الجمهورية الجزائرية الديمقراطية الشعبية

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DR. YAHIA FARES DE MEDEA

FACULTE DE SCIENCES

Département mathématique et informatique



# Mémoire de Fin d'Études

**Présenté pour l'obtention du Diplôme**
**De Master en**

*Option* : Ingénierie des systèmes et technologies du web.

## THEME

# SYSTEME DE CRYPTOGRAPHIE BASEE SUR COURBE ELLIPTIQUE.

**Proposé et dirigé par :**

Mr. CHETOUANE MOHAMED

**Préparé et présenté par :**

Mr. TAREK DJEDDOU

Année Universitaire 2018-2019

**ملخص:**

في هذا العمل، نقدم تصميم وتنفيذ نظام تشفير المفتاح العموميECC. الضرب العددي هو العملية الرئيسية لتشفير / فك تشفير ECC. توفر خوارزمية النافذة الزلقة ذات الحجم $w = 3$ تكلفة أفضل، وبالإضافة فإن استخدام الإحداثيات المختلطة يزيد من سرعة أداء الضرب القياسي. تم تطوير تطبيق برنامج ECC بلغة برمجة جافا. استخدمنا منحنى بيضاوي الشكل NISTالموصي به على جسم $F_{256}$ محدود.

**الكلمات المفتاحية:** نظام تشفير، ECC، الضرب العددي.

**Résumé :**

Dans ce travail, nous présentons la conception et l'implémentation du cryptosystème à clé publique ECC. La multiplication scalaire est l'opération principale du chiffrement/déchiffrement ECC. L'algorithme de la fenêtre glissante avec la taille $w = 3$ donne un meilleur coût, aussi l'utilisation des coordonnées mixtes augmente la performance de la multiplication scalaire. L'implémentation logicielle d'ECC est développée en langage de programmation Java. Nous avons utilisé la courbe elliptique recommandée du NIST sur un corps fini $F_{256}$.

**Mots-clés :** cryptosystème, ECC, multiplication scalaire.

**Abstract:**

In this work, we present the design and implementation of the cryptosystem ECC (Elliptic Curve Cryptography). The scalar multiplication is the main operation of ECC. The algorithm sliding window with size $w = 3$ gives a better cost, so the use of mixed coordinates increases the performance of scalar multiplication. The software implementation of ECC is developed in Java programming language. We used the recommended NIST elliptical curve on a $F_{256}$ finite body.

**Keywords:** cryptosystem, ECC, scalar multiplication.

# *Remerciement*

*Notre remerciement va en premier lieu à ALLAH le tout puissant de nous avoir donné la foi et de nous avoir permis d'en arriver là.*

*Nous tenons à remercier particulièrement mon encadreur Ms*

*CHETOUANE MOHAMED pour son encadrement et pour l'intérêt qu'il a manifesté à notre travail.*

*Nous remercierons très sincèrement, les membres de jury d'avoir bien voulu accepter de faire partie de la commission d'examinateur.*

*Nous adressons également nos remerciements, à tous nos enseignants, qui ont contribué à notre formation tout au long de nos années d'étude.*

*Nous tenons également à remercier tous nos collègues de promotion que nous avons eu le plaisir de les côtoyer pendant cette période de formation.*

*Nous remercierons tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail*

# *Dédicace*

*Je commence par rendre grâce à dieu et à sa bonté, pour la patience, la compétence et le courage qu'il m'a donné pour arriver à ce stade ; Avec un énorme plaisir, un cœur ouvert et une joie immense, que je dédie ce modeste travail tout d'abord : à mes parents pour leurs amours, leurs sacrifices et leur encouragement qui ont fait de moi ce que je suis aujourd'hui ;*

*A mes chers frères, que je leurs souhaite une bonne réussite ainsi qu'à toute ma famille ;*

*Je n'oublierai pas de dédier ce travail à mes précieuses amies YOUSSOUF et RACHID et à toute personne qui me connait ;*

*A mes amis frères et sœurs chers collègues qui j'ai passé de merveilleux instants et de beaux souvenirs.*

*« DJEDDOU TAREK »*

# Table de matière :

*Remerciement* ..... *iv*
*Dédicace* ..... *v*
***Liste des Figures :*** ..... **vi**
***Liste des Tableaux :*** ..... **vi**
***Introduction générale*** ..... ***1***
***Chapitre I :*** ..... ***2***
***1. Introduction :*** ..... ***3***
***2. Terminologie :*** ..... ***3***
***3. Principaux concepts cryptographiques*** ..... ***5***
3.1. Cryptographie Symétrique : ..... 5
3.1.1. Définition ..... 5
3.1.2. Caractéristiques : ..... 6
3.2. Cryptographie Asymétrique : ..... 7
3.2.1. Définition : ..... 7
3.2.2. Caractéristiques : ..... 7
***4. Chiffrement par bloc :*** ..... ***8***
***5. DES (Data Encryption Standard) :*** ..... ***9***
***6. AES (Advanced Encryption Standard) :*** ..... ***11***
***7. Chiffrement par clé publique :*** ..... ***12***
7.1. RSA (Rivest, Shamir et Adleman) : ..... 12
7.1.1. Chiffrement/Déchiffrement par RSA : ..... 13
7.1.2. Problèmes du RSA : ..... 14
***8. Courbe elliptique :*** ..... ***14***
***9. Conclusion :*** ..... ***14***
***Chapitre II :*** ..... ***15***
***1 Introduction*** ..... ***16***
***2 Cryptographie à courbes elliptiques*** ..... ***16***
2.1 Définition d'une courbe elliptique ..... 16
2.2 Loi de Groupe ..... 16
2.3 Approche Géométrique ..... 18
2.3.1 Règles de l'addition ..... 18

2.3.2 Cas particuliers ..... 19
2.4 Choix du corps ..... 20
2.5 Calcul du nombre de points d'une courbe elliptique sur un corps fini ..... 22
2.6 Problème du logarithme discret ..... 22
2.7 Utilisation des courbes elliptiques dans la cryptographie ..... 23
2.7.1 Cryptage des messages ..... 23
2.7.2 Protocole d'échange de clés de Diffie-Hellmann ..... 24
2.7.3 Cryptage des points ..... 26
2.7.4 Chiffrer et déchiffrer de message ..... 26
2.7.4.1 Signification du symbole ..... 27
2.7.4.2 Exemple Chiffrer et déchiffrer de message ..... 27
2.8 Attaques sur la cryptographie à base des courbes elliptiques ..... 30
2.8.1 L'attaque MOV ..... 30
2.8.2 Force Brute ..... 30
2.8.3 Baby Step Giant Step ..... 31
2.9 Utilisation de la cryptographie à base des courbes elliptiques ..... 32
2.9.1 Comparaison de performance ..... 32
2.9.2 Applications qui utilisent la EC ..... 34
*3 Conclusion ..... 35*
*Chapitre III : ..... 36*
*1 Introduction ..... 37*
*2 Définition de la multiplication scalaire ..... 37*
2.1 Les méthodes existantes de la multiplication scalaire ..... 38
2.1.1 Algorithmes binaires simples ..... 38
2.1.2 Méthodes de fenêtre ..... 41
2.1.3 Implémentation et Comparaison de performance de la multiplication scalaire ..... 44
2.1.4 Résultat de temps d'exécution de la multiplication scalaire ..... 46
*3 Les systèmes de coordonnées ..... 47*
3.1 Les formules d'addition et doublement en coordonnées affine ..... 48
3.2 Les formules d'addition et doublement en coordonnées projectives ..... 48
3.3 Les formules d'addition et doublement en Jacobienne et Jacobienne Chudnovski ..... 49
3.4 Les coordonnées Jacobiennes modifies ..... 50

*3 Analyse et discussion des systèmes de coordonnées* ..... *51*
*4 Une nouvelle stratégie pour la courbe elliptique* ..... *51*
Coordonnées mixtes ..... 52
*5 Comparaison de performance de système de coordonner de la courbe elliptique* ..... *54*
6 **Résultat d'exécution de la multiplication scalaire dans les systèmes de coordonner** .... 55
*7 Conclusion* ..... *57*
***Chapitre IV :*** ..... *58*
***1Introduction*** ..... *59*
***2 Étude de l'algorithme de fenêtre glissante (Sliding window)*** ..... *59*
***3 Conception*** ..... *65*
***4 Implémentation*** ..... *67*
4.1 Outils et Environnement ..... 67
4.2 Intérface principale ..... 67
4.3 Etapes de génération de la clé publique ..... 71
***5 Résultat et discussion*** ..... *73*
5.1 Temps de chiffrement ..... 73
5.2 Temps de déchiffrement ..... 74
***6 Conclusion*** ..... *75*
***Conclusion Générale*** ..... *76*

# Liste des Figures :

**Figure I. 1: Une représentation d'un cryptosystème. ..... 4**
**Figure I. 2: les méthodes de la cryptographie moderne. ..... 5**
**Figure I. 3: Chiffrement symétrique ..... 6**
**Figure I. 4: Chiffrement asymétrique. ..... 8**
**Figure I. 5: Algorithme principal du DES. ..... 9**
**Figure I. 6 : Détails de la fonction principale d'un étage. ..... 10**
**Figure I. 7: Algorithme TDES. ..... 11**
**Figure I. 8: Algorithme Chiffrement et Déchiffrement AES. ..... 12**

**Figure II. 1: Courbe elliptique d'équation $y2= x3 - 5x2 + 3$ [12]. ..... 17**
**Figure II. 2: Addition de deux points sur une courbe elliptique sur R. ..... 18**
**Figure II. 3:Doublement d'un point sur une courbe elliptique sur R. ..... 19**
**Figure II. 4:– EC sur *Z/pZ* [7] ..... 21**
**Figure II. 5: Exemple de table de correspondance pour *E* : $y2= x3+ 2x + 9$ (mod 37)[18].. 23**
**Figure II. 6: Échange de clés de Diffie Hellman (ECDH) [20]. ..... 25**
**Figure II. 7:Longueur de clé en bit [23]. ..... 32**
**Figure II. 8:Temps d'exécution pour la génération de clés ..... 33**
**Figure II. 9: Temps d'exécution pour la signature numérique [23]. ..... 33**
**Figure II. 10:Temps d'exécution pour la vérification de signature [23]. ..... 34**

**Figure III. 1 : Le modèle à trois niveaux pour la multiplication scalaire. ..... 37**
**Figure III. 2 : algorithme 3.1 binaire de droite à gauche pour la multiplication de points. 39**
**Figure III. 3: algorithme 3.2 binaire de gauche à droite pour la multiplication de points. . 39**
**Figure III. 4: Algorithme 3.3 Calcul de la forme non-adjacente d'un entier. ..... 40**
**Figure III. 5: Algorithme 3.4 Calcul de NAF. ..... 41**
**Figure III. 6: Algorithme 3.5 Calcul de la forme non-adjacente de largeur *w* d'un entier. . 42**
**Figure III. 7: Algorithme 3.6 Doublement et addition avec fenêtres. ..... 42**
**Figure III. 8: Algorithme 3.7 Méthode de la fenêtre glissante ('sliding window') [4]. ..... 43**
**Figure III. 9: la procédure pour calculer le temps d'exécution de chaque méthode. ..... 45**
**Figure III. 10 : présentions graphique de temps (ms) pour les différentes méthodes de la multiplication dans le champ $F256$. ..... 47**
**Figure III. 11: la procédure pour calculer le temps d'exécution de chaque système de cordonner. ..... 55**
**Figure III. 12 : présentions graphique de temps (ms) pour les différentes systèmes des coordonnées. ..... 56**

**Figure IV. 1 représentation de nombre daddition et doublement par rapport la taille de fenêtre..... 62**
**Figure IV. 2: le nombre de recalculés par rapport la taille de fenêtre. ..... 63**
**Figure IV. 3 : la procédure pour calculer le temps d'exécution de chaque fenêtre. ..... 64**
**Figure IV. 4: Diagramme cas utilisation de notre système ……..….76**
**Figure IV. 5 : Diagramme d'activité du fonctionnement global de l'application. ..... 66**
**Figure IV. 6 : Interface principale de l'application. ..... 68**
**Figure IV. 7 : Générer une clé publique ..... 69**
**Figure IV. 8 : Cryptage du message ..... 70**
**Figure IV. 9 : Déryptage du message ..... 71**
**Figure IV.10 :Temps d'encryptions entre l'ECC standard et l'ECC proposé. ..... 74**
**Figure IV.11 :Temps de décryptions entre le ECC standard et leECC proposé. ..... 75**

# Liste des Tableaux :

**Tableau III. 1: les Résultats de temps (ms) pour les différentes Méthodes de la multiplication dans le champ $F256$. ..... 47**
**Tableau III. 2: Coût des opérations de systèmes de coordonner..... 51**
**Tableau III. 3:Coût des doublements et additions mixtes sur Fp [43]. ..... 53**
**Tableau III. 4:Algorithme3.8 Conversion des coordonnées Affines (x , y) en d'autres ..... 54**
**Tableau III. 5: Temps d'exécution des systèmes de cordonner de la courbe elliptique ..... 56**

**Tableau IV.1: le coût des opérations du doublement, additions et de précalculs par rapport la taille de fenêtre..... 62**
**Tableau IV.2 : Résultats de cadencement (ms) des méthodes de fenêtre sur $F_{256}$ pour différentes largeurs de fenêtre...……..75**

# Introduction générale

Avec l'apparition de l'internet et les réseaux sans fils où l'échange de l'information est partout d'où la nécessité d'assurer la sécurité des informations devient un très grand challenge.

La cryptographie peut assurer les principaux services de la sécurité telle que la confidentialité et l'intégrité des informations. Selon la notion de la clé, il existe deux types de cryptographies : la cryptographie asymétrique et la cryptographie symétrique, ce dernier à un inconvénient majeur c'est l'absence d'un canal sécurisé pour le transfert de la clé secrète. La cryptographie à clé publique évite ce problème.

Le cryptosystème à clé publique le plus utilisé nos jour est le RSA et avec l'évolution énorme des machines de calcul. A chaque fois le cryptosystème RSA est cassé en augmentant la taille de la clé, actuellement on utilise des tailles de 2048 bits pour assurer une sécurité robuste.

Cette taille de clé n'est adaptée pour les systèmes qui y'a des ressources limitées tel que la taille de la mémoire, la puissance de calcul…Tout cela pousse la recherche sur une nouvelle technique de chiffrement en utilisant les courbes elliptiques.

Dans ce mémoire, nous allons étudier l'utilisation des courbes elliptiques dans la cryptographie. Nous allons aussi décrire l'application ECC qu'on a développé pour chiffrer et déchiffrer un message clair écrit en caractères latins et arabes, prenons en considération le temps d'exécution du chiffrementet déchiffrement par le cryptosystème ECC. Afin de bien organiser le travail, nous avons décomposé notre mémoire en quatre chapitres :

Chapitre I : ce chapitre discute les principaux concepts de base de cryptographie.

Chapitre II : le deuxième chapitre présente les principaux concepts des courbes elliptiques.

Chapitre III :ce chapitre porte sur les mécanismes et les méthodes utilisés pour le calcul de la multiplication scalaire qui est le noyau d'un cryptosystème ECC.

Chapitre IV: ce dernier chapitre a amené la conception et l'implémentation du cryptosystème ECC ainsi une comparaison des résultats obtenus avec quelques résultats de la littérature**.**

On termine ce mémoire avec une conclusion générale sur le processus de développement de l'application, ainsi que des perspectives pour une amélioration éventuelle de ce travail.

**Chapitre *I :***

# GÉNÉRALITÉS SUR LA CRYPTOGRAPHIE.

# 1. Introduction :

Depuis l'émergence de l'univers et les hommes ont besoin de répondre à deux choses fondamentales, la communication et le partage des informations avec les autres et la communication secrète ont conduit à l'émergence de l'art de coder les messages de manière à ce que seules les personnes auxquelles elles sont destinées puissent comprendre ces messages sans que des personnes non autorisées à le faire.

Dans ce chapitre on va présenter une introduction générale à la cryptographie, en passant par les notions et terminologies utilisées dans cette science, puis expliquer le fonctionnement des modes et protocoles de chiffrement symétrique et asymétrique.

On va aussi donner une description de quelques fonctions et algorithmes qui sont utilisés avec le chiffrement symétrique et asymétrique pour garantir l'intégrité et l'authenticité des données.

# 2. Terminologie :

- **Cryptologie :** Il s'agit d'une science mathématique comportant deux branches : la cryptographie et la cryptanalyse [1].
- **la Cryptographie :** c'est un ensemble de méthodes permettant de transmettre des données de manière confidentielle. Afin de protéger un message, on le transforme pour le rendre incompréhensible, c'est ce qu'on appelle *le chiffrement*. Inversement, **le déchiffrement** est l'action qui permet de reconstruire le message en claire à partir du message chiffré [1].
- **La cryptanalyse :** à l'inverse, est l'étude des procédés cryptographiques dans le but de trouver des faiblesses, en particulier, de pouvoir décrypter des messages chiffrés. Le décryptement est l'action consistant à trouver le message en clair sans connaitre la clé de déchiffrement.
- **Chiffrement :** Transformation à l'aide d'une clé de chiffrement d'un message en clair en un message incompréhensible si on ne dispose pas d'une clé de déchiffrement.
- **Déchiffrement :** le déchiffrement est une action qui permet de reconstruire le texte en clair à partir du texte chiffré en utilisant une clé.
- **Texte chiffré :** Appelé également cryptogramme, le texte chiffré est le résultat de l'application d'un chiffrement à un texte clair.

- **Clef :** Une clé est un paramètre utilisé en entrée d'une opération cryptographique (chiffrement, déchiffrement, ...).
- **Coder :** Transformer un texte, une information en remplaçant les mots dans une écriture faite de signes prédéfinis, suivant des normes internationales.
- **Cryptosystème :** c'est un triplé de l'espace des clés, l'ensemble des messages chiffrés et l'ensemble des messages clairs. Ces trois éléments sont reliés par les deux opérations de chiffrement et de déchiffrement. Formellement on :

$$\begin{cases} \mathbf{C} = \mathbf{E}(\mathbf{M}) \\ \mathbf{M} = \mathbf{D}(\mathbf{E}(\mathbf{M})) \end{cases}$$

  - M : est le texte clair.
  - C : est le texte chiffré.
  - K : est la clé (dans le cas d'un algorithme à clé symétrique).
  - $E_k$ et $D_k$ dans le cas d'algorithmes asymétriques.
  - E(x) : est la fonction de chiffrement.
  - D(x) :est la fonction de déchiffrement.

D'où on peut représenter un cryptosystème par la Figure [I.1] :

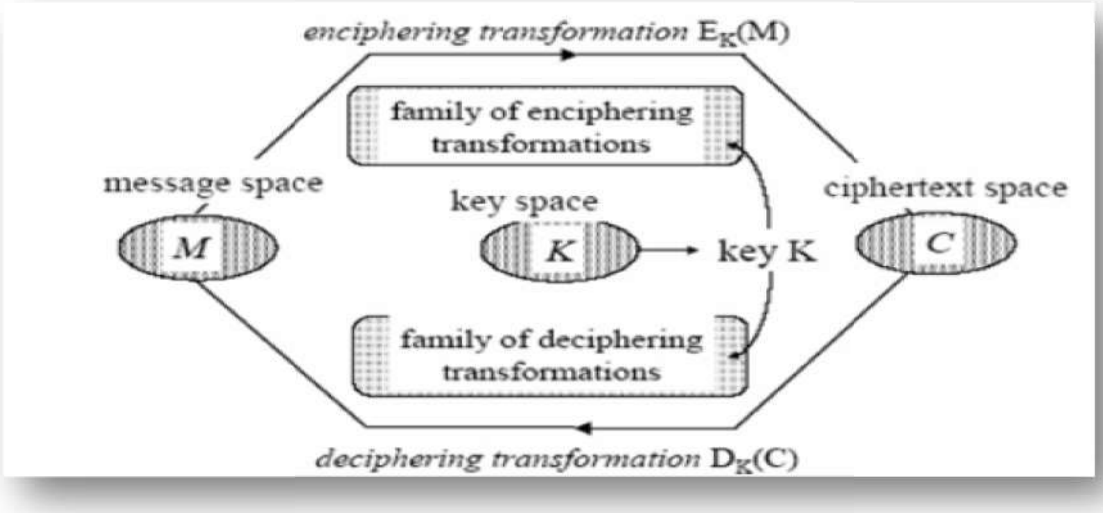



Figure I. 1: Une représentation d'un cryptosystème [2].

## 3. Principaux concepts cryptographiques

La cryptographie classique concerne la période de l'antiquité jusqu'à l'apparition des ordinateurs. Elle inclut tous les mécanismes et algorithmes basés sur des fonctions mathématiques ou logiques (exemple : Cesar, Vigenère) [2].

Selon la notion de clé on peut distinguer deux types de cryptographie : la cryptographie symétrique et la cryptographie asymétrique comme le montre la Figure [I.2] :

- La cryptographie à clés secrètes ou cryptographie symétrique.
- La cryptographie à clés

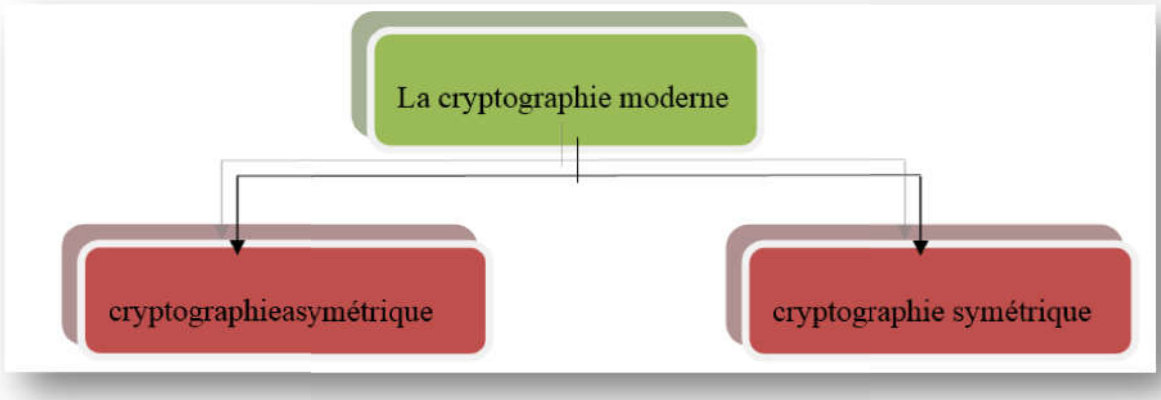



Figure I. 2:les méthodes de la cryptographie moderne.

