + Q_1$
  $Q_{n_i} \leftarrow [2]Q_{n_i}$
fin
Retourner Q_0

$$x_R = \lambda^2 - x_P - x_Q,\ y_R = \lambda(x_P - x_R) - y_P, \text{ où } \quad \lambda = \begin{cases} \frac{y_P - y_Q}{x_P - x_Q} & \text{si} \quad P \neq Q, \\ \frac{3x_P^2 + a}{2y_P} & \text{si} \quad P = Q \end{cases} . \tag{1.6}$$

L'opposé du point P est le point $-P = (x_P, -y_P)$, l'élément neutre est le point $\mathcal{O}$.

Cardinalité d'une courbe elliptique sur un corps fini :

les courbes elliptiques définies sur un corps fini $\mathbb{F}_q$ ont un cardinal fini noté $\#E(\mathbb{F}_q)$, ce nombre est estimé par le théorème de Weil-Hasse.

Théorème 5 (Weil-Hasse)**.** *Soit E une courbe elliptique définie sur $\mathbb{F}_q$. Alors*

$$q + 1 - 2\sqrt{q} \leqslant \#E(\mathbb{F}_q) \leqslant q + 1 + 2\sqrt{q} \ . \tag{1.7}$$

Le théorème de Weil-Hasse montre que $\#E(\mathbb{F}_q) \approx q$, puisque $2\sqrt{q}$ est relativement petit par rapport à q. La détermination du $\#E(\mathbb{F}_q)$ a une grande importance pour la sécurité des systèmes cryptographiques basés sur les courbes elliptiques. Actuellement, il existe des méthodes rapides et efficaces pour le calcul effectif du nombre de points rationnels d'une courbe elliptique, par exemple l'algorithme SEA, proposé par Schoof et amélioré par Elkies et Atkin, puis par d'autres mathématiciens.

L'intervalle $\left[q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}\right]$ est appelé l'intervalle de Hasse. Une autre formulation du théorème précèdent s'énonce en utilisant la trace de Frobenius. Alors, si E est une courbe elliptique définie sur $\mathbb{F}_q$, donc $\#E(\mathbb{F}_q) = q + 1 - t$ où $|t| \leqslant 2\sqrt{q}$, t est appelée la trace de E sur $\mathbb{F}_q$.

Chapitre 2

Primitives et Algorithmes Cryptographiques

Afin de sécuriser les données sensibles échangées à travers les réseaux publiques ou stockées sur les différents espaces de stockage, les concepteurs des systèmes informatiques utilisent des techniques et des protocoles cryptographiques pour assurer les fonctions de sécurité informatique telles que la confidentialité, l'intégrité, l'authenticité, la disponibilité et la non-répudiation.

Un protocole cryptographique est conçu à partir d'un ou de plusieurs algorithmes cryptographiques dont la sécurité est basée, dans la plupart des cas, sur la difficulté de résoudre certains problèmes mathématiques.

2.1 Notations Générales

Le choix des paramètres d'un protocole cryptographique (tels que les tailles des clés utilisées et les groupes mathématiques sous-jacents) se fait à partir d'un paramètre de sécurité fixé. Notons ce paramètre par λ.

Définition 5 (Fonction Négligeable)**.** On dit qu'une fonction $\varepsilon : \mathbb{N} \to \mathbb{R}$ est négligeable si pour tout polynôme $P : \mathbb{N} \to \mathbb{R}^+$ il existe $N \in \mathbb{N}$ tel que $\forall \lambda \geq N \ \varepsilon(\lambda) \leq \frac{1}{P(\lambda)}$, on note $negl(\lambda)$.

Définition 6. On dit qu'une fonction est calculable efficacement si elle peut être calculée en un temps polynomial.

Pour un algorithme $\mathcal{A}$, l'expression $y \leftarrow \mathcal{A}(x)$ signifie que $\mathcal{A}$ prend en entrée la valeur x et donne la valeur y en sortie. Aussi, si l'algorithme $\mathcal{A}$ a un accès à un oracle $\mathfrak{o}$ on note $\mathcal{A}^{\mathfrak{o}}$. Soit S un ensemble, on utilise la notation suivante $s \leftarrow_R S$ pour dire que la valeur s est choisie de façon aléatoire de l'ensemble S.

2.2 Systèmes Cryptographiques Symétriques

Dans un système cryptographique symétrique, les parties communicantes (personnes ou entités) partagent les mêmes clés qui doivent rester secrètes.

2.2.1 Chiffrement symétrique et problème de distribution des clés

Les systèmes de chiffrement symétrique utilisent une seule clé k pour assurer la confidentialité des messages échangés entre deux parties Alice et Bob. Cette clé sert à la fois pour chiffrer le message clair m et pour déchiffrer le message chiffré c.

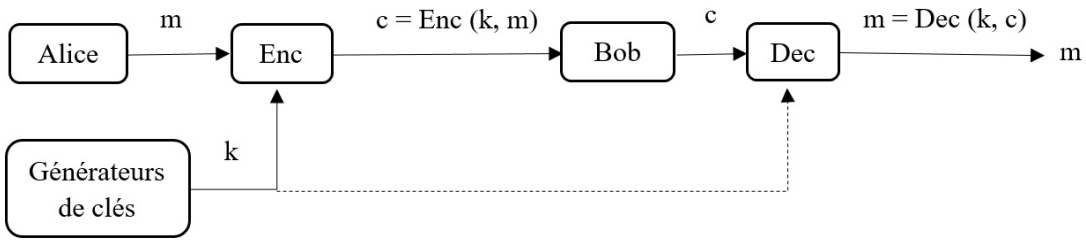

FIGURE 2.1 – Schéma général d'un cryptosystème symétrique de clé k.

La clé k est secrète, et elle ne doit être connue que par les deux interlocuteurs Alice et Bob. En conséquence, chaque couple de correspondants a besoin d'une clé k différente des autres clés, ce qui pose un problème au niveau de la gestion des clés. Par exemple, pour n parties on a besoin de créer et attribuer $\frac{n(n-1)}{2}$ clés secrètes distinctes. Un autre problème pour la mise en œuvre de ces systèmes réside dans la difficulté d'échanger ces clés. Cependant, les systèmes de chiffrement symétriques sont indispensables, ils comportent des avantages majeurs : ils sont très rapides et particulièrement adaptés au chiffrement des grandes quantités de données. Formellement, on a :

Définition 7 (Système de Chiffrement Symétrique)**.** Un système de chiffrement symétrique est la donnée de trois algorithmes polynomiaux $\mathcal{E} = (Gen, Enc, Dec)$ définis comme suit :

- **Génération des clés** : un algorithme probabiliste Gen génère une clé secrète k à partir d'un paramètre de sécurité λ.
- **Chiffrement** : un algorithme probabiliste Enc prend en entrée un message m et la clé secrète k, et produit un texte chiffré c.
- **Déchiffrement** : un algorithme déterministe Dec prend en entrée le texte chiffré c et la clé secrète k, et retourne le message clair m.

Un système de chiffrement symétrique doit vérifier la condition suivante :

Pour tout paramètre de sécurité λ, tout $k \leftarrow Gen(\lambda)$, pour tout message m et tout message chiffré $c \leftarrow Enc(k, m)$, nous avons $Dec(k, c) = m$.

2.2.2 Le système de chiffrement AES

L'AES est un système de chiffrement symétrique largement utilisé actuellement. Ce système est recommandé par plusieurs organisations de sécurité comme standard du chiffrement symétrique. En outre, AES est considéré parmi les systèmes cryptographiques post-quantiques.

L'AES opère sur des blocs de 128 bits (16 octets), il les transforme en blocs chiffrés de 128 bits au moyen d'une clé de taille 128 bits (16 octets), 192 bits (24 octets) ou 256 bits (32 octets). Le fonctionnement de ce système consiste à découper les données à chiffrer et les clés à utiliser en octets sous forme de matrices. La matrice donnée est de taille 4×4, elle comporte $t_m = 16$ octets notés m_0, $m_1, \ldots, m_{15}$. La matrice clé est une matrice de taille $4 \times N_k$ avec ($N_k = 4$, $N_k = 6$, ou $N_k = 8$) et elle contient ($t_k = 16$, $t_k = 24$, ou $t_k = 32$) octets notés k_0, k_1, …, k_{t_k-1}. Par exemple pour l'AES-128, on a $t_k = 16$ et on obtient :

$$\begin{pmatrix} m_0 & m_4 & m_8 & m_{12} \\ m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \end{pmatrix} \text{ et } \begin{pmatrix} k_0 & k_4 & k_8 & k_{12} \\ k_1 & k_5 & k_9 & k_{13} \\ k_2 & k_6 & k_{10} & k_{14} \\ k_3 & k_7 & k_{11} & k_{15} \end{pmatrix}$$

L'AES est exécuté en un nombre fini de tours noté n_r. Ce nombre de tours dépend de la taille de la clé utilisée. On a $n_r = 10$ (resp. 12 et 14) pour 128 bits (resp. 192 et 256 bits). Chaque tour se compose de quatre opérations *SubBytes*, *ShiftRows*, *MixColumns*, *AddRoundKey*, à l'exception du dernier tour qui ignore l'opération *MixColumns*. En outre, il existe une opération initiale *AddRoundKey* avant le premier tour.

L'algorithme 6 représente le fonctionnement de l'AES-128, et c'est cette version que nous utiliserons dans les chapitres suivants.

l'AES combine des opérations de substitution et de permutation. Ces opérations sont définies comme suit :

- *SubBytes* : une fonction de substitution non linéaire qui consiste à substituer chaque octet de la matrice d'entrée par un autre octet du tableau S-Box (2.1).
- *ShiftRows* : un décalage circulaire (vers la gauche) de i positions pour la $i^{\text{ème}}$ ligne, avec $0 \leq i \leq 3$.
- *MixColumns* : une transformation linéaire inversible qui combine les quatre octets de chaque colonne de la matrice d'entrée pour produire quatre autres octets.
- *AddRoundKey* : une opération qui consiste à faire un "*ou exclusif*" (XOR) entre les octets de la matrice d'entrée et les octets de la matrice clé de tour k_i extraites de la clé initiale k.

Algorithme 6 : L'algorithme AES - 128

Entrées : Le texte clair $m \in \{0,1\}^{128}$, la clé $k \in \{0,1\}^{128}$.
Sorties : Le texte chiffré $c \in \{0,1\}^{128}$.