### 3.1.Cryptographie Symétrique :

#### 3.1.1.Définition

Le cryptage symétrique est la technique la plus ancienne et la plus connue. Une clé secrète, est appliquée au texte d'un message pour modifier le contenu d'une certaine manière. Cela pourrait être aussi simple que de décaler chaque lettre d'un certain nombre d'emplacements dans l'alphabet. Tant que l'expéditeur et le destinataire connaissent la clé secrète, ils peuvent crypter et décrypter tous les messages qui utilisent cette clé.

Le problème de ce chiffrement est qu'il faut trouver un moyen de transmettre la clé unique entre les deux interlocuteurs [3].

### 3.1.2. Caractéristiques :

- Les clés sont identiques : K.C = K.D = K
- La clé doit rester secrète**.**
- Les algorithmes les plus répandus sont le DES, 3DES, AES,...
- Ces algorithmes sont basés sur des opérations de transposition et de substitution des bits du texte clair en fonction de la clé.
- La taille des clés est souvent de l'ordre de 128 bits. Le DES en utilise 56, mais l'AES peut aller jusqu'à 256.
- L'avantage principal de ce mode de chiffrement est sa rapidité.
- Le principal désavantage réside dans la distribution des clés : pour une meilleure sécurité, on préférera l'échange manuel. Malheureusement, pour de grands systèmes, le nombre de clés peut devenir conséquent. C'est pourquoi on utilisera souvent des échanges sécurisés pour transmettre les clés. En effet, pour un système à N utilisateurs, il y aura N.(N − 1)/2 paires de clés [4].

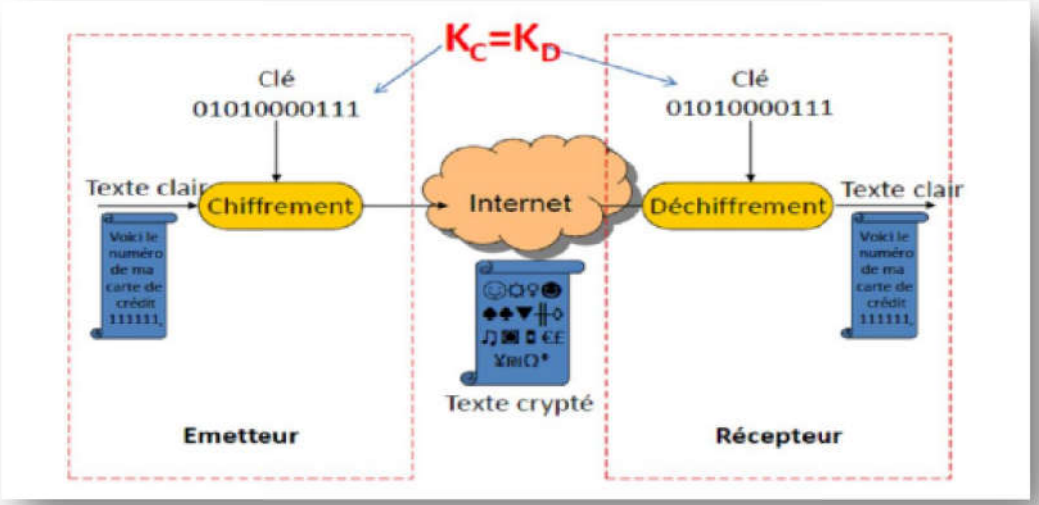



Figure I. 3:Chiffrement symétrique[4].

## 3.2.Cryptographie Asymétrique :

### 3.2.1. Définition :

La cryptographie asymétrique ne demande aucun échange préalable de secret. Chaque utilisateur possède deux clés mathématiquement liées : l'une secrète et l'autre publique. Ce qui est chiffré par l'une des deux clés n'est déchiffrable que par l'autre, c'est un procédé asymétrique. La clé publique sert à chiffrer des messages cependant la clé privée sert à déchiffrer les messages chiffrés avec la clé publique correspondante comme le montre la figure 1.4. La clé publique est connue par tout le monde. Par contre la clé privée doit rester confidentielle, c'est la seule qui permet de déchiffrer les messages. En plus il est impossible de déduire la clé privée de la clé publique [5].

### 3.2.2.Caractéristiques :

- Une clé publique PK (symbolisée par la clé verticale).
- Une clé privée secrète SK (symbolisée par la clé horizontale).
- Propriété : La connaissance de PK ne permet pas de déduire SK.
- DSK (EPK(M)) = M (DSK : Décryptions par la clé secrètes et EPK : Encryptions par la clé publiques).
- Les algorithmes se basent sur des concepts mathématiques tels que l'exponentiation de grands nombres premiers (RSA), le problème des logarithmes discrets (ElGamal), ou encore le problème du sac à dos (Merkle-Hellman).
- Le cryptosystème asymétrique le plus utilisé nos jours est le RSA.
- Le principe de ce genre d'algorithme est qu'il s'agit d'une fonction unidirectionnelle à trappe. Une telle fonction à la particularité d'être facile à calculer dans un sens, mais difficile voire impossible dans le sens inverse. La seule manière de pouvoir réaliser le calcul inverse est de connaître une trappe. Une trappe pourrait par exemple être une faille dans le générateur de clés. Cette faille peut êtresoit intentionnelle de la part du concepteur (définition stricte d'une trappe) ou accidentelle.

- La taille des clés s'étend de 512 bits à 2048 bits en standard. Dans le cas du RSA, une clé de 512 bits n'est plus sûre au sens "militaire" du terme, mais est toujours utilisable de particulier à particulier.
- Au niveau des performances, le chiffrement par voie asymétrique est environ 1000 fois plus lent que le chiffrement symétrique.
- Cependant, à l'inverse du chiffrement symétrique où le nombre de clés est le problème majeur, ici, seules n paires sont nécessaires. En effet, chaque utilisateur possède une paire (SK, PK) et tous les transferts de message ont lieu avec ces clés.
- La distribution des clés est grandement facilitée car l'échange de clés secrètes n'est plus nécessaire. Chaque utilisateur conserve sa clé secrète sans jamais la divulguer. Seule la clé publique devra être distribuée [4].

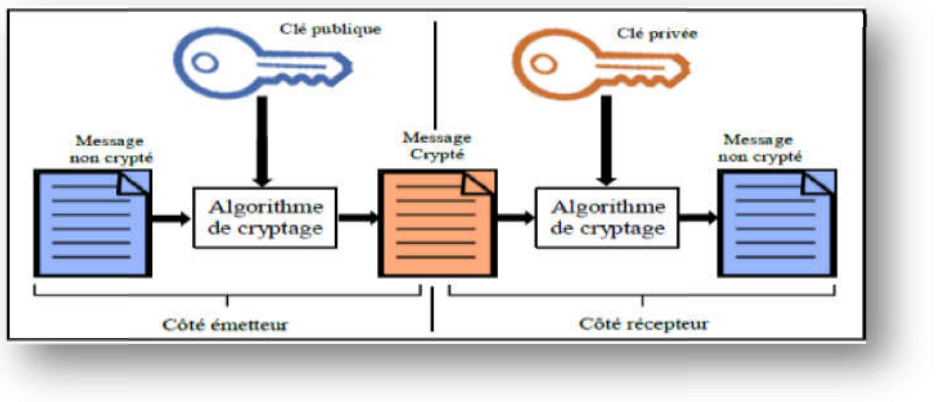



Figure I. 4:Chiffrement asymétrique [4].

## 4. Chiffrement par bloc :

Dans un algorithme de chiffrement par bloc, chaque message clair est découpé en blocs de taille fixe de même longueur et chiffré à l'aide d'une clé unique. Ces algorithmes sont en général construits sur un modèle itératif. Il utilise une fonction F qui prend une clé secrète k et un message M de n bits. La fonction F est itérée un certain nombre de fois (nombre de tours). Lors de chaque tour, la clé k est différente et on chiffre le message qui vient d'être obtenu de l'itération précédente. Les différentes clés k(i) qui sont utilisées sont déduites de la clé secrète k [**2**].

## 5. DES (Data Encryption Standard) :

Le DES est un algorithme de chiffrement par blocs produit par le NIST. C'est un algorithme de chiffrement par blocs et consiste à découper les données en blocs de taille fixe. Le point faible de cet algorithme reste sa clé qui est de petite taille (64 bits), et non variable, ce qui n'offre pas la possibilité de l'adapter aux différents niveaux de sécurité [6].

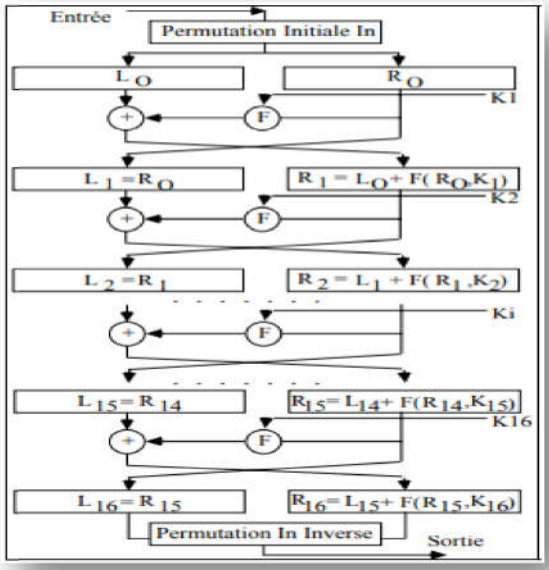



Figure I. 5: Algorithme principal du DES [6].

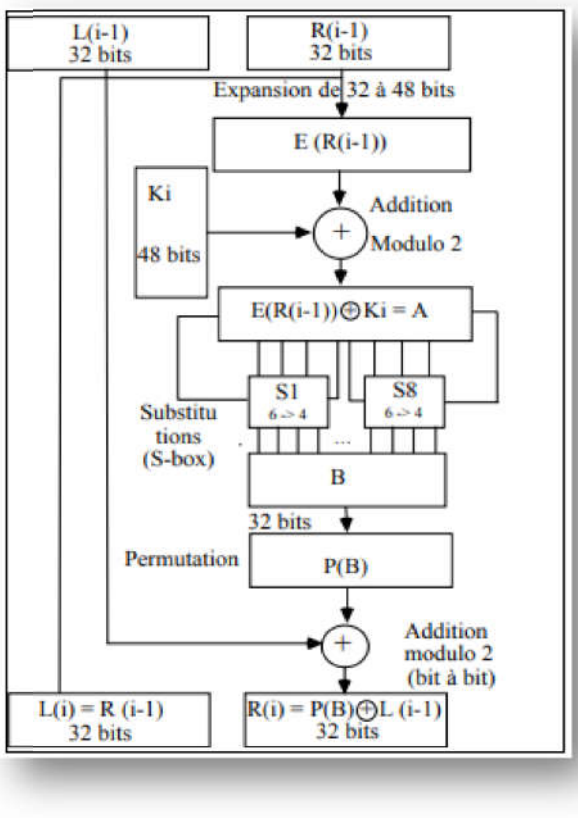



Figure I. 6 :Détails de la fonction principale d'un étage [6].

L'algorithme de chiffrement repose principalement sur 3 étapes, en plus de la gestion spécifique de la clé :

a) Permutation initiale.
b) Calcul médian (16 fois) : application d'un algorithme complexe appliqué en fonction de la clé.
c) Permutation finale.

Pour le déchiffrement, Il suffit d'appliquer le même algorithme mais inversé en tenant bien compte du fait que chaque itération du déchiffrement traite les mêmes paires de blocs utilisés dans le chiffrement.

Avec la technologie actuelle (des CPU très rapides et moins chers), il est hautement possible de cracker DES. La solution a été au premier temps d'adopter le triple DES (TDES ou 3DES) : 3 applications de DES à la suite avec 2 clés différentes (112 bits) Figure [1.7].

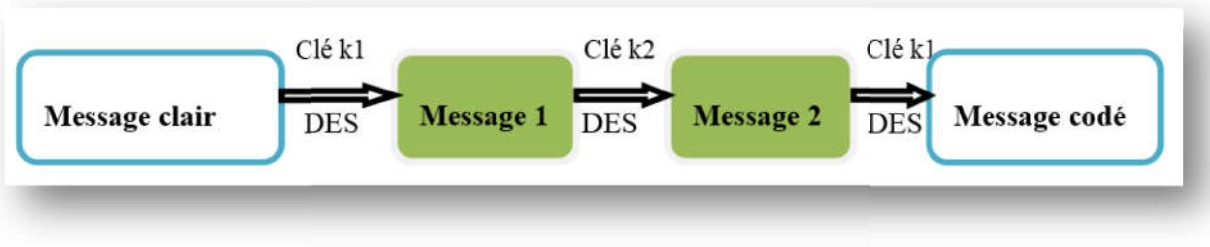



Figure I. 7:Algorithme TDES [7].

**:Algorithme TDES.**TDES utilise une taille de blocs est 64 bits et de clé comprise entre 128 bits et 192 bits. TDES est suffisant en sécurité mais il est trois fois plus lent que DES. Un nouvel algorithme remplace DES, c'est l'AES (Advanced Encryption Standard). [7]

## 6. AES (Advanced Encryption Standard) :

Suite aux failles du DES, quelques modifications ont été apportées (2DES, 3DES..), mais pas toujours avec succès. L'algorithme de AES (Advanced Encryption Standard) est apparue pour remplacer le DES et ces variantes. Le AES possède les propriétés suivantes [4] :

- Plusieurs longueurs de clef et de bloc sont possibles : 128, 192, ou 256 bits.
- Le nombre de cycles ("rondes") varie en fonction de la longueur des blocs et des clés (de 10 à 14).
- La structure générale ne comprend qu'une série transformations/permutations/- sélections.
- Il est beaucoup plus performant que le DES.
- Il est facilement adaptable à des processeurs de 8 ou de 64 bits.
- Le parallélisme peut être implémenté.

**Chiffrement et Déchiffrement :**

L'ordonnancement des étapes est illustré à la Figure [1.8].

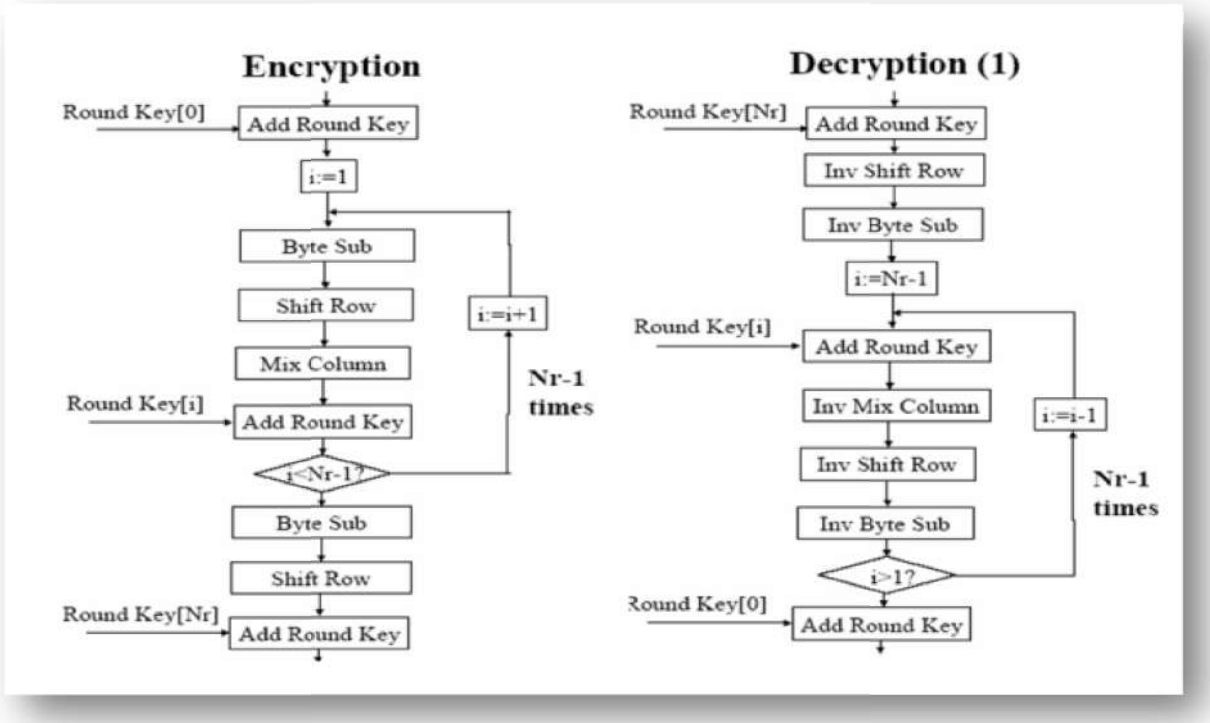



**Figure I. 8:**Algorithme Chiffrement et Déchiffrement AES [7] .

## 7. Chiffrement par clé publique :

Le protocole le plus connu dans le chiffrement par la clé publique est le protocole RSA.

### 7.1.RSA (Rivest, Shamir et Adleman) :

Le cryptage RSA, du nom de ses concepteurs, Ron Rivest, Adi Shamir et Leonard Adleman, est le premier algorithme de chiffrement asymétrique. Il a été découvert en1977. Le RSA est basé sur la théorie des nombres premiers, et sa puissance vient du fait qu'il n'existe aucun algorithme qui permet la décomposition d'un nombre en facteurs premiers [8].

RSA est basé sur le calcul exponentiel. Sa sécurité repose sur la fonction unidirectionnelle suivante : le calcul du produit de 2 nombres premiers

- Une clé publique **PK** pour RSA est composée de deux nombres entiers **PK**=(**N , e**) tels que:
  - $N=pq$ pour deux grands nombres premiers p et q de la même taille.
  - $e>1$ et $PGCD(\varphi(N), e) = 1$ (c-a-d: $e \in Z^*\varphi(N)$).

- La clé secrète associée à la clé publique (N,e) est d tel que:
  - **$d \in Z^*\varphi(N)$ et**
  - **$ed = 1 \bmod \varphi(N)$.**

### 7.1.1.Chiffrement/Déchiffrement par RSA :

Chiffrement à partir de la clé publique PK=(N, e):

Le message $M \in Z^*_N$ est chiffré comme: $C = M^e \bmod N$

Algorithmique de RSA Calcul des puissances modulo n

Soit la décomposition de e sur k bits
$e = \Sigma\, e(i)\, 2^i$
début
C := 1
pour i := k -1 à 0 faire
C := $C^2$ (mod n) ;
si ( e(i) = 1 ) alors
C := C . M (mod n) ;
finsi;
fin pour;
fin;

Même algorithme pour déchiffrer.

Déchiffrement à partir de la clé secrète SK=d:

Le cryptogramme $C \in Z^*_N$ est déchiffré comme:

$C^d \bmod N = ( M^e \bmod N )^d \bmod N$

$= M^{ed} \bmod N = M^{kQ(N)+1} 1 \bmod N$
$= (M^{Q(N)})^K \text{ k mod N} \times \text{M mod N}$
$= 1^k \times \text{M mod}$
= M.

### 7.1.2. Problèmes du RSA :

Les experts recommandent d'utiliser des nombres N de 1024 bits (309 chiffres décimaux) pour l'usage commercial est de 2048 bits (617 chiffres décimaux) pour une garantie se prolongeant sur une longue période de temps. Cette longueur tend à grandir dans le futur en même temps que la cryptanalyse et le pouvoir de calcul des machines progressent.

L'augmentation de la taille du modulo N induit beaucoup de calculs qui sont l'élévation d'un gros nombre à une grosse puissance modulo N. C'est-à-dire calculer $\mathbf{M^e}$ mod N connue sous : l' exponentiation modulaire qui est une suite de multiplications modulaires et utilisée simultanément dans le chiffrement et le déchiffrement.

Ces limites ont incité à chercher un nouveau cryptosystème basé sur les courbes elliptiques (ECC) dont les clés sont de petites tailles comparées aux clés de RSA avec une sécurité égale.

## 8. Courbe elliptique :

Il s'agit d'un concept proposé en 1985 par deux chercheurs Miller et Klobitz, de façon totalement indépendante. Ce type de cryptographie, toujours basé sur le modèle asymétrique, permet aussi bien de chiffrer que de signer. On utilise souvent l'abréviation ECC, pour Elliptic Curve Cryptography. Les clés utilisées sont plus courtes pour une sécurité égale ou supérieure [9].

La théorie sous-jacente, ainsi que l'implémentation sont plus complexes, Ils seront détaillés dans les chapitres suivants.

## 9. Conclusion :

On a présenté dans ce chapitre une vue d'ensemble sur la cryptographie moderne en mettant le point sur les terminologies et les notations essentielles dans le domaine de cryptologie. On a aussi expliqué les différents modes de cryptologie et les fonctions de hachages, ce qui nous donne une base pour passer au deuxième chapitre où on va s'intéresser au courbes elliptiques qui sont utilisées pour la cryptographie asymétrique.

Chapitre *II :*

# LA CRYPTOGRAPHIE SUR LES COURBES ELLIPTIQUES

# 1 Introduction

Les courbes elliptiques sont très utilisées dans la cryptographie dans plusieurs applications et protocoles de nos jours. Dans ce chapitre, nous allons mettre le point sur la théorie des courbes elliptiques et leurs applications dans la cryptographie.

Nous allons en premier temps donner les notions mathématiques des EC (Elliptic Curves), puis détailler la procédure complète pour chiffrer un texte clair par EC. Nous allons aussi discuter comment on peut utiliser les courbes elliptiques dans la signature numérique.

Nous allons donner une description des attaques connue sur la cryptographie à base des courbes elliptiques pour éviter d'avoir une mauvaise implémentation de cette dernière.

Nous finirons par donner une comparaison entre la cryptographie à base des courbes elliptiques et le RSA.