état $s \leftarrow m$
$s \leftarrow \text{AddRoundkey}(s, k)$
pour $i = 1 \cdots 9$ **faire**
    $s \leftarrow \text{SubBytes}(s)$
    $s \leftarrow \text{ShiftRows}(s)$
    $s \leftarrow \text{MixColumns}(s)$
    $s \leftarrow \text{AddRoundkey}(s, k_i)$
fin
$s \leftarrow \text{SubBytes}(s)$
$s \leftarrow \text{ShiftRows}(s)$
$s \leftarrow \text{AddRoundkey}(s, k_{10})$
$c \leftarrow s$
Retourner c

2.2.3 Authentification et intégrité de messages

Assurer l'intégrité des données consiste à éviter (à détecter au moins) que les données transmises sont altérées par un attaquant. Généralement, cette fonctionnalité est assurée par un code d'authentification de messages (MAC) ou par une signature numérique. Un autre type d'authentification des parties intervenantes dans un système cryptographique est défini comme un procédé par lequel une personne ou une entité se convainc de l'identité de son partenaire de communication, une telle authentification implique une authentification des données et par la suite leur intégrité.

Dans la pratique, l'utilisation principale des codes MAC est d'assurer l'intégrité d'un message m. Le principe est de calculer un code MAC (un tag) à partir du message à protéger m et d'une clé secrète k partagée avec le destinataire, ce code est ajouté au message et transmis. Si l'entité réceptrice connaît la clé secrète k, elle peut alors recalculer le MAC afin de vérifier que le message envoyé et le message reçu sont effectivement identiques. De façon plus formelle on a :

Définition 8 (Code d'Authentification de Messages)**.** Un schéma d'un code d'authentification de messages $\mathcal{M} = (\text{MKGen}, \text{MAC}, \text{MVf})$ est constitué des trois algorithmes suivants :

- **La génération des clés** : un algorithme probabiliste MKGen génère une clé secrète k à partir d'un paramètre de sécurité λ.
- **La génération du tag** : un algorithme probabiliste MAC prend en entrée un message m et la clé secrète k et produit un tag T.
- **La vérification** : un algorithme déterministe MVf prend en entrée le message reçu m, la clé secrète k, le tag T, et retourne la valeur 1 si la vérification est valide, et 0 sinon.

Un code d'authentification de messages $\mathcal{M}$ doit vérifier la condition suivante :

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	BE	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	BE	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

TABLE 2.1 – La représentation hexadécimale du S-Box utilisé par l'opération SubBytes dans l'AES

Pour tout paramètre de sécurité λ, pour tout message m et tout $k \leftarrow_R \text{MKGen}(\lambda)$, nous devons avoir $\text{MVf}(m, k, T) = 1$ pour la valeur $T \leftarrow \text{MAC}(m, k)$.

2.3 Systèmes Cryptographiques Asymétriques

L'idée cruciale derrière le concept du système cryptographique asymétrique est que, chaque utilisateur A possède une paire de clés, une clé privée n'étant connue que de A et une autre clé publique connue de tous. Donc, il n'est pas nécessaire de partager des secrets à l'avance.

2.3.1 Échange de clés

Le protocole de Diffie-Hellman :

Le protocole d'échange de clés de Diffie et Hellman permet à deux parties Alice et Bob d'échanger une clé cryptographique en se mettant d'accord sur un générateur g du groupe multiplicatif $G = \mathbb{Z}_p^*$, où p est un premier de taille 2048 bits. Ces choix sont publiques et connus de tout le monde. Alors :

— Alice (resp. Bob) choisit aléatoirement un entier a (resp. b) $\leftarrow_R [1, p-2]$.
— Alice (resp. Bob) calcule $A = g^a$ (resp. $B = g^b$).
— Alice (resp. Bob) envoie le point A (resp. B) à Bob (resp. Alice).
— Alice (resp. Bob) calcule $B^a = g^{ab}$ (resp. $A^b = g^{ab}$), ils obtiennent tous les deux la clé commune g^{ab}.

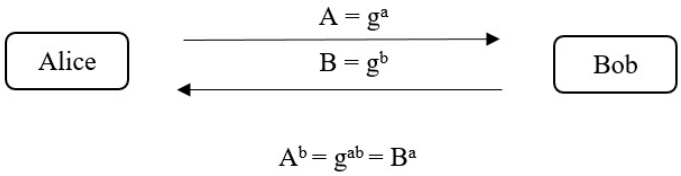

FIGURE 2.2 – Le protocole d'échange de clés de Diffie-Hellman.

2.3.2 Chiffrement asymétrique

Dans un système de chiffrement à clé publique chaque utilisateur possède une paire de clés (e, d), où e est appelée la clé publique et d est dite la clé privée. Les clés publiques de tous les utilisateurs sont répertoriées dans un annuaire publique, elles sont utilisées pour chiffrer les messages. Chaque utilisateur connaît sa propre clé privée (secrète) qui lui permet de déchiffrer ses messages chiffrés. Ainsi, si Alice veut envoyer un message m à Bob, elle obtient la clé publique de Bob e_B de l'annuaire et elle l'utilise ensuite pour chiffrer son message. En recevant le message chiffré c, Bob utilise sa clé privée d_B pour récupérer le message d'Alice.

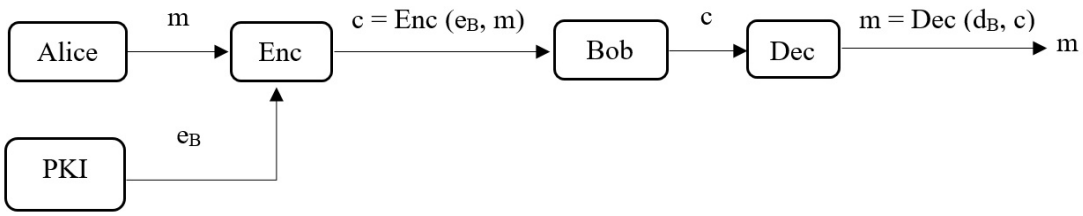

FIGURE 2.3 – Schéma général d'un cryptosystème asymétrique de clés (e, d).