# 2 Cryptographie à courbes elliptiques

## 2.1 Définition d'une courbe elliptique

Soit K un corps fini, on appelle courbe elliptique sur K une courbe dans le plan projectif, cubique et sans points singuliers, et munie d'un point distingué qui jouera un rôle particulier : élément neutre. Elle est donc dénie par un polynôme irréductible homogène en trois variables à coefficient dans K [10].Par un changement de variables homographique, on peut toujours se ramener à une équation dite de Weierstrass :

$$y^2z + a_1xyz + a_3yz^2 = x^3 + a_2x^2z + a_4xz^2 + a_5z^3;\ Avec a_1;\ a_2;\ a_3;\ a_4;\ a_5 \in K$$

La courbe elliptique E est l'ensemble des points (x, y ,z ) $\in K^2$satisfaisant cette équation et d'un pointimaginaire appelé point à l'infini [11].

## 2.2 Loi de Groupe

Les applications des courbes elliptiques en cryptographie sont principalement dues à l'existence d'une loi de groupe que nous pouvons définir sur ces dernières. En effet, l'ensemble $E \cup \partial$ peutêtre doté d'une opération d'addition qui produit un groupe abélien dont l'élément neutreest le point à l'infini $\vartheta$ [12]. En cryptologie, les courbes elliptiques sont utilisées dans le corps Fp avec p un nombre premier strictement supérieur à 3. Soit E une courbe elliptique dénie sur Fp.

L'équation affine de Weierstrass de E peut être simplifiée à cette équation :

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

Un exemple typique de courbe elliptique est donné sur la figure II.1. Son équation est $y^2 = x^3 - 5x^2 + 3$.

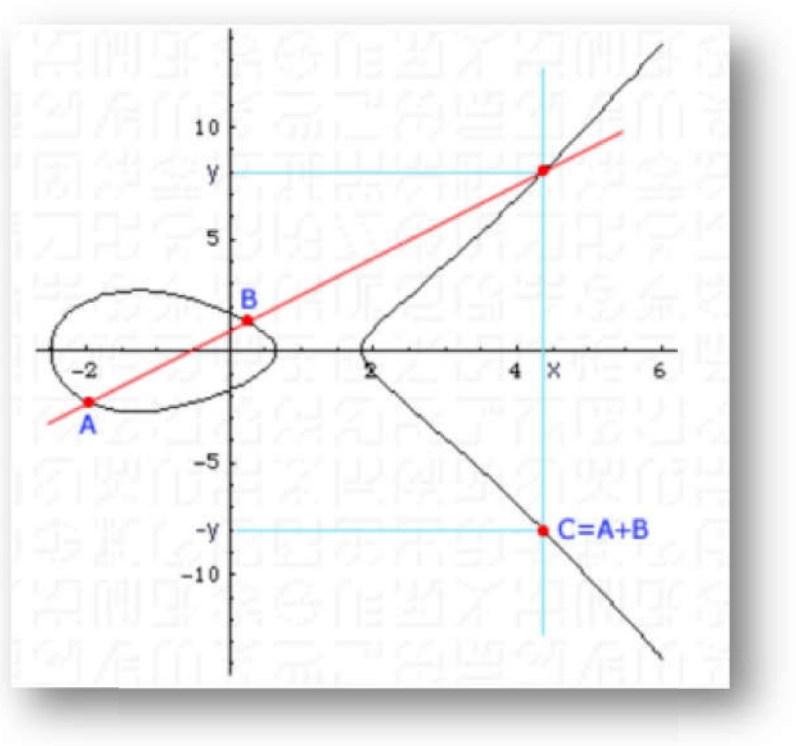



Figure II. 1: Courbe elliptique d'équation $y^2 = x^3 - 5x^2 + 3$ [12].

Soit $F_p$ un corps ni premier de sorte que p soit un nombre premier impair, et soit a, b ∈ $F_p$, tel que :

$$4a^3 + 27b^2 \neq 0 \pmod{p}$$

Puis une courbe elliptique E($F_p$) de $F_p$ dénie par les paramètres a, b ∈$F_p$ comprend l'ensemble de solutions ou de points P = (x, y) pour x, y ∈$F_p$ à l'équation :

$$y^2 = x^3 - 5x^2 + 3.$$

## 2.3 Approche Géométrique

Soient E une courbe elliptique définie sur un corps K, et deux points *P, Q* ∈ *E*(*K*), *L* la droite reliant *P* à *Q* (la tangente à *E* si *P* = *Q*) et *R* le troisième point d'intersection de L avec E. Soit *L0* la droite verticale passant par R. On définit *P* + *Q* ∈ *E*(*K*) comme étant le deuxième point d'intersection de L'avec E. Muni de cette loi de composition (*E*(*K*), +) est un groupe abélien dont l'élément neutre est le point à l'infini (*φ*) [13].

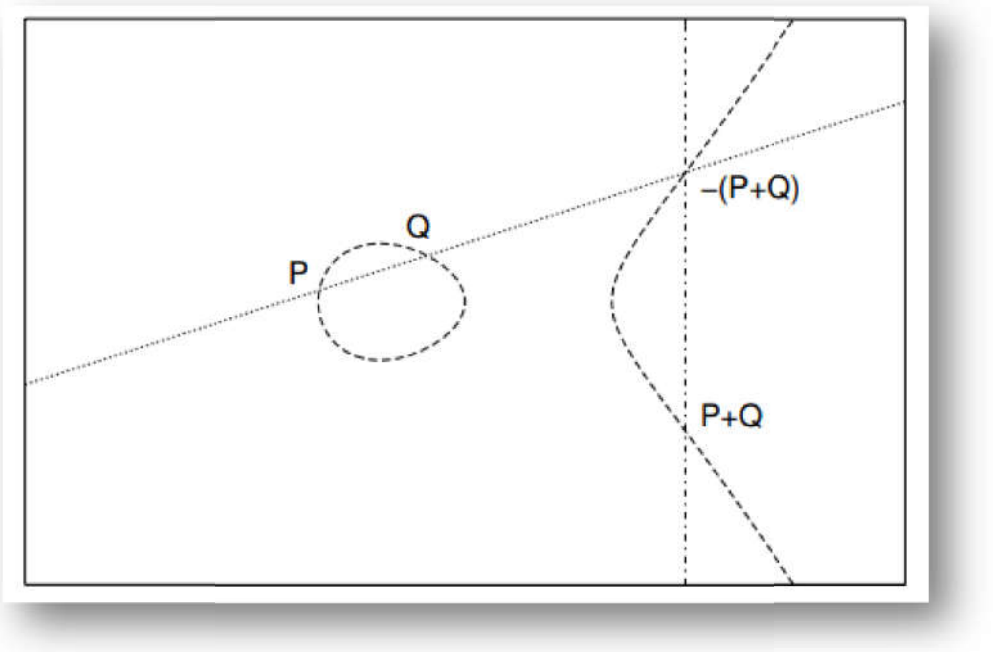



Figure II. 2: Addition de deux points sur une courbe elliptique sur R.

### 2.3.1 Règles de l'addition

Soient P = ($x_1$ ; $y_1$) et Q = ($x_2$ ; $y_2$) deux points sur E :

- Le point ($x_1$ ; -$y_1$) est l'opposé du point P et il est noté -P.
- Si Q ≠ P et Q ≠ - P, alors le point R = P+Q = ($x_3$ ; $y_3$) est déni par :

$$x_3 = \frac{y_1 - y_2}{x_1 - x_2} - x_1 - x_2$$

$$y_3 = \frac{y_1 - y_2}{x_1 - x_2}(x_3 - x_1)$$

Exemple 2 (exemple d'addition de courbe elliptique) :

- 1. soient P = (3,10) et Q = (9,7). P + Q = ($x_3$, $y_3$) est calculé comme suit :

$$\lambda = \frac{7-10}{9-3} = \frac{-3}{6} = \frac{-1}{2} \in K_{23}$$

$x_3 = 11^2 - 3 - 9 = 6 - 3 - 9 = 6 \equiv 17 \pmod{23}$, et

$y_3 = 11(3 + 6) - 10 = 11(9) - 10 = 89 \equiv 20 \pmod{23}$.

d'où P + Q = (17, 20).

### 2.3.2 Cas particuliers

- Si $P_1$ et $P_2$ sont symétriques par rapport à l'axe des abscisses, on considère alors que la somme des deux points est nulle. Il s'agit donc du point à l'infini.
- Si la droite (L) est tangente en un point de la courbe, on considère qu'elle intersecte deux fois la courbe en ce point et par conséquent la somme de $P_1$ et $P_2$ (ou bien le doublement de P) est le symétrique du point de tangence.

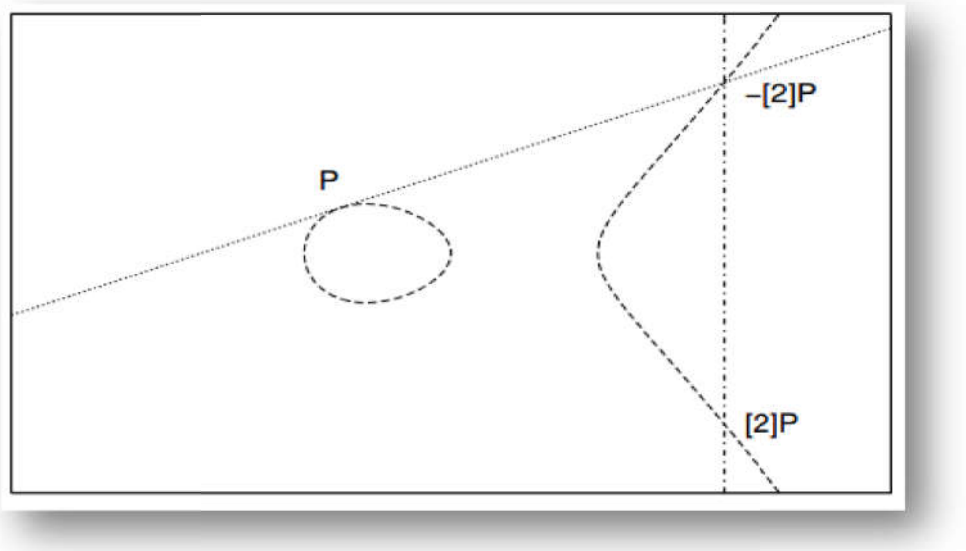



Figure II. 3:Doublement d'un point sur une courbe elliptique sur R.

❖ **Si P = Q, alors le point 2P = ($x_3$; $y_3$) est défini par :**

$$x_3 = \frac{3X_1^2 + a}{2y_1} - x_1$$

$$y_3 = x_1 + \frac{3X_1^2 + a}{2y_1}(x_3 - x_1)$$

- **Si $x_1 = x_2$ mais $y_1 \neq y_2$, alors R = .**
- **Si P = Q et $y_1 = 0$, alors R = $\partial$ .**

Exemple 2 (exemple Doublement de courbe elliptique)

$$\lambda = \frac{3(3^2+1)}{20} = \frac{5}{20} = \frac{1}{4}$$

$x_3 = 6^2 - 6 = 30 \equiv 7 \pmod{23}$, et

$y_3 = 6(3 - 7) - 10 = -24 - 10 = -11 \equiv 12 \pmod{23}$.

D'où 2P = (7, 12).

**Remarque**: Notons $\partial$le point distingué de E. Il existe une loi de groupe sur les points de E telle que $\partial$soit l'élément neutre. L'opposé de chaque point de E est son symétrique par rapport à l'axe des abscisses [14]. La somme de deux points P, Q finis et non opposés est donc le point de coordonnées ($x_3$, $-y_3$) où $x_3$, $y_3$ sont donnés par la relation (2.2). Lorsque l'un des points est à l'infini, ou lorsqu'ils sont symétriques l'un de l'autre, on a les identités P + $\vartheta$ = P et P + (-P) = $\vartheta$.

## 2.4 Choix du corps

En pratique, on a besoin d'un groupe avec un nombre fini d'éléments, pour cela on se place dans un corps *Z/pZ*. Il s'agit d'un corps de dimension finie. On va donc réaliser tous les calculs modulo p, c'est-à-dire que l'on considère uniquement les points d'abscisse entière et inférieure ou égale à p [15].

En pratique, on utilise E définie sur :

- F*p*, l'ensemble des entiers modulo *p* , avec *p* un grand nombre premier.
- $\boldsymbol{Fn^2}$, qui peut être vu comme l'ensemble des mots de longueur *n* , avec *n* premier.

**Ensemble de points :**

Déterminons l'ensemble des points d'une courbe donnée par une équation mise sous la forme réduite de Weierstrass i.e. $y^2 = x^3 + ax + b$ en se plaçant dans le corps $Z/pZ$. Pour cela, on considère $x \in [1, p]$. Pour chaque valeur de x, on calcule $x^3+ax+b \bmod p$. On en déduit s'il existe un ou plusieurs entier y vérifiant $y^2 = x^3 + ax + b$. Si c'est le cas, ces valeurs de $y \in [1, p]$ nous donnent l'ordonnée des points d'abscisse x appartenant à la courbe.

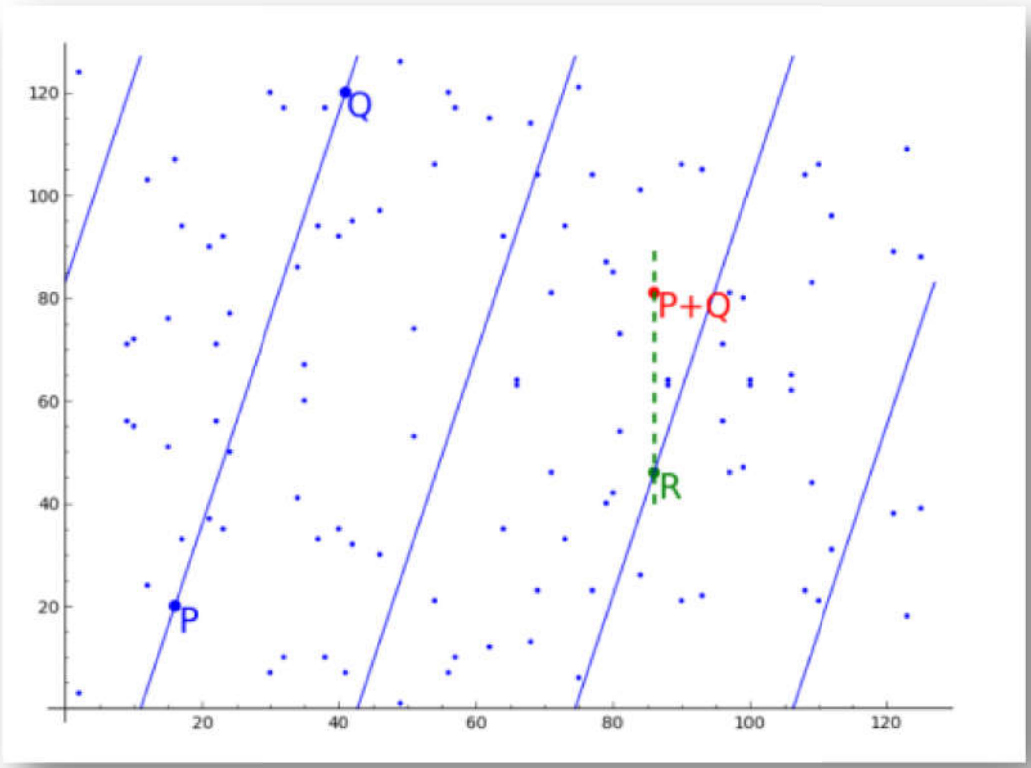



Figure II. 4:– EC sur $Z/pZ$ [7].

Soit E la courbe elliptique définie sur $Fq$ , avec $q = p > 5$ premier ou $q = 2n$ définie comme ci-dessus [16].

- L'ensemble des points de E forme un groupe de taille (borne de Hasse-Weil)

$$q + 1 - 2\sqrt{q} \leq \#E \leq q + 1 + 2\sqrt{q}$$

- Si P est un point de E d'ordre *l* (i.e *lP* = *O*), avec *l* un nombre premier, on utilise le groupe
  ***G = {P, 2P, .., lP = O}.***
- En pratique, on utilise les courbes E telles que **:**
  ***#E = ml,***
- avec *l* un grand nombre premier (de taille 160 bits environ) et *m* petit pour la cryptographie.

## 2.5 Calcul du nombre de points d'une courbe elliptique sur un corps fini

Soit E une courbe elliptique d'équation (2.1) sur *Fp* (avec p premier). Le nombre de points de E est donné par :

$$\#E = 1 + \sum_{x \in Fp}\left(1 + \left(\frac{x3+ax+b}{p}\right)\right)$$

Si le corps de base est ni, la courbe elliptique est un groupe ni et le théorème suivant donne des renseignements très utiles sur son ordre [17].

## 2.6 Problème du logarithme discret

Soit G un groupe (noté additivement) cyclique ni d'ordre N engendré par un élément P. Soit Q un élément de G. Comme G est un groupe cyclique engendré par P, il existe un unique entier k compris entre 1 et N tel que Q = kP. Cet entier k est appelé le logarithme discret de Q en base P noté $\log_P(Q)$ [10]. Cependant, le groupe que nous utiliserons par la suite est usuellement noté additivement et nous garderons donc les notations initiales. Notons bien que kP signifie P + P + ... + P, k fois où le signe "+ "est la loi du groupe G. La multiplication scalaire d'un point par un entier est l'opération de base et l'opération la plus coûteuse des protocoles basés sur les courbes elliptiques. Pour réduire le temps de calcul, on peut utiliser des algorithmes très connus d'exponentiation pour le calcul de [k]P pour un k grand. On peut également améliorer le calcul pour un k petit.

## 2.7 Utilisation des courbes elliptiques dans la cryptographie

### 2.7.1 Cryptage des messages

Il faudra ici encoder le texte clair m comme un point *Pm* de coordonnées x et y. C'est ce point qui sera chiffré. Il faut aussi rendre publique un point G et un groupe elliptique E*q*(*a, b*). Les utilisateurs doivent également choisir une clé privée et générer la clé publique correspondante. Voyons en premier temps comment encoder le texte clair comme un point sur la courbe.

**Table de correspondance**

Cette méthode consiste à ce que les parties communicantes partagent une table de correspondance, où chaque caractère est équivalent à un point sur la courbe.

| * | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| ∞ | (5,25) | (1,30) | (21,32) | (7,25) | (25,12) | (4,28) | (0,34) | (16,17) |
| I | j | k | l | m | n | o | p | q |
| (15,26) | (27,32) | (9,4) | (2,24) | (26,5) | (33,14) | (11,17) | (31,22) | (13,30) |
| r | s | t | u | v | w | x | y | z |
| (35,21) | (23,7) | (10,17) | (29,6) | (29,31) | (10,20) | (23,30) | (35,16) | (13,7) |

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| (31,15) | (11,20) | (33,23) | (26,32) | (2,13) | (9,33) | (27,5) | (15,11) | (16,20) | (0,3) |
| # | @ | ! | & | $ | % | | | | |
| (4,9) | (25,25) | (7,12) | (21,5) | (1,7) | (5,12) | | | | |

Figure II. 5: Exemple de table de correspondance pour $E : y^2 = x^3 + 2x + 9 \pmod{37}$[18].

- **Méthode de Koblitz pour l'encodage du texte clair**

Cette méthode est différente de la précédente. Elle est composée de plusieurs étapes :

- **Étape 1 :** Choisir une courbe elliptique $E_p(a, b)$.
- **Étape 2 :** On suppose que E à N points.

- **Étape 3 :** On suppose que l'alphabet est constitué des nombres 0,1,2,3,4,5,6,7,8,9 et des lettres A,B,C,...X,Y,Z codés en forme de points .
- **Étape 4 :** Cela converti notre message à une série de points entre 0 et 44.
- **Étape 5 :** Choisir un paramètre auxiliaire $k$ = 20 (les deux parties communicantes doivent se mettre d'accord sur son choix).
- **Étape 6 :** Pour chaque nombre m, on calcule $x = mk + 1$ et on essaye de trouver le correspondant y sur la courbe.
- **Étape 7 :** Si on ne trouve pas y, on essaye $x = mk + 2$ et $x = mk + 3$ jusqu'à trouver y.
- **Étape 8 :** En pratique, on va trouver y avant $x = mk + k - 1$. Puis on prend le point $(x, y)$ et on le considère comme m sur E. De cette manière, le message devient une suite de points sur la courbe.

Notons que la probabilité de ne pas trouver y (et donc ne pas pouvoir associer m à un point) est de $1/2^k$[19].

➢ **Décodage**

Pour retrouver $m$, il suffit de calculer $(x - 1)/y$ et de prendre le plus grand entier inférieur à cette valeur.

**Exemple** Soit une EC tq : p(751), a(-1), b(188), n(727).

- On souhaite convertir le caractère 'B' vers un point sur la courbe.
- 'B' est représenté par le nombre 11.
- $x = mk + 1$ i.e $11 * 20 + 1 = 221$ n'a pas de solution tq : $\boldsymbol{y^2 = x^3} + ax + b \bmod p$ .
- De même, pas de solution pour $x = mk + 2$ et $x = mk + 3$.
- Pour $x = mk + 4$, $x = 224$ et y existe ($y = 248$).
- Le point (224, 248) est donc crypté et décrypté en tant que message.

## 2.7.2 Protocole d'échange de clés de Diffie-Hellmann

On prend un point de départ $G(x_1, y_1)$ dans $E_q(a, b)$ dont l'ordre n est élevé. L'ordre n d'un point sur une EC est le plus petit entier positif tel que nG = O. L'échange d'une clé par ECC entre deux entités A et B se déroule comme suit :A choisit un $n_A$ inférieur à n qui sera sa clé privée. A génère alors sa clé publique $P_A = \boldsymbol{n_A} * G$.

- B choisit un $n_B$ inférieur à n qui sera sa clé privée. B génère alors sa clé publique $P_B = n_B * G$.
- A génère la clé secrète $K = n_A * P_B$ et B génère la clé secrète $K = n_B * P_A$.