Formellement, on a :

Définition 9 (Système de Chiffrement Asymétrique)**.** Un système de chiffrement asymétrique est la donnée de trois algorithmes polynomiaux $\mathcal{E} = (Gen, Enc, Dec)$ définis comme suit :

- **Génération des clés :** à partir du paramètre de sécurité λ, un algorithme probabiliste Gen génère une paire de clés (d, e), où d (resp. e) désigne la clé privée de déchiffrement (resp. la clé publique de chiffrement).
- **Chiffrement :** à partir d'un message m et de la clé publique e, l'algorithme probabiliste Enc produit un texte chiffré c.
- **Déchiffrement :** à partir d'un texte chiffré c et de la clé privée d, l'algorithme déterministe Dec fournit un message m.

Un système de chiffrement asymétrique doit vérifier la condition suivante :

Pour tout paramètre de sécurité λ, tout couple de clés $(e, d) \leftarrow Gen(\lambda)$, pour tout message m et tout message chiffré $c \leftarrow Enc(e, m)$, nous avons $Dec(d, c) = m$.

2.3.3 Les systèmes de chiffrement RSA et ElGamal

Le système RSA :

Le système de chiffrement RSA permet à Alice d'envoyer un message secret m à Bob, ce système fonctionne de la façon suivante :

- Bob choisit deux grands nombres premiers distincts p et q tels que ($p \approx q \approx 1024$ bits), il calcule ensuite $N = pq$ et $\phi(N) = (p-1)(q-1)$ où ϕ est la fonction d'Euler. Bob choisit ensuite un entier e tel que $pgcd(e, \phi(N)) = 1$ et $e \geq 65537$ (afin d'éviter les attaques de Coppersmith), le couple (e, N) représente la clé publique de Bob utilisé pour le chiffrement. Il existe une autre clé d telle que $ed \equiv 1 \pmod{\phi(N)}$. Cela découle de l'algorithme d'Euclide pour trouver le $pgcd(e, \phi(N))$. d est une clé privée gardée secrètement par Bob, elle est utilisée pour le déchiffrement du message envoyé par Alice.
- Alice calcule le message chiffré c par $c \equiv m^e \pmod{N}$. elle envoie c à Bob.
- A la réception du message chiffré c, Bob retrouve le message clair m en calculant $m \equiv c^d \pmod{N}$.

Le système d'ElGamal :

En 1985, Taher ElGamal a proposé un système de chiffrement à clé publique basé sur le problème du logarithme discret dans le groupe multiplicatif d'un corps fini. Le système cryptographique d'ElGamal, permet à Alice d'envoyer un message secret m à Bob en toute sécurité sans aucune communication préalable.

Alors :

— Bob choisit un grand nombre premier p ($\approx$ 2048 bits) et un générateur g du groupe multiplicatif $G = \mathbb{Z}_p^*$, il choisit de façon aléatoire un entier $a \leftarrow_R [2, p-2]$ et calcule $b \equiv g^a \pmod{p}$. Le triplet (p, g, b) représente la clé publique de Bob, alors que a représente sa clé privée.
— Alice choisit de façon aléatoire un entier $k \leftarrow_R [2, p-2]$, elle calcule $x \equiv g^k \pmod{p}$ et $y \equiv mb^k \pmod{p}$, son message chiffré est $c = (x, y)$.
— A la réception du chiffré c, Bob récupère le message m en utilisant sa clé privée a par le calcul de $m \equiv x^{-a}y \pmod{p}$.

On peut vérifier aisément que :

$$x^{-a}y = (g^k)^{-a}mb^k \equiv (g^k)^{-a}m(g^a)^k \equiv m \pmod{p}.$$

2.3.4 Signature cryptographique

La signature cryptographique (numérique) est un mécanisme très important pour assurer l'authenticité et l'intégrité des informations échangées sans partage préalable de clés secrètes. Elle dépend du message et de l'identité du signataire. Un schéma de signature est composé d'un algorithme de signature et d'un algorithme de vérification. L'algorithme de signature est paramétré par une clé secrète propre au signataire, alors que l'algorithme vérification utilise la clé publique du signataire pour vérifier la validité de la signature. Les signatures cryptographiques sont des outils très importants dans la conception des protocoles d'échange de clés où les différentes parties (dans une session) ont besoin de s'assurer de la légitimité de leurs partenaires. Ainsi, la signature assure aussi la non-répudiation.

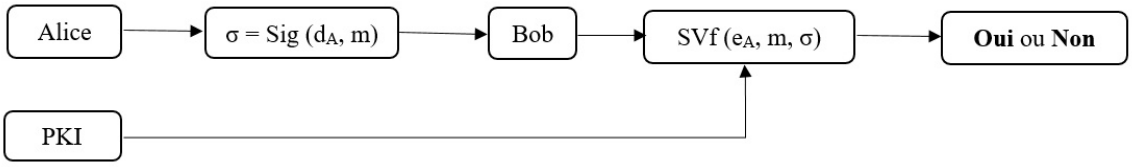

FIGURE 2.4 – Schéma d'une signature cryptographique.