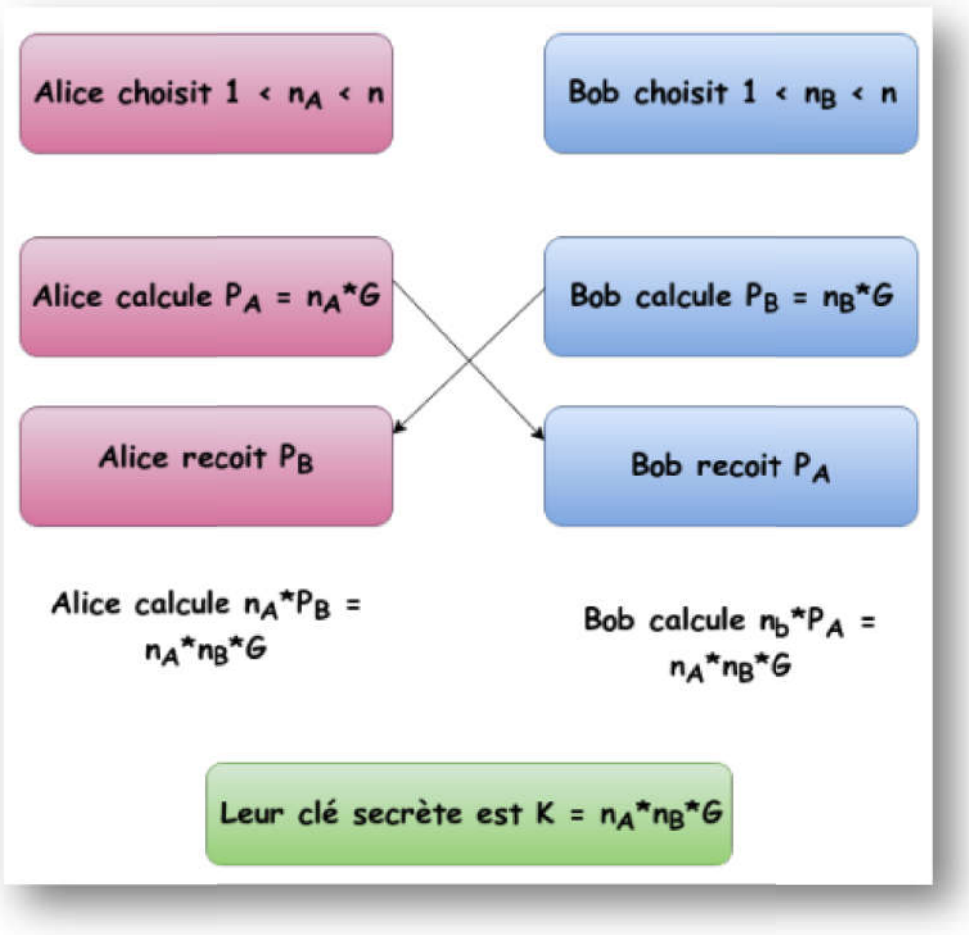



Figure II. 6: Échange de clés de Diffie Hellman (ECDH) [20].

Notons que $K$ peut être utilisé comme clé secrète pour la cryptographie symétrique (AES,..) [20].

### 2.7.3 Cryptage des points

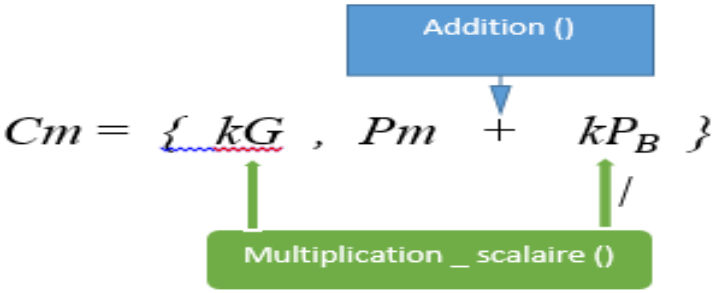
Pour utiliser les courbes elliptiques en cryptographie, il faut trouver un problème difficile (tel que la factorisation d'un produit en ses facteurs élémentaires dans le cas du RSA).

- **Considérons l'équation :**
  - $Q = K_p$
  - **où** $Q, P \in E_p(a, b)$ **et** $k < p$.

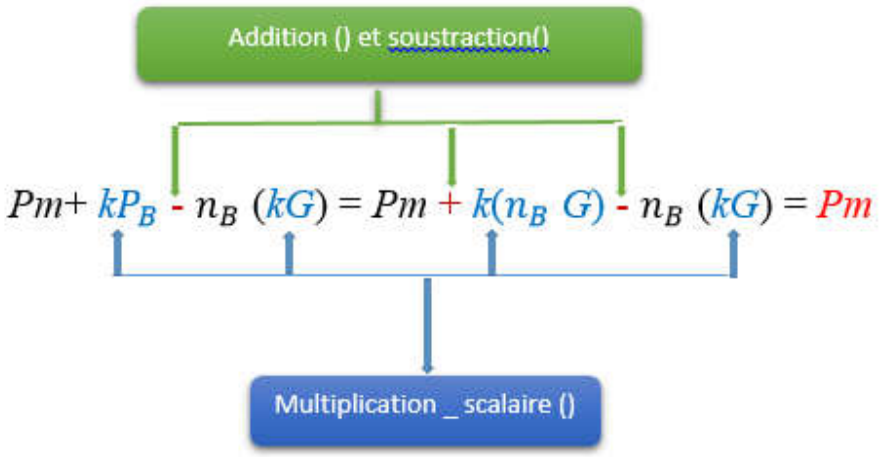
Il est facile de calculer $Q$ connaissant $k$ et $P$. Mais il est difficile de déterminer $k$ si on connaît $Q$ et $P$. Il s'agit du problème du logarithme discret pour les courbes elliptiques : $log_p(Q)$.

### 2.7.4 Chiffrer et déchiffrer de message

Pour chiffrer un message, *On* détermine aléatoirement un nombre entier positif $k$ et on définit *Cm* comme un couple de points tel que :




Pour déchiffrer, *B* devra multiplier le premier point par sa clé privée, et soustraire le résultat du second point reçu [20] :




### 2.7.4.1 Signification du symbole

**G :** point de base.

**Pm :** message clair.

**$P_B$ :** Clé publique de b

**Cm** : message chiffré.

### 2.7.4.2 Exemple Chiffrer et déchiffrer de message

Considérons une courbe elliptique dont l'équation est$y^2 = x^3 - 5x^2 + 3$. Le graphique de la fonction est présenté dans la Figure II. 1.

Considérons maintenant une courbe elliptique$(y^2 = x^3 - 5x^2 + 3)_{\bmod 37}$, $E_{37}(2,9)$. Les points sur la courbe elliptique[18].

Soit C = (9,4)

Alice sélectionne un nombre aléatoire α = 5,tout point A = (10,20) sur la courbe elliptique. Elle calcule:

A 1 = α (C+A) = Multiplication_scaliare (α, addition(C, A))= = (24,15),

A 2 = α A= Multiplication_scaliare (α,A)= = (13,40).

Elle conserve le nombre aléatoire α = 5 et le point A de la courbe elliptique comme clés secrètes et publie les symboles A 1 et A 2 comme clés publiques.

Bob sélectionne β = 7, B = (11,20) sur la courbe elliptique. Il calcule

B 1 = β (C + B)= multiplication_scalaire (β, addition(C, B))= (11,17)

B 2 = β B = Multiplication_scaliare (β,B)= (23,30).

Il conserve le nombre aléatoire β = 7 et le point B de la courbe elliptique comme clé secrète et publie B 1 et B 2 comme clé publique.

Alice calcule A B = α B 2 = (15,11) et Bob calcule B A = β A 2 = (2,13). Alice publie A B en tant que clé publique spécifique pour Bob et Bob publie B A en tant que clé publique spécifique pour Alice.

- **Chiffrer du message**

  Si Bob souhaite communiquer le message «attaque» à Alice, il convertit tous les caractères du message en points sur les courbes elliptiques à l'aide de la table de codes convenue qui est présentée à la Figure II.5 .

  - Dans le message "attaque", le premier caractère "a" correspond au point (5,25) utilisant la table de codes. Bob sélectionne un nombre aléatoire γ = 8 pour chiffrer le caractère ‘a’. Ensuite, le point (5,25) est crypté en tant que

    **E1** = γ C = (1,30), ce qui correspond au caractère «b» du tableau de conversion.

    **E2** = M + (β + γ) A 1 – γ A 2 + A B = (2,13) ce qui correspond à «5» dans le code

    Donc, le caractère "a" dans le texte brut est chiffré en deux caractères {b, 5} dans le texte chiffré.
  - ‘t’est un point (10,17) dans la table de codes. Soit γ = 12

    **E1** = (21,32), ce qui correspond à «c» dans la table de codes.

    **E2** = M + (β + γ) A 1 - γ A 2 + A B = (2,24) ce qui correspond à «l» dans le code

    Donc, "t" est crypté comme {c, l}.
  - ‘t’estun point (10,17) danslatable de codes. soit γ = 19.

    **E1** = (4,9), ce qui correspond à «#» dans la table de codes.

    **E2** = M + (β + γ) A 1 - γ A 2 + A B = (27,32) ce qui correspond à «j» dans le code

    Donc "t" est crypté comme {#, j}.
  - ‘a’ est un point (5,25) dans la table de codes. Soit γ = 2

**E1** = (29,31), ce qui correspond à «v» dans la table de codes.

**E2** = M + (β + γ) A 1 - γ A 2 + A B = (1,30) ce qui correspond à «b» dans le code

Donc, "a" est crypté comme {v, b}

- ‘q’ est un point (13,30) dans la table de codes. Soit γ = 3.

  **E1** = (1,30), ce qui correspond à «b» dans la table de codes.

  **E2** = M + (β + γ) A 1 - γ A 2 + A B = (31,22) qui correspond à «p» dans le code

Donc, ‘q’ est crypté avec {b, p}.

**E1** = (25,25), ce qui correspond à «@» dans la table de codes.

**E2** = M + (β + γ) A 1 - γ A 2 + A B = (4,28) qui correspond à «f» dans le code

- **‘**e’ est un point (25,12) dans la table de codes. Soit γ = 23

  **E1** = (25,25), ce qui correspond à «@» dans la table de codes.

  **E2** = M + (β + γ) A 1 - γ A 2 + A B = (4,28) qui correspond à «f» dans le code

  Donc, "e" est crypté comme {@, f}.

Donc le message chiffre «attaque» = {b,5; c,l; #,j; v,b; b,p; @,f }

- **déchiffrer de message**

  Alice après avoir reçu le texte chiffré {b, 5; c, l; #, j; v, b; b, p; @,f }

  Convertit les caractères chiffrés en points (1,30), (2,13), (21,32), (2,24) (4,9), (27,32) (29,31) (1,30) (1,30) (31,22) (25,25) (4,28). Elle déchiffre le message en prenant deux points à la fois comme les points E1 et E2.

  - **M = E2 - (αE1 + α B 1 + B A) = (5,25)** qui correspond au caractère "a" dans la table de codes.
  - **M = E 2 - (αE1 + α B 1 + B A) = (10,17),** ce qui correspond au caractère «t» dans la table de codes.
  - **M = E2 - (αE1 + α B 1 + B A) = (10,17)**, ce qui correspond au caractère «t» dans la table de codes.

- **M = E2 - (αE1 + α B 1 + B A) = (5,25)** qui correspond au caractère "a" dans la table de codes.
- **M = E2 - (αE1 + α B 1 + BA) = (13,30)** qui correspond au caractère «q» dans la table des codes.
- **M = E2 - (αE1 + α B 1 + B A) = (25,12)**qui correspond au caractère «e» dans le champ

Donc “attaque" est le message original.

## 2.8 Attaques sur la cryptographie à base des courbes elliptiques

Puisque le cryptage des messages avec des courbes elliptiques se base sur la difficulté de résoudre le problème du logarithme discret en un temps raisonnable généralement, il est important de savoir dans quels cas nous pouvons le résoudre rapidement pour éviter ces cas-là.

### 2.8.1 L’attaque MOV

Les travaux menés en 1993 par Menezes, Okamoto et Vanstone, qui ont donné leurs noms à l’attaque MOV, montrent que le problème du logarithme discret sur $E(F_p)$ peut être ramené à celui du logarithme discret dans $(F_{pk})_*$. En particulier dans le cas de courbes super singulières, il peut être ramené à celui du logarithme discret sur $F_{pk}$ avec $k \in \{1, 2, 3, 4, 6\}$, généralement $k = 2$. Cela induit un risque car la réduction à un tel problème se fait en temps polynomial en $O$ ($\mathbf{log_P}$), et la résolution du problème du logarithme discret est alors réalisable par un algorithme probabiliste sous-exponentiel [10].

### 2.8.2 Force Brute

L’attaque par force brute consiste à calculer $P$, $2P$, ..., jusqu’à ce qu’on trouve $k$. Au pire des cas n l’attaque nécessite $n$ opérations (l’ordre de la courbe).
En 1996 une attaque sur un système cryptographique basé sur les EC de clé de 120bits a été esquissée et réalisée après 3 ans. L’attaque était basée sur un matériel dédié

qui peut faire 25 million d'opérations en parallèle. Le coût de construire cette machine était de $10 million et le temps nécessaire pour obtenir la clé était 32 jours. [22]

### 2.8.3 Baby Step Giant Step

Cette méthode, développée par D. Shanks, fait environ $\sqrt{N}$ pas et stocke environ $N$ données[5].

Nous supposerons qu'il existe un nombre entier $k$ tel que $Q = kP$ avec $P, Q \in E(F_q)$ et que $N$ , l'ordre de E, est connu.

L'algorithme se déroule comme suit :

- ✓ Choisir un entier $m \geq \sqrt{N}$ et calculer mP .
- ✓ Calculer et stocker dans une liste les iP pour $0 \leq i < m$.
- ✓ Calculer les points Q - jmP pour j = 0, 1, ..., m - 1 jusqu'à ce qu'un de ces éléments correspondent à un iP de la liste précédente.
- ✓ Si iP = Q - jmP , nous avons Q = kP avec $k \equiv i + jm \pmod N$.

Puisque $\boldsymbol{m^2 > N}$ **,** nous avons $\boldsymbol{0 \leq k < m^2}$. Ecrivons $\boldsymbol{k = k_0 + mk_1}$, ainsi $\boldsymbol{k \equiv k_0 \pmod m}$ avec $\boldsymbol{0 \leq k_0 < m}$et $\boldsymbol{k_1 = (k - k_0)/m}$et donc $\boldsymbol{0 \leq k_1 < m.}$ Posons $\boldsymbol{i = k_0}$ *et* $\boldsymbol{j = k_1}$, nous obtenons donc :

$$\boldsymbol{Q - k_1 mP = kP - k_1 mP = k_0 P}$$

Est la relation voulue.

**Exemple :** Soit $G = E(F_41)$, où E est donné par :

$$\boldsymbol{y^2 = x^3 + 2x + 1.}$$

Soient $P = (0, 1)$ et $Q = (30, 40)$. Par le théorème de Hasse, nous savons que l'ordre de G est au plus 56, posons $m = 8$. Les points $iP$ pour : $\boldsymbol{0 \leq i \leq 7}$ **sont :**

**(0, 1), (1, 39), (8, 23), (38, 38), (23, 23), (20, 28), (26, 9)**

Calculons $Q - jmP$ pour : $j = 0, 1, 2$

(30, 40), (9, 25), (26, 9), où nous nous arrêtons puisque le troisième point correspond à $7P$ . Nous avons donc

$\boldsymbol{Q = (7 + 2 * 8)P = 23P}$

Et nous trouvons $k = 23$.

## 2.9 Utilisation de la cryptographie à base des courbes elliptiques

### 2.9.1 Comparaison de performance

Afin d'évaluer et de comparer la performance de RSA et ECC, 2 implémentations utilisant respectivement ces 2 cryptosystèmes sont testés dans [23]. Comme nous avons présenté au début du chapitre, ECC peut avoir le même niveau de sécurité que RSA avec une clé beaucoup plus courte. Les longueurs de clé utilisées sont suggérées dans le tableau II. 7) [24], et les longueurs qui se situent dans la même colonne sont censées pouvoir fournir le même niveau de robustesse. D'ailleurs, l'algorithme de signature utilisé pour ECC est ECDSA, une variante de DSA conçue pour les courbes elliptiques, et un texte de longueur 100 Ko est utilisé pour tester la signature.

| Algorithme | Longueur de clé (bit) | | | | |
|---|---|---|---|---|---|
| ECC | 163 | 233 | 283 | 409 | 571 |
| RSA | 1024 | 2240 | 3072 | 7680 | 15360 |

Figure II. 7:Longueur de clé en bit [23].

Les résultats de test sont donnés dans les tableaux II. 8 , II. 9 et II. 10. Nous pouvons constater que pour atteindre le même niveau de sécurité, premièrement, la consommation de mémoire est beaucoup moins importante avec ECC, car nous utilisons des clés plus courtes. Deuxièmement, les calculs de la génération de clé et de la signature de message sont plus rapides avec ECC. La vérification de signature est plus rapide avec RSA, car il suffit d'effectuer une exponentiation modulaire.

| Longueur de clé | | Temps de calcul (s) | |
|---|---|---|---|
| ECC | RSA | ECC | RSA |
| 163 | 1024 | 0.08 | 0.16 |
| 233 | 2240 | 0.18 | 7.47 |
| 283 | 3072 | 0.27 | 9.80 |
| 409 | 7680 | 0.64 | 133.90 |
| 571 | 15360 | 1.44 | 679.06 |

Figure II.8:Temps d'exécution pour la génération de clés

Le même type de test a été effectué dans [32, 26], les résultats de test ont aussi prouvé que ECC était plus avantageux en terme de consommation de mémoire et vitesse de calcul. C'est la raison principale pour laquelle ECC devient de plus en plus le choix préféré pour les systèmes embarqués qui disposent d'une mémoire et d'une puissance de calcul très limitée.

Actuellement RSA demeure toujours le cryptosystème asymétrique le plus largement utilisé, car il est sorti beaucoup plus tôt par rapport à ECC. RSA est publié en 1978 [27] et standardisé en 1993 [28], tandis que ECC est proposé dans les années 80 [29, 30] et standardisé vers la fin des années 90. Les 2 cryptosystèmes sont initialement protégés.

| Longueur de clé | | Temps de calcul (s) | |
|---|---|---|---|
| ECC | RSA | ECC | RSA |
| 163 | 1024 | 0.15 | 0.01 |
| 233 | 2240 | 0.34 | 0.15 |
| 283 | 3072 | 0.59 | 0.21 |
| 409 | 7680 | 1.18 | 1.53 |
| 571 | 15360 | 3.07 | 9.20 |

Figure II.9: Temps d'exécution pour la signature numérique [23].

| Longueur de clé | | Temps de calcul (s) | |
|---|---|---|---|
| ECC | RSA | ECC | RSA |
| 163 | 1024 | 0.23 | 0.01 |
| 233 | 2240 | 0.51 | 0.01 |
| 283 | 3072 | 0.86 | 0.01 |
| 409 | 7680 | 1.80 | 0.01 |
| 571 | 15360 | 4.53 | 0.03 |

Figure II. 10 :Temps d'exécution pour la vérification de signature [23].

Par des brevets. Du fait que le brevet de RSA a expiré depuis 2000, il peut être utilisé librement par tout le monde, mais celui de ECC est toujours valable [31]. Une autre raison qui limite l'utilisation de ECC est que mathématiquement RSA est relativement plus simple. De nombreuse spécifications de ECC existent [32], mais les implémentations sont souvent incomplètes, c'est-à-dire que seulement quelques courbes décrites dans les spécifications sont implémentées.

### 2.9.2 Applications qui utilisent la EC

De nos jours, plusieurs protocoles et plateformes utilisent la cryptographie à base des courbes elliptiques à cause de sa rapidité et sa sécurité. On cite :

- TLS Le protocole de sécurisation d'échange sur internet (Transport Layer Security) utilise la ECC dans la phase "Handshake" pour l'agrément sur la clé à utiliser pour chaque session.
- Smart Cards A coté de RSA, les nouvelles smart-card support la ECC.
- Passport Biométrique La ECC est utilisée pour construire la clé de session.
- Bitcoin La ECC est utilisée pour la signature électronique pendant les transactions des paiement avec Bitcoin.

# 3 Conclusion

Dans ce chapitre nous avons exploré la théorie des courbes elliptiques et le problème du logarithme discret, qui est la base de la cryptographie sur des courbes elliptiques. Nous avons choisi la courbe elliptique adéquate afin d'avoir un cryptosystème simple. Ce choix est primordial.

Nous avons vu des méthodes de codage et de décodage basées sur des courbes elliptiques qui reposent beaucoup sur la multiplication numérique et que nous présenterons au chapitre 3 avec des moyens de les améliorer.

Nous avons vu les différentes attaques connues sur les courbes elliptiques et que, malgré que la théorie derrière la cryptographie à base des courbes elliptiques soit solide, la mauvaise implémentation sur machine peut rendre une attaque par canaux cachés très facile, d'où il faut bien prendre ce point en compte lors du développement.

Nous avons exploité aussi les différentes applications et protocoles de nos jours qui utilisent la cryptographie à base des courbes elliptiques. Il a été possible de comparer cette méthode de chiffrement avec le RSA. Les résultats on démontrés la rapidité et la sécurité de la cryptographie à base des courbes elliptiques par rapport à RSA avec une petite clé.

**Chapitre *III :***

# LES CHOIX OPTIMAUX DE LA MULTIPLICATION SCALAIRE

# 1 Introduction

Un des avantages d'ECC est la rapidité de calcul car nous utilisons des clés plus courtes. l'ECC est souvent un choix adapté pour les matériels qui ne disposent que d'une mémoire et d'une puissance de calcul limitées (carte à puce, micro-contrôleur etc.). Comme nous avons vu dans le chapitre précédent que l'opération la plus complexe sur les courbes elliptiques est la multiplication scalaire, elle est aussi fréquemment utilisée dans les protocoles cryptographiques. La performance de la multiplication scalaire a une forte influence sur la performance de l'ensemble de cryptosystème ECC.

Dans ce chapitre nous allons voir de différentes techniques mathématiques et algorithmiques qui nous permettent d'accélérer le calcul de multiplication scalaire.

# 2 Définition de la multiplication scalaire

La multiplication scalaire est une opération complexe qui contient un ensemble d'additions et un ensemble de doublements des points sur une courbe elliptique.

Chaque addition ou doublement contient aussi un ensemble d'opérations arithmétiques (Multiplication modulaire, addition/soustraction modulaire et l'inversion modulaire) comme montre la figure ci-dessous :

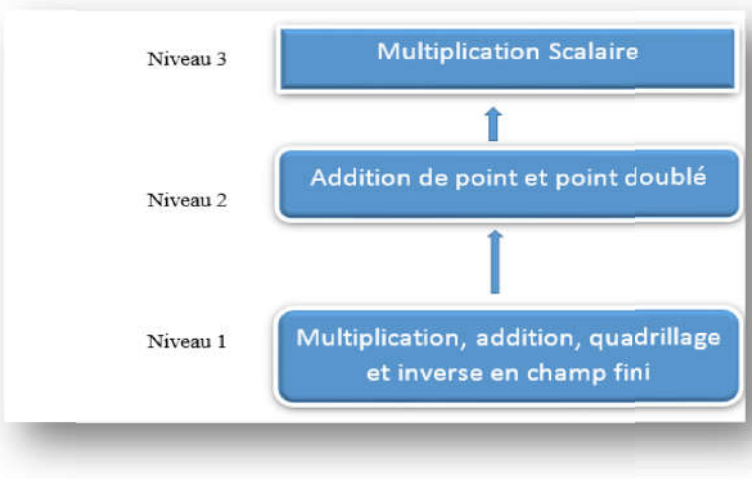



Figure III. 1 **:** Le modèle à trois niveaux pour la multiplication scalaire.

## 2.1 Les méthodes existantes de la multiplication scalaire

L'opération la plus coûteuse dans ECC est la multiplication scalaire. De point de vue de la mise en œuvre, nous devons développer des algorithmes peu coûteux en ce qui concerne son temps. Nous pouvons définir la multiplication scalaire comme suit :d × P = P + · · · (d fois) · · · + P, où P soit tout point sur E et d est un entier positif. Le calcul est connu sous le nom d'opération de multiplication scalaire [33]. Deux versions de l'algorithme binaire sont définies ci-dessous pour calculer la multiplication scalaire. Où ADD et DBL désignent l'addition de deux pointset le doublement des points respectivement. La fonction ADD calcule P + Q en parlant en entrée de deux points P et Q. De la même manière, DBL renvoie 2P calcule sa double prise 2P en entrée P.

Dans cette section, différents types de méthodes de multiplication seront expliqués sur la courbe elliptique. Lorsque nous devons calculer une courbe ellipsoïdale, nous devons avoir un état connu P. Cela signifie que nous disposons d'informations préalables sur le point de la courbe elliptique qui atteindra le nombre standard. L'utilisation des informations EC avancées par la méthode binaire de droite à gauche, méthode binaire de gauche à droite, mode binaire, méthode de fenêtre ou méthode de fenêtre glissante. Ces méthodes seront décrites.

### 2.1.1 Algorithmes binaires simples

Les deux formules (1.30) et (1.29) donnent deux manières de calculer la multiplication scalaire à partir de la décomposition binaire du scalaire. Les algorithmes correspondants sont similaires aux algorithmes classiques d'exposant carré et multiplication et sont appelés doubleand-add en raison de la structure de groupe additive.

$$kP = \mathrm{k}_0 P + \mathrm{k}_1 2P + \text{.....} + k_{l-1} 2^{l-1} P \qquad (1.29).$$

$$kP = k_0 P + 2\ (k_1 P + \text{.....} + 2\ (k_{l-1} P) \text{.....})) \qquad (1.30).$$

L'algorithme suivant présente la méthode Double and Add.

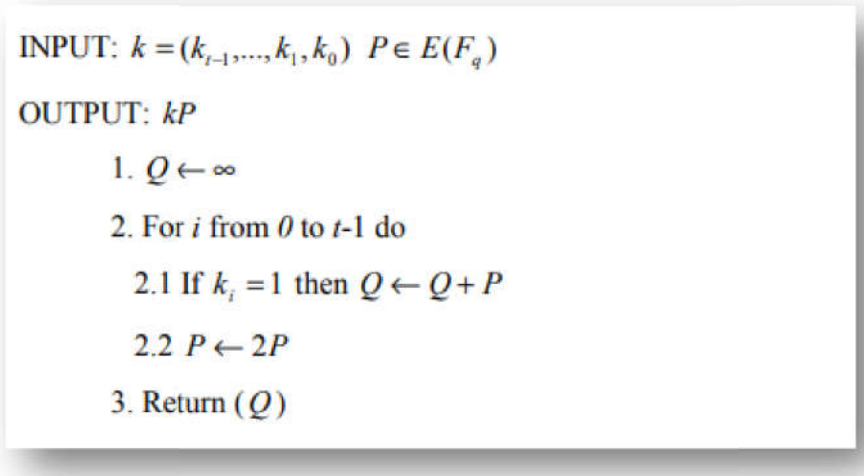



Figure III. 2 : algorithme 3.1 binaire de droite à gauche pour la multiplication de points.

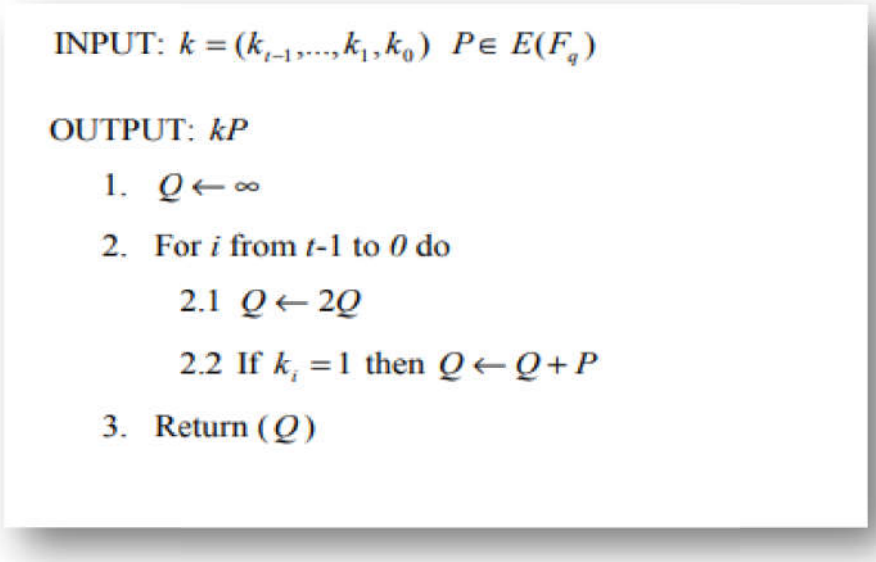



Figure III. 3:algorithme 3.2 binaire de gauche à droite pour la multiplication de points.

Le coût de la multiplication en méthode binaire dépend du nombre d'éléments non nuls et de la longueur de la représentation binaire de k. la méthode binaire nécessite (L- 1) le double de point set (W-1) additions où L'est la longueur du développement binaire de k et W le poids de Hemming du k qui est le nombre d'éléments non nuls en expansion de k[34] .

Par exemple, si k = 629 = (1001110101) 2, il faudra (W-1) = 6 - 1 =5 additions et L- 1 = 10 -1 =9 opérations de doublement.

Puisque nous allons nous intéresser au temps d'exécution des algorithmes de multiplication, nous pouvons prédire le temps d'exécution en regardant les étapes de l'algorithme. Le scalaire k a t bits et il est prévu que t / 2 nombre de uns soit dans la représentation binaire de k. Ainsi, Alg.3.1 et Alg.3.2peuvent avoir la durée de fonctionnement de $\frac{t}{2}$ A + tD où A représente l'addition de points de courbe elliptique et D, les temps de doublement de points de courbe elliptique.

La soustraction de points sur une courbe elliptique est aussi efficace que l'addition. Ainsi, une représentation numérique signée peut être utilisée pour k scalaire. Une forme utile de chiffre signé est la forme non adjacente (NAF). Une forme non adjacente (NAF) d'un entier positif k est une expression k= $\sum_{i=0}^{l-1} k_i 2^i$

La représentation NAF est notée NAF (k) et dans cette représentation, aucun des bits consécutifs n'est non nul. NAF (k) a moins de chiffres non nuls que toute représentation signée de k. La longueur de NAF (k) est presque un chiffre de plus que la longueur de la représentation binaire de k. Le point attiré de cette représentation est la densité de chiffres non nuls. C'est environ 1/3 [35] . Maintenant, nous allons d'abord décrire comment représenter un nombre représenté binaire dans NAF, puis nous substituerons cette forme à scalaire k dans les méthodes de multiplication de courbes elliptiques.

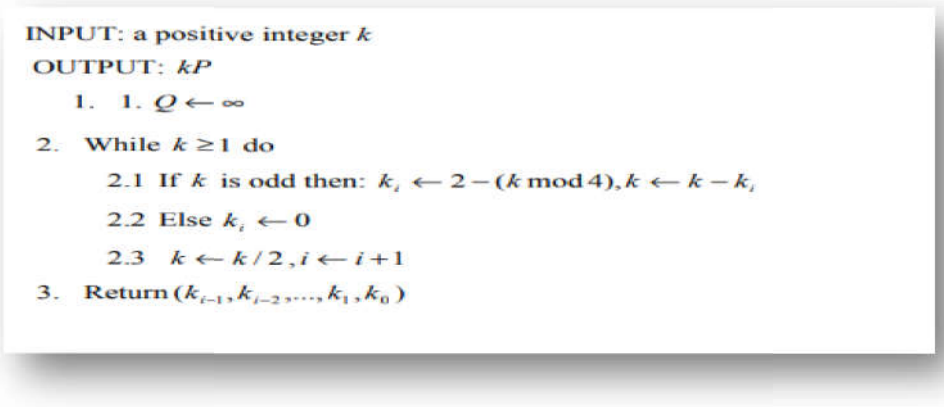

INPUT: a positive integer $k$
OUTPUT: $kP$
1. 1. $Q \leftarrow \infty$
2. While $k \geq 1$ do
2.1 If $k$ is odd then: $k_i \leftarrow 2-(k \bmod 4), k \leftarrow k-k_i$
2.2 Else $k_i \leftarrow 0$
2.3 $k \leftarrow k/2, i \leftarrow i+1$
3. Return $(k_{i-1}, k_{i-2}, \ldots, k_1, k_0)$

Figure III. 4:Algorithme 3.3 Calcul de la forme non-adjacente d'un entier.

En utilisant un formulaire non adjacent, la méthode de gauche à droite peut être modifiée. La méthode de multiplication modifiée est la suivante.

INPUT: Positive integer $k$, $P \in E(F_q)$

OUTPUT: $kP$

1. Compute $NAF(k) = \sum_{i=0}^{l-1} k_i 2^i$
2. $Q \leftarrow \infty$
3. For i from $l$-1 to 0 do
   - 3.1 $Q \leftarrow 2Q$
   - 3.2 If $k_i = 1$ then $Q \leftarrow Q + P$
   - 3.3 If $k_i = -1$ then $Q \leftarrow Q - P$
4. Return $(Q)$

Figure III. 5:Algorithme 3.4 Calcul de NAF.

### 2.1.2 Méthodes de fenêtre

Si nous avons un peu plus de mémoire dans notre plate-forme de mise en œuvre, le temps d'exécution des algorithmes de multiplication peut être réduit en utilisant une méthode de fenêtre. Dans les méthodes de fenêtre, w nombres de k sont traités à la fois, de sorte que les temps d'exécution des algorithmes diminuent. La notation w signifie la largeur de la fenêtre. Avant d'entrer dans les méthodes de multiplication de points EC, nous voulons décrire une version de la représentation NAF appelée wNAF. Dans cette représentation, nous représentons le scalaire k par un nombre impair.

Le scalaire k a une représentation NAF unique en largeur-w, notée wNAF (k). La densité moyenne des chiffres non nuls de la représentation NAF en largeur-w est d'environ 1 / (w + 1) [36]. Le calcul de la largeur-w NAF la représentation est la suivante.

INPUT: Window width $w$, a positive integer $k$.

OUTPUT: $\mathrm{NAF}_w(k)$

1. $i \leftarrow 0$
2. While $k \geq 1$ do
   - 2.1. If $k$ is odd then: $k_i \leftarrow k \bmod 2^w, k \leftarrow k - k_i$
   - 2.2. Else $k_i \leftarrow 0$
   - 2.3. $k \leftarrow k/2, i \leftarrow i+1$
3. Return $(k_{i-1}, k_{i-2}, \ldots, k_1, k_0)$

Figure III. 6**:** Algorithme 3.5 Calcul de la forme non-adjacente de largeur *w* d'un entier [36].

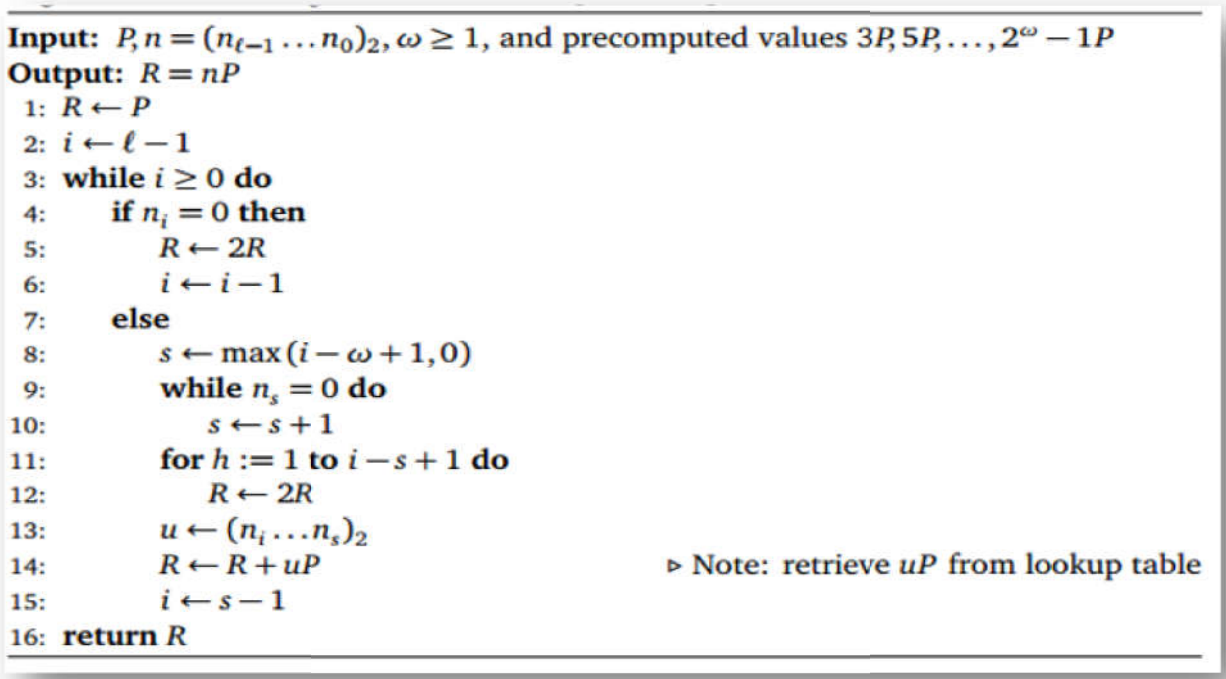
On peut donc maintenant donner un algorithme de multiplication par un scalaire semblable à l'algorithme 3.1, tirant cette fois parti de la forme adjacente de largeur *w* du scalaire :

Input: An integer $d = \Sigma_{j=0}^{n-1} d_j 2^j$ in $w$-NAF
Output: $dP$
pre-compute $X_j \leftarrow iX$ , $i$ in $[-2^{w-1}, 2^{w-1} - 1]$, $j$ is odd
$Y \leftarrow X$
**for** $j = n - 1$ to 0 **do**
  $Y = 2Y$
  **if** $d \neq 0$ **then**
    **if** $d_j > 0$ **then**
      $Y = Y + X_{d_j}$
    **else**
      $Y = Y - X_{d_j}$
    **end if**
  **end if**
**end for**
**return** $Q$

Figure III. 7: Algorithme 3.6 Doublement et addition avec fenêtres [36].

**Algorithme de fenêtre glissante (Sliding window)**

Dans la méthode « sliding window », la fenêtre commence par le bit le plus à gauche et se déplace vers la droite jusqu'à la plus à gauche du octet est 1. La valeur à extraire de la table de recherche est la plus grande sous-chaîne parmi celles-ci octet se terminant par un 1 à droite. Ainsi, la table de recherche ne doit contenir que des multiples impairs de P. L'algorithme 3.7décrit la méthode de la « sliding window »[37].



**Input:** $P, n = (n_{\ell-1} \dots n_0)_2, \omega \geq 1$, and precomputed values $3P, 5P, \dots, 2^{\omega} - 1P$
**Output:** $R = nP$

```
1: R ← P
2: i ← ℓ − 1
3: while i ≥ 0 do
4:     if n_i = 0 then
5:         R ← 2R
6:         i ← i − 1
7:     else
8:         s ← max(i − ω + 1, 0)
9:         while n_s = 0 do
10:            s ← s + 1
11:        for h := 1 to i − s + 1 do
12:            R ← 2R
13:        u ← (n_i ... n_s)_2
14:        R ← R + uP          ▷ Note: retrieve uP from lookup table
15:        i ← s − 1
16: return R
```

Figure III. 8**:** Algorithme 3.7 Méthode de la fenêtre glissante (‘sliding window’)[4].

L'algorithme nécessite que la scalaire n soit exprimé dans une représentation binaire. Ainsi, n’est réécrit dans une représentation binaire sous la forme $(n_{11}n_{10}\dots n_0)_2$ =$100100101001_2$.

Pour décrire la manière dont les fenêtres sont déplacées et les valeurs réelles à extraire de la table de consultation, les bits ci-dessus sont regroupés comme suit :

100100101001

Où les parties soulignées soulignent les fenêtres et la valeur à rechercher.

Les 5 premiers bits (à gauche) sont $(n_{11}n_{10}n_9n_8n_7)_2$ = 10010. Le bit le plus à gauche est déjà 1, il n'est donc pas nécessaire de déplacer cette fenêtre. Le bit le plus à droite est 0, alors ignorez ce bit. Ainsi, la fenêtre à prendre en compte est $(n_{11}n_{10}n_9n_8)_2 = 1001_2 = 9$. Pour la première fenêtre, la valeur 9P est extraite de la table de recherche et affectée à R. Par conséquent, R devient 9P

Les deux bits suivants $n_7n_6$ sont 0. Ceci, R est doublé deux fois et devient

$R = 2^2 \cdot 9P = 36P$.

Les 5 bits suivants dont le bit le plus à gauche est 1 sont $n_5n_4n_3n_2n_1 = 10100_2$. Les deux bits les plus à droite sont 0, ainsi ces deux bits sont ignorés. Ainsi, la fenêtre à prendre en compte est $(n_5n_4n_3)_2 = 101_2 = 5$. La valeur R est doublée trois fois. La valeur 5P est extraite de la table de recherche et ajoutée à R. Par conséquent, R devient $2^3$*36P + 5P = 293P.

Les deux bits suivants $n_2n_1$ sont 0. Ainsi, R est doublé deux fois et devient

$R = 2^2 \cdot 293P = 1172P$.

### 2.1.3 Implémentation et Comparaison de performance de la multiplication scalaire

cette section, nous appliquons les méthodes de multiplication numériques décrites dans la section précédente en les appliquantsur un processeur Pentium 4 à 2,7 GHz. Avec une sélection de champ de 256 bit, le calcul de routine du domaine est écrit en Java avec la sélection des coordonnées de projection. Pour déterminer le temps nécessaire pour effectuer chaque méthode de la multiplication scalaire, Pour ce faire, nous suivons les étapes selon l'organigramme illustré dans la figure ci-dessous.

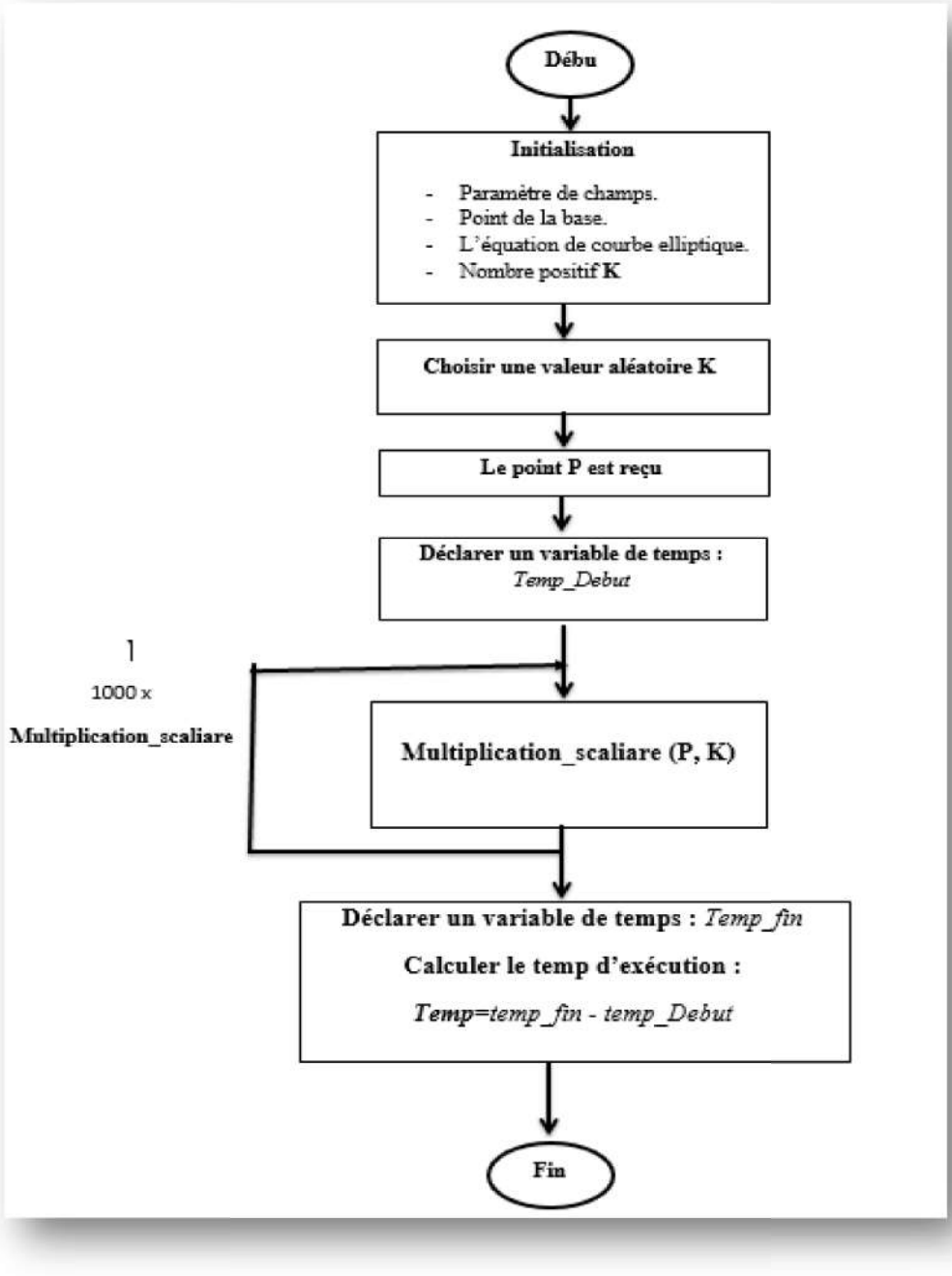



Figure III. 9: la procédure pour calculer le temps d'exécution de chaque méthode.