Formellement, on a :

Définition 10 (Signature Cryptographique)**.** Un schéma de signature cryptographique est la donnée de trois algorithmes polynomiaux $\mathcal{S} = (Gen, \text{Sig}, \text{SVf})$ définis par :

- **Génération des clés :** à partir du paramètre de sécurité λ, un algorithme probabiliste Gen génère une paire de clés (d, e), où d (resp. e) désigne la clé secrète de signature (resp. la clé publique de vérification).
- **Signature :** à partir de la clé de signature d et d'un message m un algorithme probabiliste Sig produit une signature σ.
- **Vérification :** à partir de la clé de vérification e, du message m, et de la signature σ un algorithme déterministe SVf retourne 1 si la signature est valide, et 0 sinon.

Une signature valable doit vérifier la condition suivante :

Pour tout paramètre de sécurité λ, tout couple de clés $(e, d) \leftarrow Gen(\lambda)$, pour tout message m et toute signature $\sigma \leftarrow \text{Sig}(d, m)$, nous avons $\text{SVf}(e, m, \sigma) = 1$.

2.3.5 Le standard DSS de signature

Le NIST a défini le standard de la signature cryptographique dans la note FIPS 186-4. Dans ce standard, trois algorithmes sont recommandés : le RSA, le DSA et l'ECDSA. Nous décrivons les algorithmes de signature RSA et DSA dans ce chapitre, et l'algorithme ECDSA dans le chapitre 4.

Supposons que Alice veut envoyer un message signé à Bob.

La signature RSA :

On reprend les notations utilisées dans le paragraphe 2.3.3, soient (N, e_A) la clé publique d'Alice et d_A sa clé secrète.

- Alice signe son message m en utilisant sa clé privée d_A en calculant $\sigma \equiv m^{d_A} \pmod N$.
- Bob reçoit le message signé (m, σ) d'Alice, il calcule $m' \equiv \sigma^{e_A} \pmod N$. Si $m' = m$, alors la signature est valide.

La signature DSA :

Soient p et q deux nombres premiers de tailles respectivement 3072 bits et 256 bits tels que $q/p - 1$ et $g \in \mathbb{Z}_p^*$ d'ordre q.

- Alice fixe les paramètres (p, q, g) et choisit de façon aléatoire $d_A \leftarrow_R \mathbb{Z}_q$. Elle calcule ensuite $e_A \equiv g^{d_A} \pmod p$. d_A représente la clé privée d'Alice et e_A sa clé publique.
- Alice utilise sa clé privée d_A pour signer son message m, elle choisit un entier $k \leftarrow_R [1, q-1]$ et calcule $r \equiv (g^k \pmod p) \pmod q$, $z = H(m)$ où H est une fonction de hachage, $s \equiv k^{-1}(z + d_A r) \pmod q$. $\sigma = (r, s)$ représente la signature associée au message m.
- Bob utilise la clé publique d'Alice pour vérifier la validité de sa signature. D'abord, il vérifie que $0 < r < q$ et $0 < s < q$. Après, Bob calcule $w \equiv s^{-1} \pmod q$, $z = H(m)$, $u_1 \equiv wz \pmod q$, $u_2 \equiv wr \pmod q$ et $v \equiv (g^{u_1} e_A^{u_2} \pmod p) \pmod q$. Si $v = r$, alors la signature est valide, sinon le message est rejeté.

2.3.6 Certificats électroniques et infrastructures de gestion des clés

L'utilisation de la cryptographie asymétrique nécessite la publication des clés publiques en toute sécurité. En d'autres termes, un utilisateur d'un système cryptographique à clé publique doit s'assurer qu'une clé est bien associée à l'identité de son détenteur légitime. Cette assurance est obtenue à partir d'un certificat délivré et signé par une autorité de certification. Ainsi, un certificat d'une clé formalise le lien entre l'identité d'un utilisateur et sa clé. Le même problème persiste dans la cryptographie symétrique sous la forme de la distribution ou la mise en accord des clés secrètes.

L'ensemble des ressources humaines, matérielles et logicielles déployées pour gérer les clés et leurs certificats est appelé infrastructure de gestion des clés. Cette gestion comporte l'enregistrement des demandes de certifications, la génération, la publication ou encore la révocation des certificats.

2.4 Fonctions de Hachage et Dérivation des Clés

2.4.1 Fonctions de hachage

Les fonctions de hachage sont des fonctions qui transforment une suite de bits de longueur arbitraire en une suite de bits de taille fixe. Ces fonctions jouent un rôle très important dans la conception et l'analyse de sécurité de tout système cryptographique. Formellement, on a :

Définition 11 (Fonction de Hachage)**.** Une fonction $H : \{0,1\}^m \mapsto \{0,1\}^n$ est dite fonction de hachage si les conditions suivantes sont remplies :

1. H est efficacement calculable, et
2. $m > n$.

Une fonction de hachage doit posséder les propriétés suivantes :

— Résistance au **calcul d'une pré-image** : étant donné $y \in \{0,1\}^n$, il impossible de calculer pratiquement et en un temps raisonnable, un $x \in \{0,1\}^m$ tel que $H(x) = y$.

— Résistance au **calcul d'une deuxième pré-image** : étant donné $x \in \{0,1\}^m$, il impossible de calculer pratiquement et en un temps raisonnable, un autre $x' \in \{0,1\}^m$ tel que $H(x) = H(x')$.

— Résistance aux **collisions** : il impossible de trouver pratiquement et en un temps raisonnable, deux valeurs x, $x' \in \{0,1\}^m$ tels que $H(x) = H(x')$ et $x \neq x'$.