La figure 9 représente la procédure suivi pour calculer le temps d'exécution de chaque méthode de multiplication scalaire, tel que :

1. Donner une valeur préliminaire à chacun des :
    - Paramètre de champs : choisir un champ par exemple champ $\mathbf{F_{256}}$
    - Point de la base.
    - L'équation de courbe elliptique : pour définir les point de courbe elliptique il faut définie L'équation : $\mathbf{y^2} = \mathbf{x^3} + ax + b$

2. Choisir une valeur aléatoire k : est une valeur naturelle limitée par 3 et P.
3. Le point P est reçu : c'est une point qui représente un paramètre d'entré de la méthode Multiplication.
4. Déclarer un variable du temps temp_Debut : est de but de capter le temps de début d'exécution de la fonction.
5. Multiplication_scaliare (P, K) : est une fonction qui prends comme paramètre P et K, et retourner une point (le résultat de kP).
6. Déclarer un variable de temps Temp_fin : est de but de capter le temps de fin d'exécution de la fonction, et ensuite de calculerle temp d'exécution final (temp_Debut - Temp_fin).

### 2.1.4 Résultat de temps d'exécution de la multiplication scalaire

Grâce à l'expérience décrite ci-dessus, nous avons obtenu les résultats (présentés dans le tableau 3.6) des méthodes de multiplication scalaire de courbes elliptiques sur les champs premiers en général. Dans la conception du système cryptographique que nous allons utiliser en courbe elliptique cryptographie, nous devons choisir l'opération de multiplication la mieux adaptée à notre système. C'est à cause du grand rôle d'opération de multiplication de la courbe elliptique.

| Méthode de multiplication | Coordonnés de system | Résultats de temps (ms) |
|---|---|---|
| Right-to-Left | Affine | 4 |
| Left-to-Right | Affine | 2.43 |
| Binary NAF | Affine | 2.406 |
| Window NAF | Affine | 2 |
| Sliding Window | Affine | 1.81 |

Tableau III. 1:les résultats de temps (ms)pour les différentes méthodes de la multiplication dans le champ $F_{256}$.

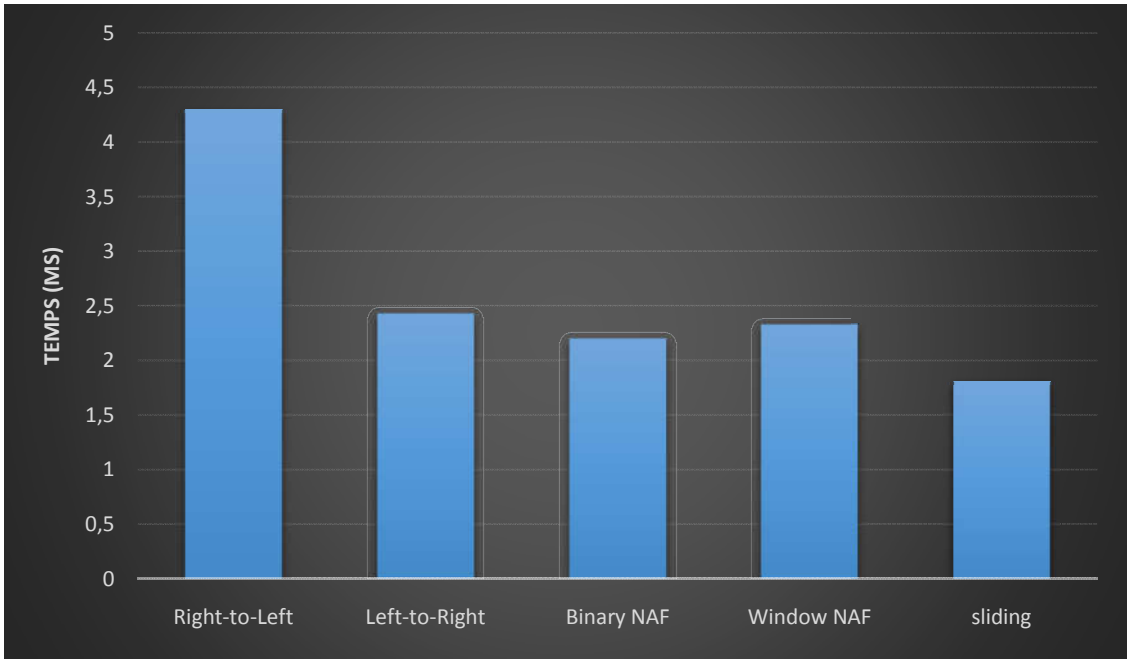



Figure III. 10 **:** présentation graphique de temps (ms) pour les différentes méthodes de la multiplication dans le champ $F_{256}$.

D'après le résultat illustré dans le tableau précédent, nous remarquons que la méthode sliding window a un temps d'exécution inferieur par rapport les autres méthodes, et alors la méthode sliding window est la plus adaptée pour implémenter la courbe elliptique de la cryptographie.

## 3 Les systèmes de coordonnées

Une courbe elliptique peut être représentée par plusieurs systèmes de coordonnées. Nous donnons ici les formules d'addition et de doublement pour les coordonnées affines [38], les

coordonnées projectives [39], les coordonnées Jacobienne [40 ,41] et les coordonnées Jacobienne de Chudnovski [40], ainsi que le nombre nécessaire d'opérations sur le terrain.

À partir de maintenant, nous supposons que$F_p$est un champ avec p> 3.

## 3.1 Les formules d'addition et doublement en coordonnées affine

Soit l'équation d'une courbe elliptique E sur $F_p$ :

$$E: y^2 = x^3 + ax + b \ (a;\ b \in F_p\ ;\ 4a^3 + 27b^2 \neq 0).$$

Les formules d'addition pour les coordonnées affines sont les suivantes.

Soit P =$(x_1;y_1)$, Q = $(x_2;y_2)$ et P + Q = $(x_3;y_3)$ soient des points sur E ($IF_p$).

- Formules d'addition des points en coordonnées affines (P 6 = ± Q)

$$x_3 = \lambda^2 - x_1 - x_2;\quad y_3 = \lambda(x_1 - x_3) - y_1; \text{où } \lambda = (y_2 - y_1) / (x_2 - x_1);$$

- Formules de doublement d'un point en coordonnées affines (P = Q)

$$x_3 = \lambda^2 - 2\,x_1;\quad y_3 = \lambda(x_1 - x_3) - y_1; \qquad \text{où } \lambda = (3x_1^2 + a) / (2y_1).$$

Nous discutons ici en détail des temps de calcul pour ces formules. Par souci de simplicité, nous négligeons l'addition, la soustraction et la multiplication par une petite constante dans $IF_p$ car elles sont beaucoup plus rapides que la multiplication et l'inversion dans$IF_p$. Notons le temps de calcul d'une addition (resp. D'un doublement) par t (A + A) (resp. T (2A)) et représentons la multiplication (resp. Inverse, resp. Au carré) dans Ifppar M (resp. I, resp. S). Alors nous voyons que t (A + A) = I + 2M + S et t (2A) = I + 2M + 2S.

## 3.2 Les formules d'addition et doublement en coordonnées projectives

Pour les coordonnées projectives, on pose x = X/ Z et y = Y / Z, donnant l'équation

$$E_p: Y^2Z = X^3 + aXZ^2 + b\,Z^3\ ;$$

Les formules d'addition en coordonnées projectives sont les suivantes :

Soit P = $(X_1; Y_1; Z_1)$, Q = $(X_2; Y_2; Z_2)$ et P + Q = R = $(X_3; Y_3; Z_3)$.

- Formules d'addition de courbes en coordonnées projectives (P 6 = ± Q)

$X_3 = vA$; $Y_3 = u\,(v^2X_1Z_2 - A) - v^3Y_1Z_2$;$Z_3 = v^3Z_1Z_2$;

Où $u = Y_2Z_1 - Y_1Z_2$; $v = X_2Z_1 - X_1Z_2$; $A = u^2Z_1Z_2 - v^3 - 2\,v^2X_1Z_2$;

- Formules de doublement de courbes en coordonnées projectives (R = 2P)

$$X_3 = 2hs;\ Y_3 = w\,(4B - h) - 8Y_1^2s^2;\ Z3 = 8s^3;$$

où $$w = aZ_1^2 + 3X_1^2;\ s = Y_1Z_1;\ B = X_1Y_1s;\ h = w^2 - 8B.$$

Les temps de calcul sont t (P + P) = 12M + 2S et t (2P) = 7M + 5S,

Où P signifie coordonnées projectives.

## 3.3 Les formules d'addition et doublement en Jacobienne et Jacobienne Chudnovski

Dans les coordonnées Jacobienne on a : $x = X/Z^2$ et $y = X/Z^3$, on obtient l'équation :

$$E_J : Y^2 = X^3 + aXZ^4 + b\,Z^6;$$

Les formules d'addition dans les coordonnées jacobiennes sont les suivantes :

Soit $P = (X_1; Y_1; Z_1)$, $Q = (X_2; Y_2; Z_2)$ et $P + Q = R = (X_3; Y_3; Z_3)$.

- Formules d'addition sur les courbes en coordonnées jacobiennes ($P \neq \pm Q$)

$X_3 = -H^3 - 2U_1H^2 + r^2$; $Y_3 = -S_1H^3 + r(U_1H^2 - X_3)$; $Z_3 = Z_1Z_2H$

où $U_1 = X_1Z_2^2$; $U_2 = X_2Z_1^2$; $S_1 = Y_1Z_2^3$; $S_2 = Y_2Z_1^3$; $H = U_2 - U_1$; $r = S_2 - S_1$;

- Formules de doublementsur les courbes en coordonnées jacobiennes (R = 2P)

$X_3 = T$; $Y_3 = -8Y_1^4 + M(S - T)$; $Z_3 = 2Y_1Z_1$;

où $S = 4X_1Y_1^2$; $M = 3X_1^2 + aZ_1^4$; $T = -2S + M^2$.

Les temps de calcul sont t (J + J) = 12M + 4S et t (2J) = 4M + 6S, où J signifie les coordonnées Jacobiennes.

Nous voyons que les coordonnées jacobiennes offrent un doublement plus rapide et une addition plus lente que les coordonnées projectives. Afin de faire un ajout plus rapidement, nous devrions représenter intérieurement un point Jacobienne comme le quintuple (X ; Y ; Z ; Z2 ; Z3)[40]Ceci s'appelle la coordonnée Chudnovski Jacob et est noté J Les formules d'addition dans les coordonnées jacobiennes de Chudnovski sont les suivantes.

Soit P = ($X_1$; $Y_1$; $Z_1$; $Z_1^2$; $Z_1^3$), Q = ($X_2$; $Y_2$; $Z_2$; $Z_2^2$;$Z_2^3$) et P + Q = R = ($X_3$; $Y_3$; $Z_3$; $Z_3^2$; $Z_3^3$).

- Formules d'addition de courbes en coordonnées jacobiennes de Chudnovski (P $\neq \pm$ Q)

  $X_3 = -H^3 - 2U_1H^2 + r^2$; $Y_3 = -S_1H^3 + r(U_1H^2 - X_3)$ ; $Z3 = Z_1Z_2H$ ; $Z_3^2 = Z_3^2$; $Z_3^3 = Z_3^3$;

  Où $U_1 = X_1 (Z_2^2)$ ; $U_2$ = X2 ($Z_1^2$) ; S1 = Y1 ($Z_2^3$) ; S2 = Y2 ($Z_1^3$) ; H = $U_2$ - $U_1$ ; r = $S_2$- $S_1$;

- Formule de doublement de courbes en coordonnées jacobiennes de Chudnovski (R = 2P)

  $X_3$= T ; $Y_3 = -8Y_1^4 + M(S - T)$ ; $Z_3 = 2\,Y_1Z_1$; $Z_3^2 = Z_3^2$; $Z_3^3 = Z_3^3$;

  Où S = 4 $X_1Y_1^2$; $M = 3X_1^2 + a\,(Z_1^2)^2$; T = -2S +$M^2$.

Les temps de calcul sont t ($J^c$+$J^c$) = 11 M + 3S et t ($2J^c$) = 5 M + 6S.

**3.4 Les coordonnées Jacobiennes modifies**

Ici, nous modifions les coordonnées jacobiennes pour obtenir les doublements les plus rapides possibles.Pour cela, nous représentons en interne les coordonnées jacobiennes en tant que quadruple (X ; Y ; Z ; aZ4). Nous appelons cela le système de coordonnées Jacobienne modifié,et le désignons par$J^m$. Les formules d'addition dans les coordonnées jacobiennes modifiées sont les suivantes .Soit P = ($X_1$; $Y_1$; $Z_1$;a $Z_1^4$) , Q = ($X_2$; $Y_2$; $Z_2$;a $Z_2^4$) etP + Q = R = ($X_3$; $Y_3$; $Z_3$;a $Z_3^4$)

- Formules d'addition de courbes en coordonnées jacobiennes modifiées (P $\neq \pm$ Q)

  $X_3$; = $-H^3 - 2U_1H^2 + r^2$; $Y_3 = -S_1H^3 + r(U_1H^2 - X_3)$; $Z_3 = Z_1Z_2H$; a $Z_3^4$ = a $Z_3^4$;

  Où$U_1 = X_1Z_2^2$; $U_2 = X_2Z_1^2$; $S_1 = Y_1Z_2^3$; $S_2 = Y_2Z_1^3$; H = $U_2$-$U_1$; r = $S_2$-$S_1$;

- Formules de doublement de courbes en coordonnées jacobiennes modifiées (R = 2P)
  $X_3 = T$ ; $Y_3 = M(S - T) - U$ ; $Z_3 = 2\,Y_1Z_1$; a $Z_3^4 = 2U(aZ_1^4)$ ;
  Où $S = 4\,X_1Y_1^2$; $U = 8\,Y_1^4$; $M = 3X_1^2 + (aZ_1^4)$ ; T = -2S + M2. [77].

## 3 Analyse et discussion des systèmes de coordonnées

Les temps de calcul sont alors t ($J^m$+$J^m$) = 13 M 6S et t ($2J^m$) = 4 M 4S. Il est évident qu'un doublement de coordonnées jacobiennes modifié est plus rapide qu'un doublement projectif, Jacobienne ou Jacobienne Chudnovski. De plus, il est plus rapide qu'un doublement de coordonnées affines sauf si I <3. 6M (S est défini sur 0. 8M), ce qui semble extrêmement improbable si p est supérieur à 100 bits, indépendamment du champ de définition $IF_p$ et de la mise en œuvre de l'inversion.

| système de cordonner | Additions | Doublement |
|---|---|---|
| $\mathcal{A}$ | I + 2M + S | I + 2M + 2S |
| $\mathcal{P}$ | 12M + 2S | 7M + 5S |
| $\mathcal{J}$ | 12M + 4S | 4M + 6S |
| $\mathcal{J}^c$ | 11 M + 3S | 5 M + 6S |
| $\mathcal{J}^m$ | 13 M 6S | 4 M 4S |

Tableau III. 2:Coût des opérations de systèmes de coordonner.

Après l'étude et l'analyse présentées précédemment et résumées dans le tableau III. 1, nous considérons que les systèmes jacobiennes modifiées sont les meilleurs en termes de coût total des opérations(doublement et addition).

## 4 Une nouvelle stratégie pour la courbe elliptique

Dans cette section, nous étudions une nouvelle stratégie d'exponentiation de la courbe elliptique. Jusqu'à présent, puisqu'un seul type de système de coordonnées était utilisé, il était nécessaire qu'il offre à la fois une addition et un doublement à une vitesse raisonnable (pas le plus rapide mais pas trop lent) ([42, 27,26, 39]). Le système de coordonnées jacobiennes

modifiées en est un bon exemple : il réduit le temps de calcul d'une addition en augmentant légèrement le temps de doublement, mais cela reste utile car les coordonnées jacobiennes ont un temps de doublement plus rapide mais plus lent que les coordonnées projectives.

Au contraire, nous améliorons encore le système de coordonnées Jacobiennes afin d'offrir des doublements encore plus rapides, et il n'y aura pas de perte d'exponentiation de la courbe elliptique puisque nous allons utiliser une nouvelle stratégie de systèmes de coordonnées mixtes.

**Coordonnées mixtes**

Il est évidemment possible de mélanger différentes coordonnées, c'est-à-dire d'ajouter deux points où l'un est donné dans un système de coordonnées et l'autre point dans un autre système de coordonnées. Nous pouvons également choisir le système de coordonnées du résultat. Puisque nous avons cinq types différents de systèmes de coordonnées (représentés par les symboles A, P, J, $J^c$ et$J^m$),cela donne un grand nombre de possibilités. En généralisant légèrement la notation utilisée ci-dessus, notons t (C1 + C2 = C3) le temps pouraddition de points dans les coordonnées C1 et C2 donnant un résultat dans les coordonnées C3 et par t (2C1 = C2)le temps nécessaire pour doubler un point dans les coordonnées C1 donnant un résultat dans les coordonnées C2. Le tableau 1 donne les temps de calcul pour les additions et les doublements dans différentes coordonnées (toutes les combinaisons possibles ne sont pas données, seules les plus utiles).

Le tableau 3 donne le coût de l'addition et du doublement de points pour divers systèmes de coordonnées mixtes. Dans le tableau 3, on utilise la notation C1+C2 = C3. Les points à ajouter sont indiqués par C1, C2 et leur somme est C3. Par exemple

$j^m$+ A = $j$ signifie, les deux entrées sont en coordonnée affine et la sortie en coordonnée Jacobienne modifiée. Nous pouvons remarquer dans le tableau 3 que la coordonnée Jacobienne produit le doublement de points le plus rapide, tandis que les coordonnées mélangées produisent l'ajout de points le plus rapide.

| *Point Doubling* | | *Point Addition* | |
|---|---|---|---|
| *Operation* | *Cost* | *Operation* | *Cost* |
| $2P$ | 7M + 5S | $J^m + J^m$ | 13M+6S |
| $2J^c$ | 5M + 6S | $J^m + J^c = J^m$ | 12M+5S |
| $2J$ | 4M + 6S | $J + J^c = J^m$ | 12M+5S |
| $2J^m = J^c$ | 4M + 5S | $J + J$ | 12M+4S |
| $2J^m$ | 4M + 4S | $P + P$ | 12M+2S |
| $2A = J^c$ | 3M + 5S | $J^c + J^c = J^m$ | 11M+4S |
| $2J^m = J$ | 3M + 4S | $J^c + J^c$ | 11M+3S |
| $2A = J^m$ | 3M + 4S | $J^c + J = J$ | 11M+3S |
| $2A = J$ | 2M + 4S | $J^c + J^c = J$ | 10M+2S |
| - | - | $J + A = J^m$ | 9M+5S |
| - | - | $J^m + A = J^m$ | 9M+5S |
| - | - | $J^c + A = J^m$ | 8M+4S |
| - | - | $J^c + A = J^c$ | 8M+3S |
| - | - | $J + A = J$ | 8M+3S |
| - | - | $J^m + A = J$ | 8M+3S |
| - | - | $A + A = J^m$ | 5M+4S |
| - | - | $A + A = J^c$ | 5M+3S |
| $2A$ | 1 + 2M + 2S | $A + A$ | I+2M+S |

Tableau III. 3:Coût des doublements et additions mixtes sur Fp [43].

Pour utiliser des coordonnées mixtes, il est parfois nécessaire de convertir une représentation de points d'un système de coordonnées en un autre système afin que l'entrée ait le format requis pour l'algorithme d'addition de points et de doublement de points. Un point dans une coordonnée affine (x, y) peut être converti en toute autre coordonnée à l'aide de l'algorithme 3.8. Aucune des conversions ne nécessite une quadrature, une multiplication ou une inversion modulaire les rendant assez rapides.

| Coordonnées affines (x, y) ⟶ | autrecoordonnée |
|---|---|
| Affine (x, y) | Projective (x, y, 1) |
| Affine (x, y) | jacobiennes (x, y, 1) |
| Affine (x, y) | jacobiennes Chudnovski (x, y, 1, 1, 1) |
| Affine (x, y) | jacobiennes modifiées(x, y, 1, a, 1) |

Tableau III. 4:Algorithme3.8 Conversion des coordonnées Affines (x, y) en d'autres Coordonnées [44].

## 5 Comparaison de performance de système de coordonner de la courbe elliptique

Dans cette section, nous appliquons la méthode glissante (sliding) avec les systèmes de cordonner décrites dans la section précédente en les appliquant sur un processeur Pentium 4 à 2,7 GHz. Avec une sélection de champ de 256 bit, le calcul de routine du domaine est écrit en Java avec la sélection des coordonnées de projection. Pour déterminer le temps nécessaire pour effectuer chaque système de cordonner, Pour ce faire, nous suivons les étapes selon l'organigramme illustré dans la figure ci-dessous.