2.4.2 Fonctions de dérivation de clés

Les fonctions de dérivation de clés (KDF) sont des fonctions permettant d'extraire une ou plusieurs clés à partir d'une valeur secrète telle que la clé obtenue après un échange de type Diffie-Hellman, d'un générateur de nombres aléatoires, d'une clé master, etc. En pratique, les fonctions de dérivation de clés sont construites à partir des fonctions de hachage (par exemple, les fonctions X9.63-KDF) ou en utilisant des fonctions pseudo-aléatoires (comme dans le cas des fonctions NIST-800-108-KDF).

2.5 Générateurs de Nombres (pseudo-)Aléatoires

La génération de nombres aléatoires et les fonctions à sens unique sont deux éléments fondamentaux dans la conception des systèmes cryptographiques et dans l'analyse de leurs sécurités. Il existe plusieurs sources de l'aléa dans la nature comme les phénomènes physiques (quantique, radioactivité, bruit thermique, etc.) qui présentent de vrais aléas. Cependant, dans les applications informatiques il est possible de générer du pseudo-aléa en utilisant des générateurs pseudo-aléatoires. Ces générateurs permettent de produire des suites de nombres aléatoires ayant de bonnes propriétés statistiques de répartitions et d'imprévisibilité à partir d'un nombre initial appelé germe. La génération des nombres pseudo-aléatoires est effectuée en utilisant une suite récurrente dont le germe est le premier terme de cette suite.

2.6 Tailles Recommandées pour les Clés Cryptographiques

Le tableau 2.2 présente les tailles minimales des clés cryptographiques recommandées par le NIST, l'ANSSI et le BSI.

Ce tableau montre que la cryptographie basée sur les courbes elliptiques (ECC) assure le même niveau de sécurité (128 bits) en utilisant seulement des clés cryptographiques de tailles 256 bits au lieu de 2048 ou 3072 bits pour le RSA ou le DLP dans $\mathbb{F}_p^*$, ce qui rend les systèmes cryptographiques basés sur les courbes elliptiques plus efficaces, en particulier dans le cas des environnements embarqués.

Années	Symétrique	RSA	$\mathbb{Z}_p^*$	ECC	Hachage
2017-2022	128	2048	2048	256	SHA-256
2022-2030	128	3072	3072	256	SHA-256

TABLE 2.2 – Estimations des paramètres de sécurité en bits pour les années 2017-2030

Chapitre 3

Cryptographie Basée sur les Courbes Elliptiques

De nos jours, la cryptographie basée sur les courbes elliptiques (ECC) est une technologie très efficace pour la mise en œuvre des systèmes cryptographiques à clés publiques et à la conception des infrastructures de gestion des clés publiques (PKI), elle offre des solutions de chiffrement/déchiffrement, de signature numérique et d'échange de clés. L'ECC est largement utilisée dans les systèmes cryptographiques actuels, en particulier, dans les systèmes embarqués vue les avantages qu'elle présente en termes de sécurité et d'efficacité comparée à d'autres systèmes cryptographiques asymétriques tel que le RSA. Par exemple, les courbes elliptiques sont utilisées dans les protocoles TLS et SSH pour sécuriser la navigation sur Internet, ou bien dans des protocoles comme PACE pour sécuriser les cartes d'identités et les passeports électroniques.

En général, deux grandes familles de corps finis sont utilisées pour définir les courbes elliptiques destinées à des applications cryptographiques. Notamment, les corps premiers $\mathbb{F}_p$ et les corps binaires $\mathbb{F}_{2^m}$. Pour des raisons de sécurité, discutées dans les chapitres suivants, on s'intéresse de plus aux courbes elliptiques définies sur des corps premiers $\mathbb{F}_p$, où p est un nombre premier assez grand.

3.1 Le Problème du Logarithme Discret dans les Courbes Elliptiques (ECDLP)

La sécurité des systèmes cryptographiques basés sur les courbes elliptiques repose essentiellement sur la difficulté du problème du logarithme discret dans une courbe elliptique bien choisie.

Définition 12. Soit E une courbe elliptique sur un corps fini $\mathbb{F}_q$, $P \in E(\mathbb{F}_q)$ un point d'ordre n, et $Q \in < P >$ un point dans le sous-groupe de $E(\mathbb{F}_p)$ engendrée par P. Le problème du logarithme discret dans cette courbe consiste à trouver $k \in [1, n-1]$ tel que $Q = [k]P$, et on note $k = \log_P Q$.

3.2 Le Problème de Diffie-Hellman dans une Courbe Elliptique

En fait, Il existe deux types de problèmes dits de Diffie-Hellman dans une courbe elliptique, l'un est calculatoire et l'autre est décisionnel. Le premier est noté EC-CDH et s'énonce comme suit :

Définition 13 (EC-CDH)**.** Le problème calculatoire de Diffie-Hellman consiste à calculer le point $[ab]P$ étant donné les deux points $[a]P$ et $[b]P \in < P >$.

Si on arrive à résoudre l'ECDLP, clairement on peut résoudre le EC-CDH. La réciproque en général n'est pas connue.

Le problème décisionnel de Diffie-Hellman dans une courbe elliptique est noté EC-DDH, il est défini par :

Définition 14 (EC-DDH)**.** Etant donné trois points $[a]P$, $[b]P$, et $R \in < P >$, le problème décisionnel de Diffie-Hellman consiste à décider si $R = [ab]P$.

Si on parvient à résoudre l'EC-CDH, alors on peut résoudre l'EC-DDH. Dans l'autre sens on ne peut rien dire.