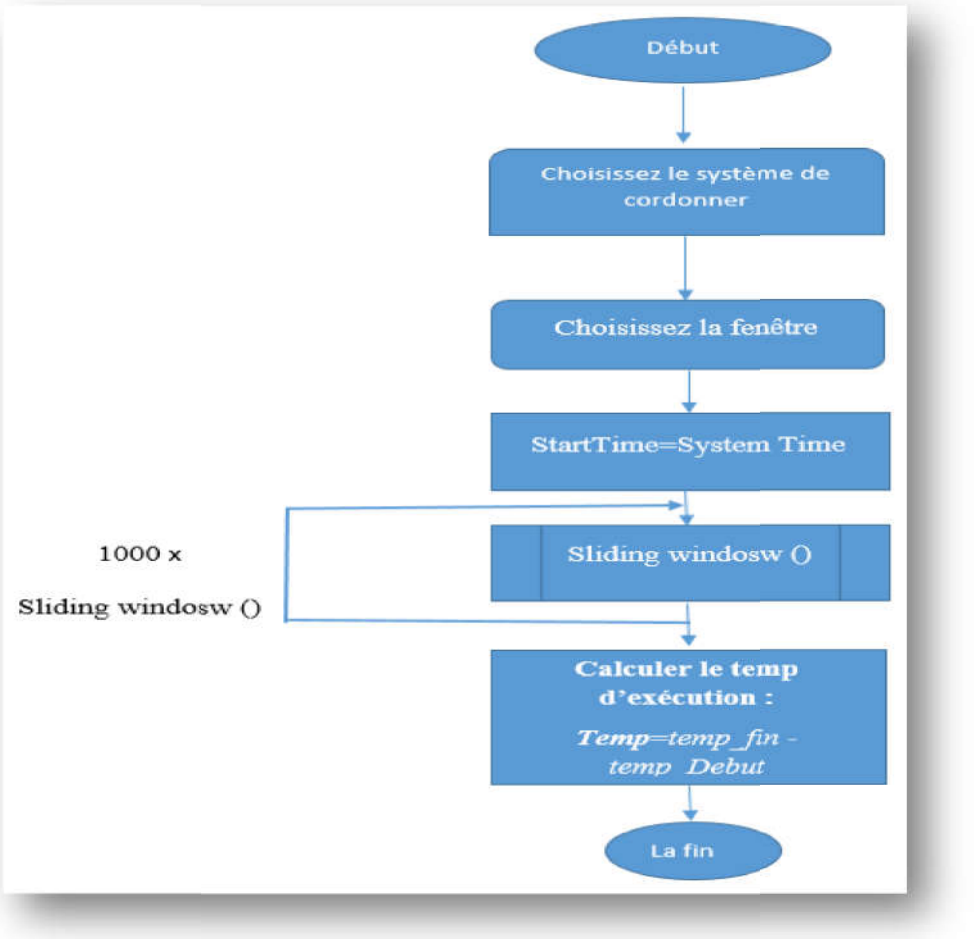



Figure III. 11**:** la procédure pour calculer le temps d'exécution de chaque système de cordonner.

## 6 Résultat d'exécution de la multiplication scalaire dans les systèmes de coordonner

Grâce à l'expérience décrite ci-dessus, nous avons obtenu les résultats présentés dans le tableau 1d'exécution des systèmes de cordonner de la courbe elliptique sur les champs premiers en général. Dans la conception du système cryptographique que nous allons utiliser en courbe elliptique cryptographie, nous devons choisir système de cordonner la mieux adaptée à notre système.

| systèmes de coordonner | multiplication scalaire | Résultats de temps (ms) |
|---|---|---|
| **affine** | Sliding | 1,81 |
| **projectives** | Sliding | 1.65 |
| J_Chudnovski | Sliding | 1.6 |
| J_modifiées | Sliding | 1.2 |
| coordonnées mixtes | Sliding | 0 .7 |

Tableau III. 5: Temps d'exécution des systèmes de cordonner de la courbe elliptique

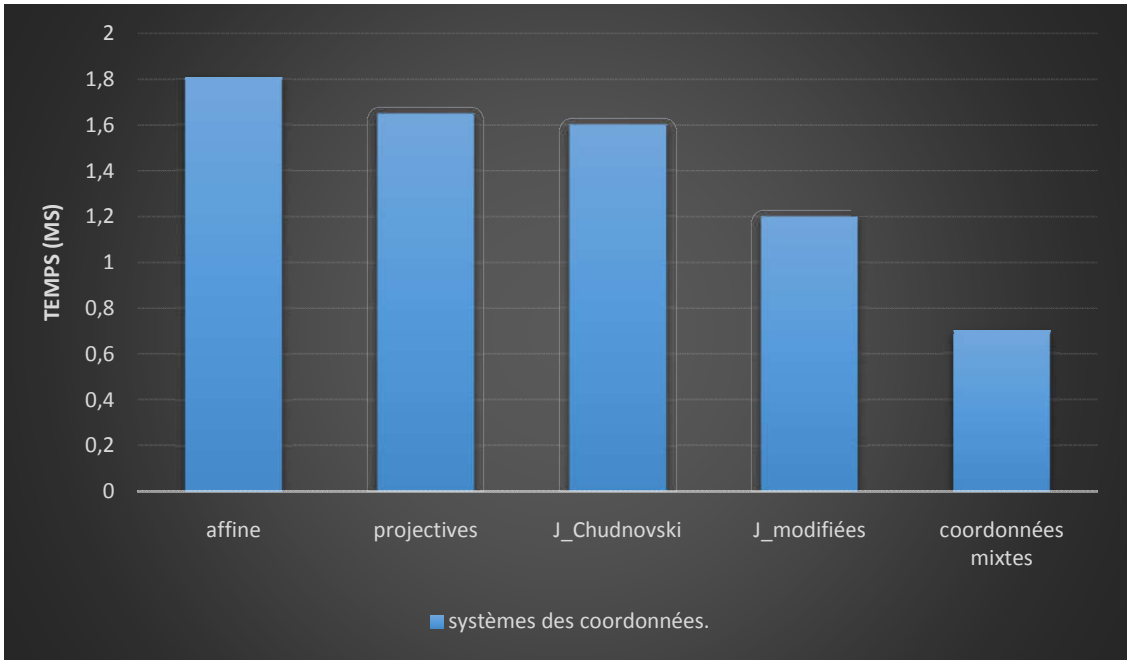



Figure III. 12 **:** présentation graphique de temps (ms)pour les différentes systèmes des coordonnées.

Enfin, après l'étude et l'analyse fournies à l'avance et résumées dans le tableau III. 5, nous considérons que les systèmes de coordonnées mixtes sont les meilleurs en termes de coût total des opérations (doublement et addition). Est-ce une solution à ce problème. Par conséquent, nous utiliserons ce système pour implémentation opération de la multiplication scalaire " sliding ".

En résumé cela signifie que, durant notre algorithme de multiplication de point, le doublement précédent se fera à l'aide de l'opération $2\,j \rightarrow j$ et les additions se feront en utilisant $j^m + A \rightarrow j^m$.

## 7 Conclusion

Nous avons exploré les différentes méthodes de la multiplication scalaire ainsi les différentes systèmes de coordonnés afin de choisir un meilleur algorithme et un meilleur système de coordonner.

D'après l'étude, analyse et l'expérience que n'a faite, on a constaté que l'algorithme à fenêtre glissante donne un temps d'exécution optimale par rapport aux autres algorithmes.

Sur un niveau plus pas on a constaté que l'utilisation des coordonnés mixtes donne une valeur positif à la performance de la multiplication scalaire.

Au chapitre suivant voir l'implémentation de cryptosystème ECC avec l'algorithme de la fenêtre glissante et les coordonnées mixtes.

**Chapitre *IV :***

# CONCEPTION ET IMPLÉMENTATION DU COURBE ELLEPTIQUE

# 1Introduction

La difficulté d'implémenter la cryptographie à base des courbes elliptiques a causé un manque dans ce domaine. Avec le mouvement vers l'IoT 1, il est devenu impérativement nécessaire d'aller vers la cryptographie à base des courbes elliptique à cause de sa rapidité remarquable.

Nous allons montrer dans ce chapitre les étapes et moyens qu'on a utilisés pour implémenter notre application qui peut utiliser des clés de 256 bits.

Nous allons en premier temps expliquer la conception de notre application, puis détailler le code source et les fonctions principales de notre application.

# 2 Étude de l'algorithme de fenêtre glissante (Sliding window).

Nous allons faire certain étude sur les fenêtres pour choisir la meilleure fenêtre de l'algorithme sliding.

Calculons [763] P (autrement dit k = 763) avec l'algorithme de fenêtre glissante avec K recodé sous forme binaire avec différentes tailles de fenêtre allant de 2 à 10. Il est observé qu'à mesure que la taille de la fenêtre augmente, le nombre de calculs préalables Augmente géométriquement. Dans le même temps, le nombre d'additions et de doublement des opérations diminue.

Maintenant, nous présentons les détails de la taille de la fenêtre pour trouver la taille de fenêtre optimale via cet exemple [45] :

| | |
|---|---|
| ▪ **Taille de la fenêtre w = 2**<br>▪ **Nb de précalculs = $2^w$ - 1 = $2^2$ - 1 = [3] P**<br>▪ **763 = 1011111011**<br>▪ **Les valeurs intermédiaires de Q sont P, 2P 4P, 8P, 11P, 22P, 44P, 47P, 94P, 95P, 190P, 380P, 760P, 763P**<br>▪ **Coût de calcul = 9Doublement,** | ▪ **Taille de la fenêtre w** = 3<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^3$ - 1 = [7] P**<br>▪ **Les valeurs intermédiaires de Q sont 5P, 10P, 20P, 40P, 47P, 94P, 188P, 376P, 381P, 762P, 763P** |

| | |
|---|---|
| **4additions et 1 précalculs.** | ▪ **Coût informatique = 7 doublons, 3 additions et 3précalculs.** |
| ▪ **Taille de la fenêtre w** = 4<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^4$ - 1 = [15] *P***<br>▪ **Les valeurs intermédiaires de Q sont 11P, 22P, 44P, 88P, 95P, 190P, 380P, 760P, 763P**<br>▪ **Coût informatique** =6 **doublons,** 2 **additions et** 7**précalculs.** | ▪ **Taille de la fenêtre w** = 5<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^5$ - 1 = [31] *P***<br>▪ **Les valeurs intermédiaires de Q sont 23P, 46P, 92P, 184P, 368P, 736P, 763P**<br>▪ **Coût de calcul** = 5 **doublons,** 1 **additionset 15 précalculs** |
| ▪ **Taille de la fenêtre w** = 6<br>▪ **Nb de précalculs $2^w$ - 1 = $2^6$ - 1 = [63]P 2P,3P, 5P, 7P, 9P,11P,13P,15P,17P,19P, 21P, 23P, 25P, 27P,29P, 31P, 33P, 35P, 37P, 39P, 41P, 43P, 45P, 47P, 49P, 51P, 53P, 55P, 57P, 59P, 61P**<br>▪ **Les valeurs intermédiaires de Q sont 47P, 94P, 188P, 376P, 752P, 763P**<br>▪ **Coût de calcul** = 4 **doublons,** 1 **addition** | ▪ **Taille de la fenêtre w = 7**<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^7$ - 1 = [127]P 2P,3P, 5P, 7P, 9P,11P,13P,15P,17P, 19P, 21P, 23P, 25P, 27P,29P, 31P, 33P, 35P, 37P, 39P, 41P, 43P, 45P, 47P, 49P, 51P, 53P, 55P, 57P, 59P, 61P, 63P, 65P, 67P, 69P, 71P, 73P, 75P,77P, 79P, 81P, 83P, 85P, 87P, 89P, 91P, 93P, 95P, 97P, 99P, 101P, 103P, 105P, 107P, 109P, 111P, 113P, 115P, 117P, 119P,** |

| | |
|---|---|
| **et 31 précalculs** | **121P**<br>▪ **763 = 1011111 011**<br>▪ **Les valeurs intermédiaires de Q sont 95P, 190P, 380P, 760P, 763P**<br>▪ **Coût de calcul =** 3 **doublons,** 1 **addition et** 61 **précalculs** |
| ▪ **Taille de la fenêtre w =** 8<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^8$ - 1 = [255] P**<br>▪ **La valeur intermédiaire $Q$ est 95P, 190P, 380P, 760P, 763P**<br>▪ **Coût de calcul =** 3 **doublons,** 1 **addition et** 127 **précalculs** | ▪ **Taille de la fenêtre w = 9**<br>▪ **Nb de précalculs $2^w$ - 1 = $2^9$ - 1 = [511] P**<br>▪ **763 = <u>1011111011</u>**<br>▪ **La valeurs intermédiaires Q sont 381P, 762P, 763P**<br>▪ **Coût de calcul =** 1 **doublon,** 1 **addition et** 251 **précalculs** |
| ▪ **Taille de la fenêtre w = 10**<br>▪ **Nb de précalculs =$2^w$ - 1 = $2^{10}$ - 1= [1023] P**<br>▪ **763 = <u>1011111011</u>**<br>▪ **La valeurs intermédiaires Q sont 763P**<br>▪ **Coût de calcul =** 0 **doublons,** 0 **addition set 510 précalculs** | |

Le compromis entre le coût de calcul et le la taille de la fenêtre est indiqué dans le tableau 1.

| La taille de la fenêtre | Nombre Doublement | Nombre Additions | Nombre de précalculs |
|---|---|---|---|
| 2 | 9 | 4 | 1 |
| 3 | 7 | 3 | 3 |
| 4 | 6 | 2 | 7 |
| 5 | 5 | 1 | 15 |
| 6 | 4 | 1 | 31 |
| 7 | 3 | 1 | 61 |
| 8 | 3 | 1 | 127 |
| 9 | 1 | 1 | 151 |
| 10 | 0 | 0 | 501 |

Tableau IV. 1:le coût des opérations dudoublement, additions et de précalculs par rapport la taille de fenêtre.

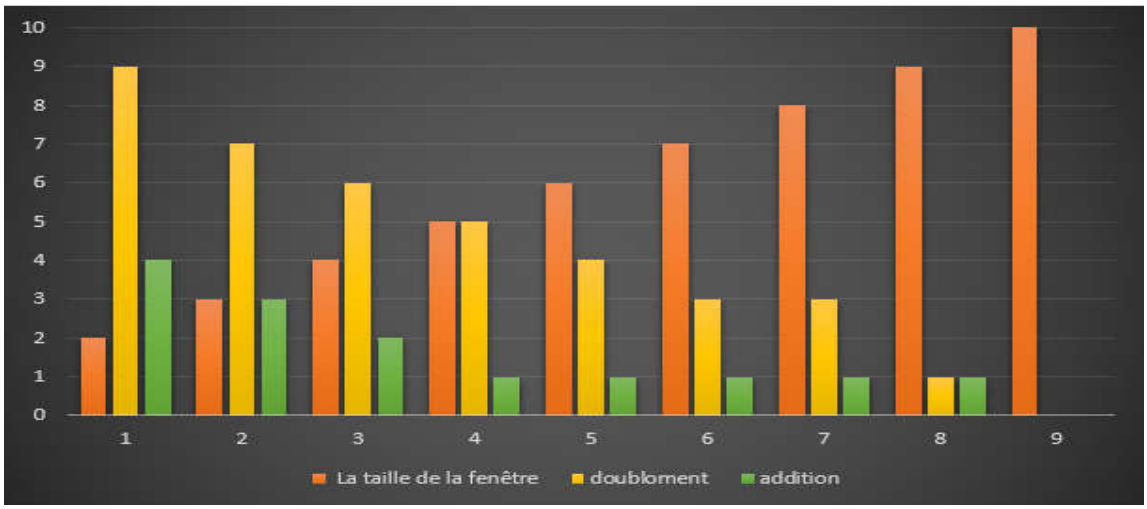



Figure IV. 1représentation de nombre d'addition et doublement par rapport la taille de fenêtre

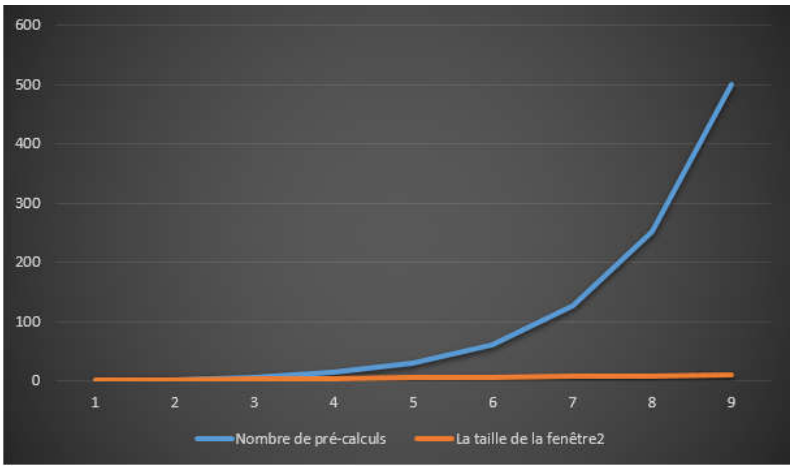



Figure IV. 2: le nombre de precalculés par rapport la taille de fenêtre.

Afin d'observer l'effet de largeur de fenêtre sur la méthode de multiplication de courbe elliptique sur les champs premiers, nous implémentons notre algorithme avec un système de coordonnées affine et mixte. La fenêtre est remplacée à chaque fois. Nous avons choisi le champ premier $F_{256}$ pour l'observation de l'effet de largeur de fenêtre. Dans le tableau 3.6, les résultats de chronométrage sont indiqués avec l'indication du système de coordonnées mis en œuvre. La largeur de fenêtre appropriée peut différer pour différentes méthodes de multiplication de points. Nous pouvons avoir un avis sur l'effet de largeur de fenêtre Selon les valeurs de synchronisation, nous pouvons choisir la largeur de fenêtre appropriée pour la méthode de multiplication de points EC qui dominera notre cryptosystème. Pour déterminer la meilleure fenêtre de multiplication scalaire « sliding windosw » Pour ce faire, nous suivons les étapes selon l'organigramme illustré dans la figure ci-dessous.

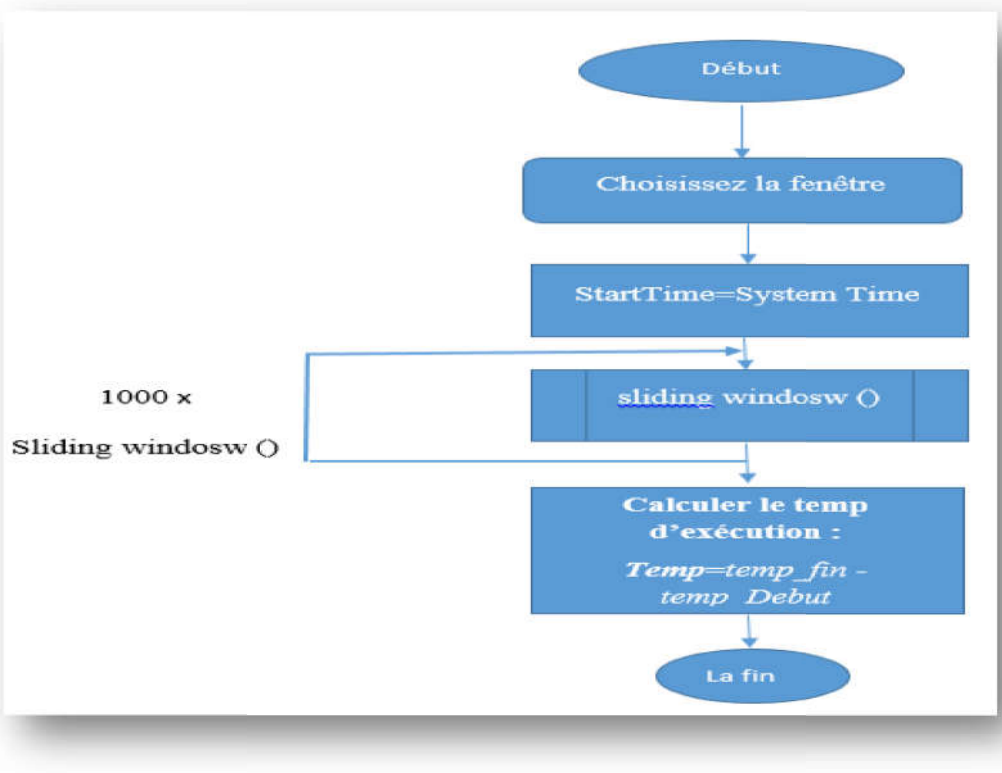



Figure IV. 3:la procédure pour calculer le temps d'exécution de chaque fenêtre.

| Fenêtres / Méthode | Fenêtre =3 | Fenêtre =4 | Fenêtre =5 | Fenêtre =6 |
|---|---|---|---|---|
| Sliding window | 1.32 | .1.87 | 1.85 | 1.89 |

Tableau IV.2 : résultats de cadencement (ms) des méthodes de fenêtre sur $F_{256}$ pour différentes largeurs de fenêtre.

D'après l'étude précédente présentée au tableau III.1 : au tableau III. 2 : et à la figure 2, le nombre de précalculs est associé à la taille de la fenêtre, comme indiqué dans le tableau 2. Il convient de noter que la meilleure fenêtre en termes de nombre de précalculs, de multiples et d'ajouts et de durée minimale de mise en œuvre de la méthode de multiplicationscalaire , Ce qui nous donne la meilleure et la meilleure solution, ainsi que le coût total inférieur par rapport aux autres fenêtres sont la fenêtre numéro 3.

## 3 Conception

L'étude de conception est la phase la plus importante du cycle de développement d'un système informatique. En effet, elle permet de confronter la spécification et l'analyse avec l'implémentation. Elle présente le point de convergence des deux aspects : <Le quoi faire >(analyse) et le <comment faire >(réalisation). Ceci justifie, parfois, le retour vers une étape antérieure afin de rectifier un besoin pour l'implémenter, ou affiner l'analyse. Pour cette raison, on consacre cette partie à la conception de notre application.