3.3 Protocoles Cryptographiques Basés sur les Courbes Elliptiques

3.3.1 Le protocole d'échange de clés ECDHE

Généralement, les systèmes cryptographiques à clés secrètes sont utilisés pour chiffrer les grandes quantités de données grâce à leur rapidité comparée aux systèmes à clés publiques. Cependant, ces systèmes souffrent du problème d'échange des clés secrètes. Pour cette raison, les systèmes cryptographiques à clés publiques sont souvent utilisés pour échanger les clés secrètes utilisées ultérieurement dans les systèmes à clés secrètes.

Nous décrivons ici le protocole d'échange de clés de Diffie-Hellman basé sur les courbes elliptiques (ECDHE). Dans ce protocole, deux parties Alice et Bob se mettent d'accord sur une courbe elliptique E et un point de base $P \in E$ d'ordre n. Ces choix sont publiques et connus de tout le monde. Alors :

— Alice (resp. Bob) choisit aléatoirement un entier a (resp. b) $\in [1, n-1]$.
— Alice (resp. Bob) calcule $A = [a]P$ (resp. $B = [b]P$).
— Alice (resp. Bob) envoie le point A (resp. B) à Bob (resp. Alice).
— Alice (resp. Bob) calcule $[a]B = [ab]P$ (resp. $[b]A = [ab]P$), ils obtiennent tous les deux la clé $[ab]P$.

Oscar est un attaquant passif qui observe le canal de communication, il voit passer $[a]P$ et $[b]P$ mais il n'arrive pas à reconstituer $[ab]P$ compte tenu de la difficulté du problème de Diffie-Hellman.

Cependant, si Oscar est un attaquant actif, il peut mener une attaque dite *homme au milieu*. Dans cette attaque, Oscar intercepte les échanges entre Alice et Bob, il établit une clé commune avec Alice et une autre avec Bob. Les deux utilisateurs Alice et Bob pensent communiquer entre eux, alors que chacun d'entre eux communique avec Oscar.

Le fait que l'ECDHE ne permet pas d'authentifier les auteurs des messages émis représente une faiblesse de ce protocole, alors une authentification mutuelle des correspondants est nécessaire.

3.3.2 Le protocole de chiffrement d'ElGamal

Nous présentons par la suite la version courbe elliptique du système de chiffrement d'ElGamal. Dans ce protocole :

— Alice et Bob se mettent d'accord sur un corps fini $\mathbb{F}_p$, une courbe elliptique $E/\mathbb{F}_p$, et un point $P \in E(\mathbb{F}_p)$.
— Bob choisit de façon aléatoire un entier secret a et calcule le point $A = [a]P \in E(\mathbb{F}_p)$.
— Bob publie le point A comme clé publique propre, et garde a comme clé privée.
— Alice choisit son message clair $M \in E(\mathbb{F}_p)$ et un entier aléatoire k. Ensuite, elle calcule les deux points suivants : $B_1 = [k]P \in E(\mathbb{F}_p)$ et $B_2 = M + [k]A \in E(\mathbb{F}_p)$.
— Alice envoie son message chiffré (B_1, B_2) à Bob à travers un canal non sécurisé.
— Bob utilise sa clé privée pour calculer $B_2 - [a]B_1 \in E(\mathbb{F}_p)$, ainsi, il peut récupérer le message d'Alice M.

On peut vérifier facilement que :

$$B_2 - [a]B_1 = (M + [k]A) - [a][k]P = M + [k][a]P - [a][k]P = M.$$

3.3.3 Le protocole de signature numérique ECDSA

Le mécanisme de signature permet à Alice d'utiliser sa clé privée pour signer un document numérique m, de telle manière que Bob puisse utiliser la clé publique d'Alice pour vérifier la validité de la signature.

Dans la suite, nous décrivons le protocole de signature numérique ECDSA.

— Alice et Bob se mettent d'accord sur un corps fini $\mathbb{F}_p$, une courbe elliptique $E/\mathbb{F}_p$, et un point $P \in E(\mathbb{F}_p)$ d'ordre premier n.
— Alice choisit de façon aléatoire un entier secret a et calcule le point $A = [a]P \in E(\mathbb{F}_p)$.
— Alice publie le point A, cette clé publique sert à vérifier la validité de la signature, la clé privée a est utilisée pour signer un document m.
— Alice choisit le document numérique $m \pmod n$ à signer. Elle choisit aléatoirement un entier $k \pmod n$ et calcule $[k]P$, $s_1 \equiv x_{[k]P} \pmod n$ et $s_2 \equiv (m + as_1)k^{-1} \pmod n$. Alice envoie le document m et avec la signature (s_1, s_2).
— Bob calcule $v_1 \equiv ms_2^{-1} \pmod n$ et $v_2 \equiv s_1 s_2^{-1} \pmod n$. Ensuite, il calcule $[v_1]P + [v_2]A$ et vérifie si $x_{[v_1]P+[v_2]A} \equiv s_1 \pmod n$.

On peut vérifier que :

$$\begin{aligned}[v_1]P + [v_2]A &= [ms_2^{-1}]P + [s_1 s_2^{-1}][a]P \\ &= [s_2^{-1}(m + as_1)]P \\ &= [k]P\end{aligned}$$

d'où $x_{[v_1]P+[v_2]A} = x_{[k]P} \equiv s_1 \pmod n$.

3.4 Encodage et Compression des Points d'une Courbe Elliptique.

En général, les algorithmes cryptographiques impliquent des opérations utilisant plusieurs types de données. Ces types de données sont convertis en chaînes de bits ou d'octets selon l'architecture du dispositif cryptographique où le protocole cryptographique serait implanté.

Par la suite, on définit quatre fonctions de conversion : la fonction I2OS qui permet de convertir un entier positif en une chaîne d'octets et son inverse OS2I, la fonction FE2OS permettant de convertir un élément d'un corps fini en une chaîne d'octets et son inverse OS2FE.

Soient x un entier positif, la fonction I2OS permet de convertir x en une chaîne de l octets, où $256^l > x$, sinon I2OS renvoie un message d'erreur.

L'idée est d'écrire x sous la forme :

$$x = x_{l-1}.256^{l-1} + x_{l-2}.256^{l-2} + \cdots + x_1.256 + x_0, \quad 0 \leq x_i < 256 \quad \text{pour} \quad 0 \leq i \leq l-1.$$

Alors, on note la chaîne des octects par :

$$X = X_{l-1}X_{l-2}\cdots X_1X_0, \quad \text{où} \quad X_i = x_i \quad \text{pour} \quad 0 \leq i \leq l-1.$$

Maintenant, soit $x \in \mathbb{F}_p$, cet élément est représenté comme un entier positif $x \in \{0, 1, \cdots, p-1\}$, en utilisant la fonction I2OS on définit la fonction FE2OS qui permet de convertir x en une chaîne d'octets de longueur $l = \lceil \log_{256} p \rceil$, avec FE2OS(x)=I2OS$(x, l)$.

Inversement, une chaîne d'octets X est convertie en un élément du corps fini en appliquant la fonction OS2I et en réduisant le résultat modulo p, on a : OS2FE(X) = OS2I$(X) \pmod p$.

D'autre part, dans le cas des protocoles cryptographiques basés sur les courbes elliptiques, nous aurons besoin de stocker ou transmettre un point $P = (x_P, y_P)$ de la courbe elliptique choisie. Dans ce cas, soit on transmet x_P suivie de y_P, ou bien, au lieu de transmettre les deux valeurs x_P et y_P, nous utilisons la technique de compression des points en transmettant seulement x_P avec un bit supplémentaire pour dire quelle valeur de y_P nous devrions prendre à la réception de x_P.

3.4.1 Encodage sans compression :

Dans l'encodage non compressé, le point P est représenté par ses deux coordonnées affines x_P et y_P qui sont deux éléments du corps fini $\mathbb{F}_p$. Si p est de taille t bits, le stockage ou le transfert de (x_P, y_P) nécessite $2t$ bits (en excluant d'autres données supplémentaires nécessaires pour l'encodage).

Encodage :

L'encodage non compressé P_U du point P est défini par $P_U = C\|X\|Y$, où
- $C = 0x04$
- $X = \text{FE2OS}(x_P)$
- $Y = \text{FE2OS}(y_P)$

Décodage :

Étant donné P_U, le point P est récupéré par $P = (\text{OS2FE}(X), \text{OS2FE}(Y))$. Avant d'utiliser le point P, on doit s'assurer que P est toujours un point de la courbe E en vérifiant que $y_P^2 = x_P^3 + ax_P + b$, où a et b sont les coefficients de l'équation réduite de Weierstraß.

3.4.2 Encodage avec compression :

Dans l'encodage compressé, le point P est représenté par x_P et un bit supplémentaire y'_P pour identifier de façon unique y_P. Plus précisément, ce bit est défini comme étant le bit de poids faible (le dernier bit à droite dans la représentation) de y_P, c'est-à-dire $y'_P = 0$ si et seulement si y_P est pair.

Encodage :

L'encodage compressé P_C du point P est défini par $P_C = C\|X$ où
- $C = 0x02$ si $y'_P = 0$
- $C = 0x03$ si $y'_P = 1$
- $X = \text{FE2OS}(x_P)$

Décodage :

Étant donné P_C, le point P est récupéré en tant que $P = (\text{OS2FE}(X), y_P)$. L'algorithme suivant permet de calculer y_P :

1. Poser $\alpha = x_P^3 + ax_P + b$.
2. Vérifier si α est un carré dans $\mathbb{F}_p$. Si α n'est pas un carré, renvoyer une erreur et terminer.
3. Si $\alpha = 0$, alors $y_P = 0$, renvoyer y_P et terminer.
4. Calculer une racine carrée $\beta \in \mathbb{F}_p$ de $\alpha \in \mathbb{F}_p$.
5. Si le bit de poids faible de β est égal à y'_P, alors $y_P = \beta$, sinon $y_P = p - \beta$. Renvoyer y_P.

On peut utiliser le symbole de Legendre pour vérifier si α est un carré dans $\mathbb{F}_p$ ou non. Plus précisément, α est carré dans $\mathbb{F}_p$ si et seulement si $\left(\frac{\alpha}{p}\right) = 1$.

Maintenant, si α est un carré dans $\mathbb{F}_p$ et $p \equiv 3 \pmod 4$ alors les deux racines carrées $\pm\beta$ de α dans $\mathbb{F}_p$ sont efficacement calculées où $\beta \equiv \alpha^{\frac{(p+1)}{4}} \pmod p$.

En effet, $\beta^2 \equiv \alpha^{\frac{(p+1)}{2}} \equiv \alpha^{\frac{(p-1)}{2}}\alpha \pmod p$, comme α est un carré alors $\alpha^{\frac{(p-1)}{2}} \equiv \left(\frac{\alpha}{p}\right) \equiv 1 \pmod p$ d'après le lemme d'Euler, d'où $\beta^2 \equiv \alpha \pmod p$.