Nous voulons développer une application de bureau qui nous permet d'effectuer un cryptage basé sur des courbes elliptiques. L'utilité de cette application est une présentation du principe des courbes elliptiques. Pour cela, l'utilisateur aura une fenêtre avec deux parties, une partie de cryptage et une partie de décryptage

Dans cette étape on a utilisé le langage de modélisation UML, où notre conception est faite seulement avec le digramme cas utilisation et le diagramme d'activité. Au premier lieu, le diagramme cas utilisation décrivent les principales fonctions de notre application comme montre la figure ci-dessus

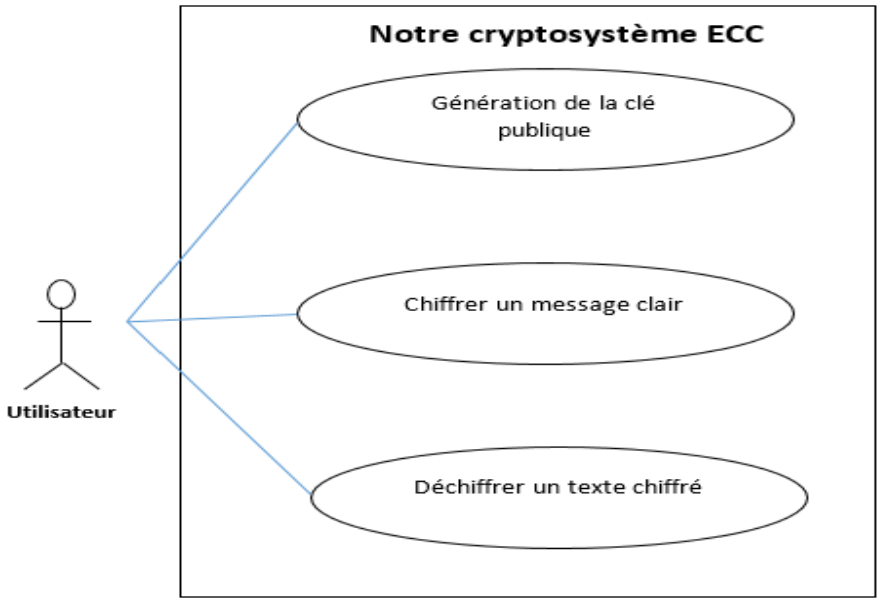



Figure IV. 4: 1Diagramme cas utilisation de notre système

La figure suivante montre le diagramme d'activité des différentes tâches effectuées par notre application :

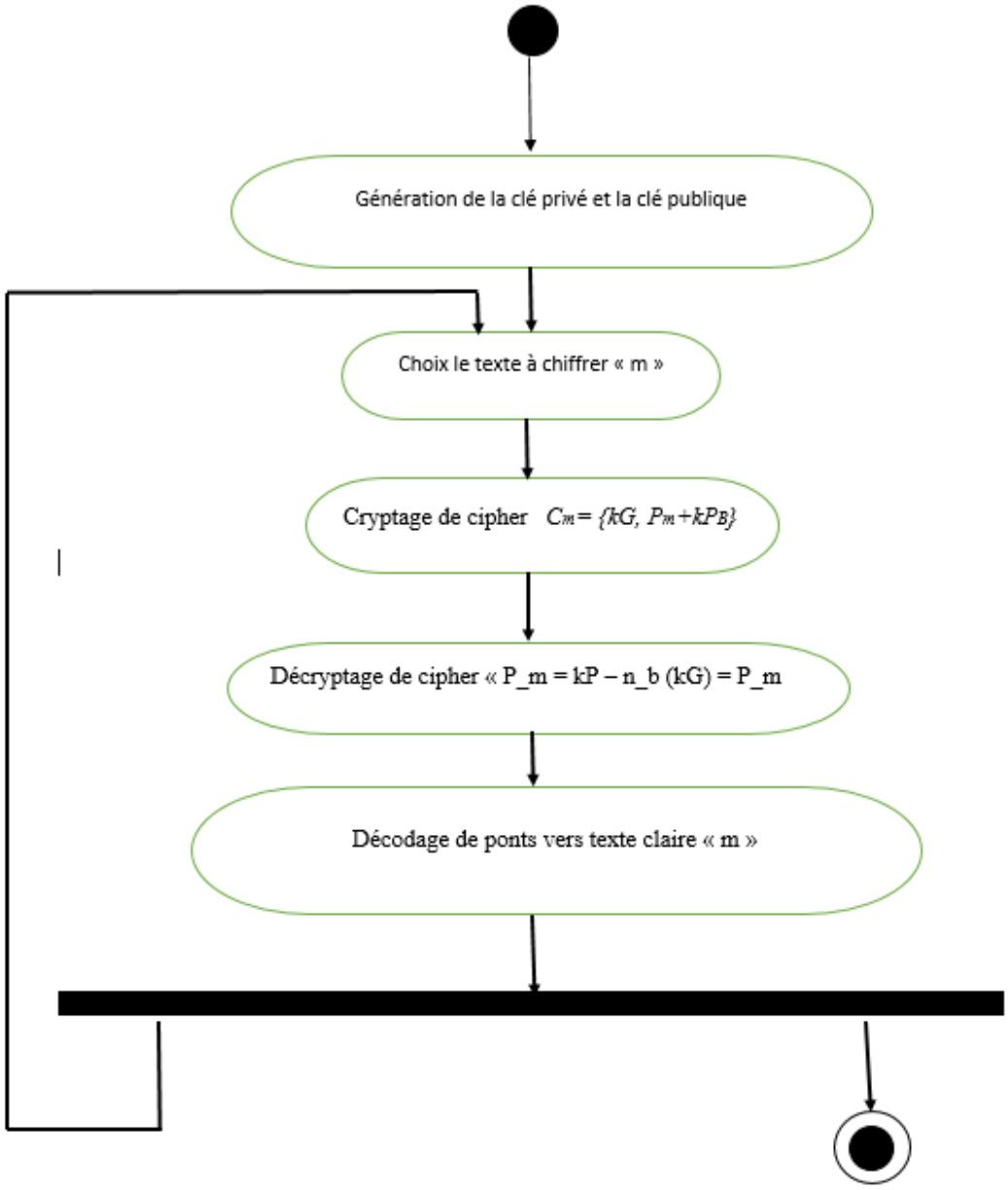



Figure IV. 5:Diagramme d'activité du fonctionnement global de l'application.

# 4 Implémentation

Dans cette partie, on va aborder le coté implémentation de l'application, où on parlera des différentes classes et ressources utilisées. On va donner un soin particulier pour détailler quelques morceaux de code afin de mieux expliquer le fonctionnement de l'application. On discutera aussi les différents problèmes rencontrés durant le développement de l'application ainsi que les moyens utilisés pour les résoudre.

## 4.1 Outils et Environnement

- Pc HP G420, processeur Intel core TM i5, RAM 4Go.
- Système d'exploitation Windows 8.1.
- NetBeans IDE 8.0.2.
- Langage de programation : java.

## 4.2 Intérface principale

L'application commence par une courbe NIST p-256.

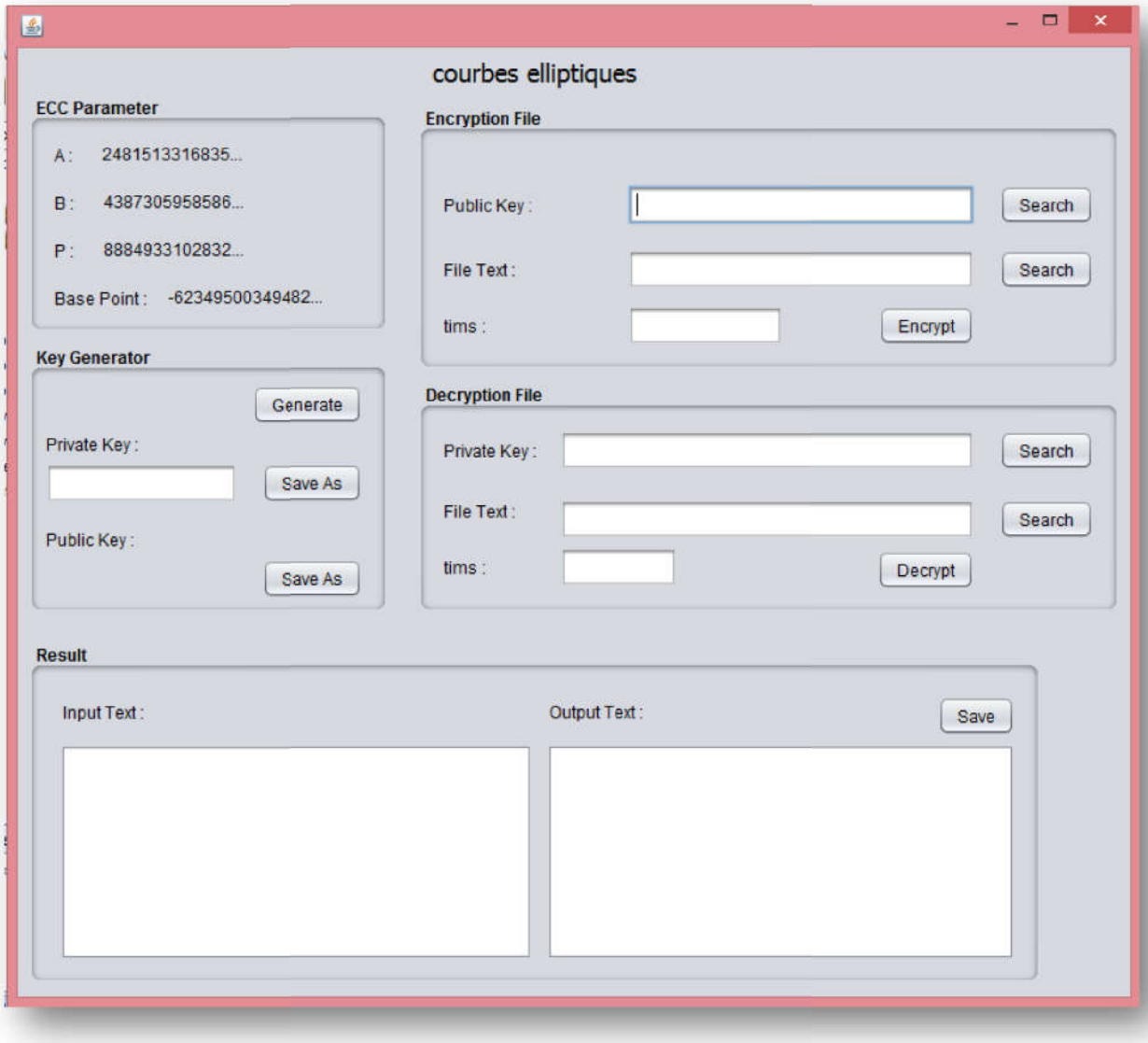



Figure IV. 6:Interface principale de l'application.

Chaque fois que l'application est lancée, l'utilisateur sélectionne la clé privée, puis clique sur le bouton "Generate" et obtient la clé publique.




Figure IV. 7:Générer une clé publique

Pour chiffrer un message, l'utilisateur entre un fichier dans le champ "*file text*", entre la clé publique précédemment stockée dans un fichier dans le champ "public key", puis clique sur le bouton "*Encrypt*". Il comportera une série de lettres, de chiffres et de symboles représentant le texte chiffré à clé publique..

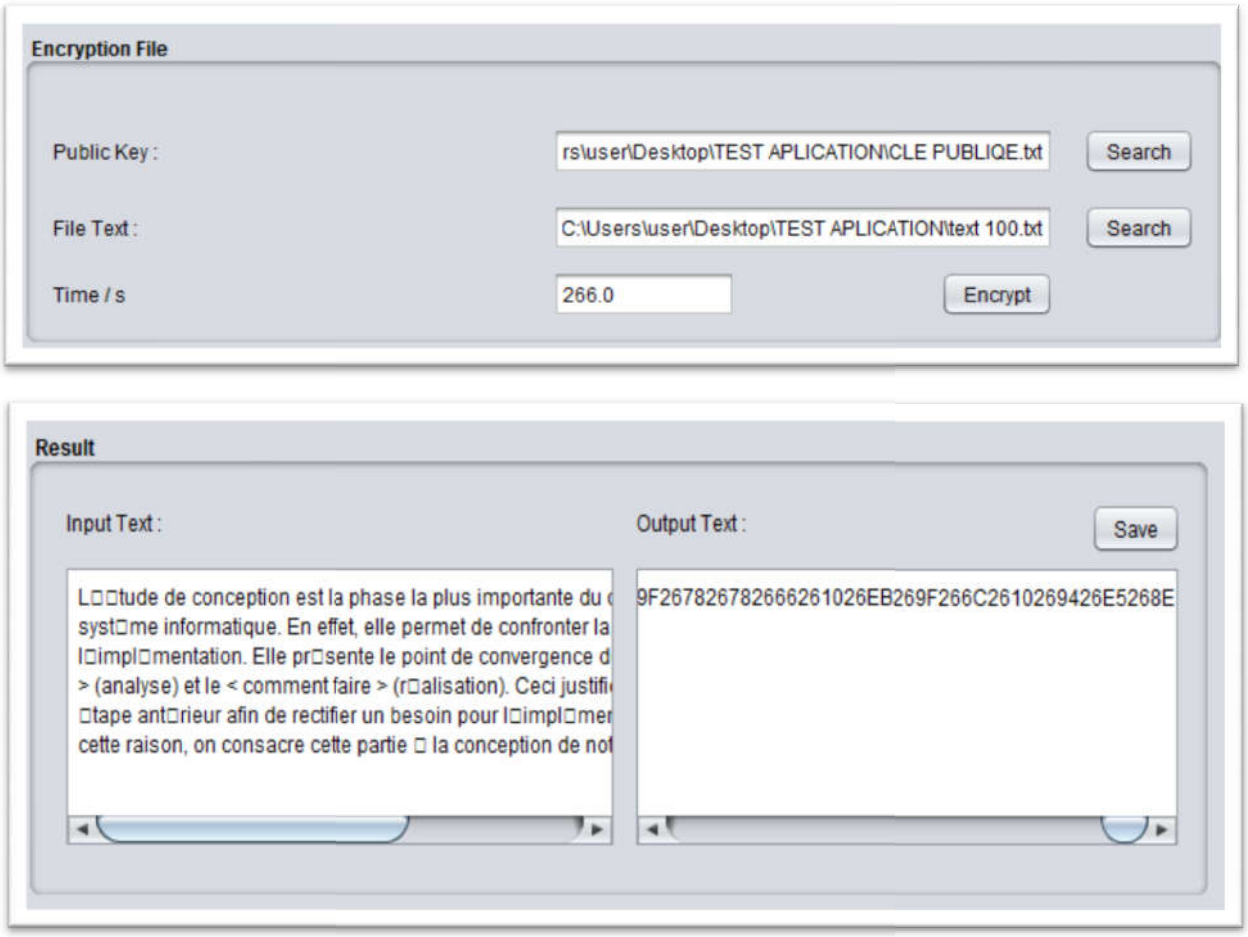



Figure IV. 8:Cryptage du message

Pour déchiffrer un message, l'utilisateur entre un fichier dans le champ "*file text*", entre la clé privée précédemment stockée dans un fichier, l'insère dans le champ "*private key*", puis clique sur le bouton "*decrypt*". Il aura le texte original.

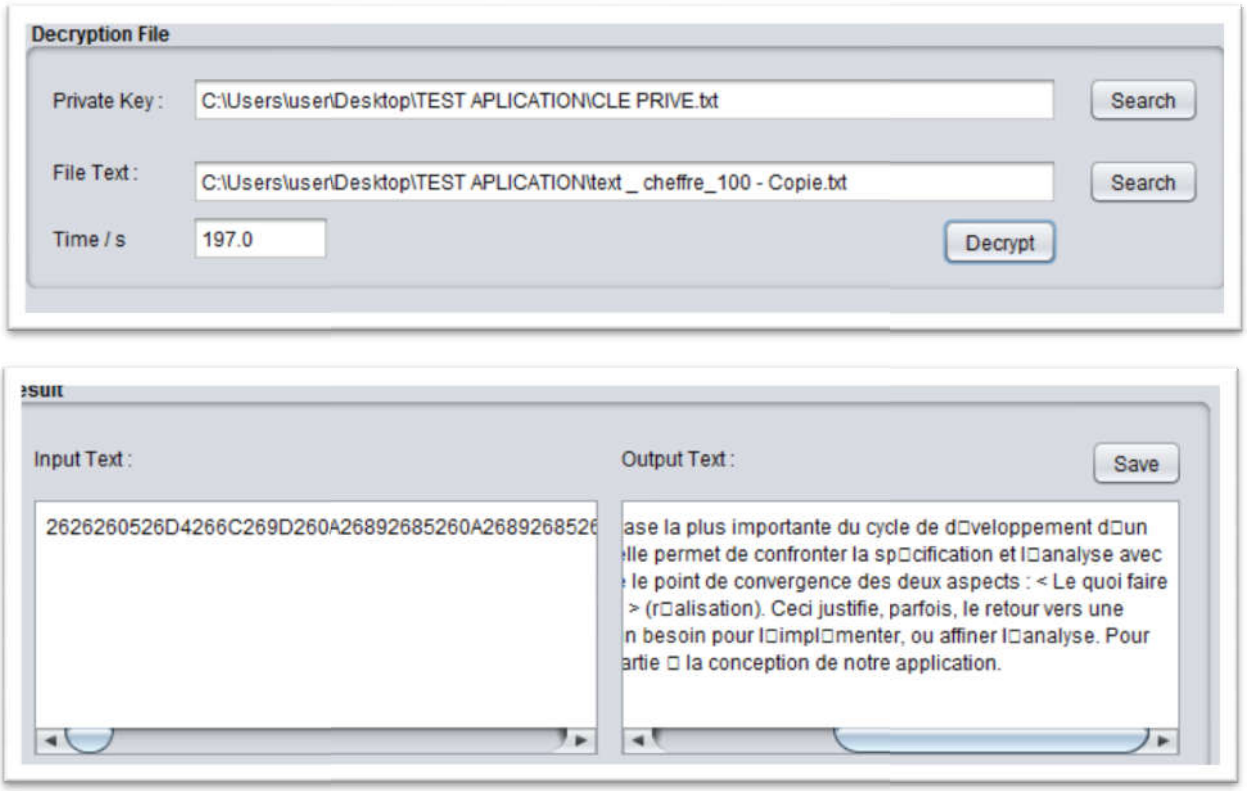



Figure IV. 9:Décryptage du message

## 4.3 Etapes de génération de la clé publique

Après avoir sélectionné la courbe et le point de base, l'utilisateur crée cette clé privée, puis la clé publique

La clé publique *PA* est la multiplication de *n* (La clé privée est un nombre aléatoire)avec le point de base *G* : $n * G$. La fonction *generatePublicKey()* nous permet de faire génération de la clé publique .

```
public Point generatePublicKey(long pri) {
    PointProccessor processor = new PointProccessor();
    Point result = processor.sliding(pri, base,3);
    return result;
}
```

## *generatePublicKey()*

En effet, la fonction *generatePublicKey()*, détaillons alors son fonctionnement :

On fait appelle à la fonction : *sliding*() nous permet de faire cette multiplication, cette fonction a été expliquée dans le chapitre précédent.

```
public Point sliding(long L,Point P,int w){
    /*
    bloc for eleptic corb
    */
    String n = Long.toBinaryString(L);
    Point result = new Point();
    Point base = new Point(P.getX(), P.getY());
    result = base;


    /*******************************/
    int l=n.length();
    int i=l-1;
     int pos;
     int st;
     int s;
     int u;


    while (i >=0) {   //11
        pos=l-(i+1);//0
        st=Character.getNumericValue(n.charAt(pos));//relevre la premier char a gauche
        if (st==0) {//suprimer les 0 agauche
            result = doublePointj(result);

           i--;//0
           //avecI++;
        }else {

            s=Min(l-(i-(w-1)), l);//5

            while (Character.getNumericValue(n.charAt(s-1))==0) { //Ns  supprmer les 0 adroit
                s--;//4
            }
            /*************************************/
            for (int j = 1; j < (s-pos+1); j++) {// parcourire la taille de fentre moin-1 fois
                result = doublePointj(result);

            }
            /*************************************/
            /*System.out.println("la valeur de s est: "+(s-pos+1));*/
            u=Integer.parseInt(n.substring(pos,s),2);//converter la valeure de fentre au binaire,(l-i) est camculé
            /*System.out.println("la debut est :"+pos+" la fin est :"+s+" ====>("+n.substring(pos,s)+") ===> "+u);*/
            /*************************************/
            result =add(result, slidding(u, P));
            /*************************************/
            i=l-s-1;  //
        }
    }
    return result;//it reurns value of R
}
```

Au lieu d'utiliser l'algorithme *sliding*() pour faire la multiplication d'un point avec n (1 doublement (n − 1) addition), puisque il y'a un grand problème pour implémenter cette algorithme sur grand nombre des points ; on a implémenté l'algorithme "NAF", qui rend la multiplication plus rapide et un peu facile.

- *Entrée : Un point P d'une courbe elliptique E et un entier k t q : 1 < k < r*
- *Sortie : Le point Q = kP*

1. *Calculer* $k = \sum_{i=0}^{l-1} k_i 2^i$
2. $Q = \infty$
3. *tant que k >0 faire*

a. $Q := 2 * Q$
b. Si $k_i = 1$ alor $Q := Q + P$
c. Si $k_i = -1$ alor $Q := Q - P$

4. *Renvoyer Q*

Code java

```
public  Point NAF (long n, Point P ){
    Point result = new Point();
    Point base = new Point(P.getX(), P.getY());
    result = base;

    String binary = Long.toBinaryString(n);
    for (int i=binary.length()-1; i>=0; i--) {
        result = doublePoint(result);
        if (binary.charAt(i) == 1) {
            result = add(result, base);

        }
        if (binary.charAt(i) == 0) {
            result = minus(result, base);
        }

        i--;
    }
    return result;
}
```

## 5 Résultat et discussion

Une expérience est analysée pour comparer les performances du code ECC standard [46] avec le code ECC amélioré en ce qui concerne les critères d'évaluation tels que le temps (en millisecondes), la taille (en Ko) et la sécurité. Un fichier avec différentes tailles telles que 100, 200, 300 et 400 Ko a été utilisé pour l'analyse graphique. L'analyse de corrélation a également été réalisée pour étudier la relation entre le texte brut et le texte chiffré. L'évaluation de l'ECC standard et de l'ECC amélioré est vérifiée sous les paramètres suivants.

- Temps de cryptage
- Temps de décryptage

### 5.1 Temps de chiffrement

La figure 8 illustre l'évaluation de la performance du temps de cryptage entre le code ECC standard et le code ECC mis en œuvre. Ce paramètre a une signification importante car il détermine le temps nécessaire à la conversion du texte brut en texte chiffré. Depuis, le code ECC implémenté génère une clé robuste, avec l'application des points clés sur l'équation, les résultats varient légèrement de la norme ECC. De plus, le caractère répété des données est remplacé par les différents chiffrements lors du chiffrement, ce qui reflèterait également la différence entre les chiffrements entre ECC standard et implémenté.

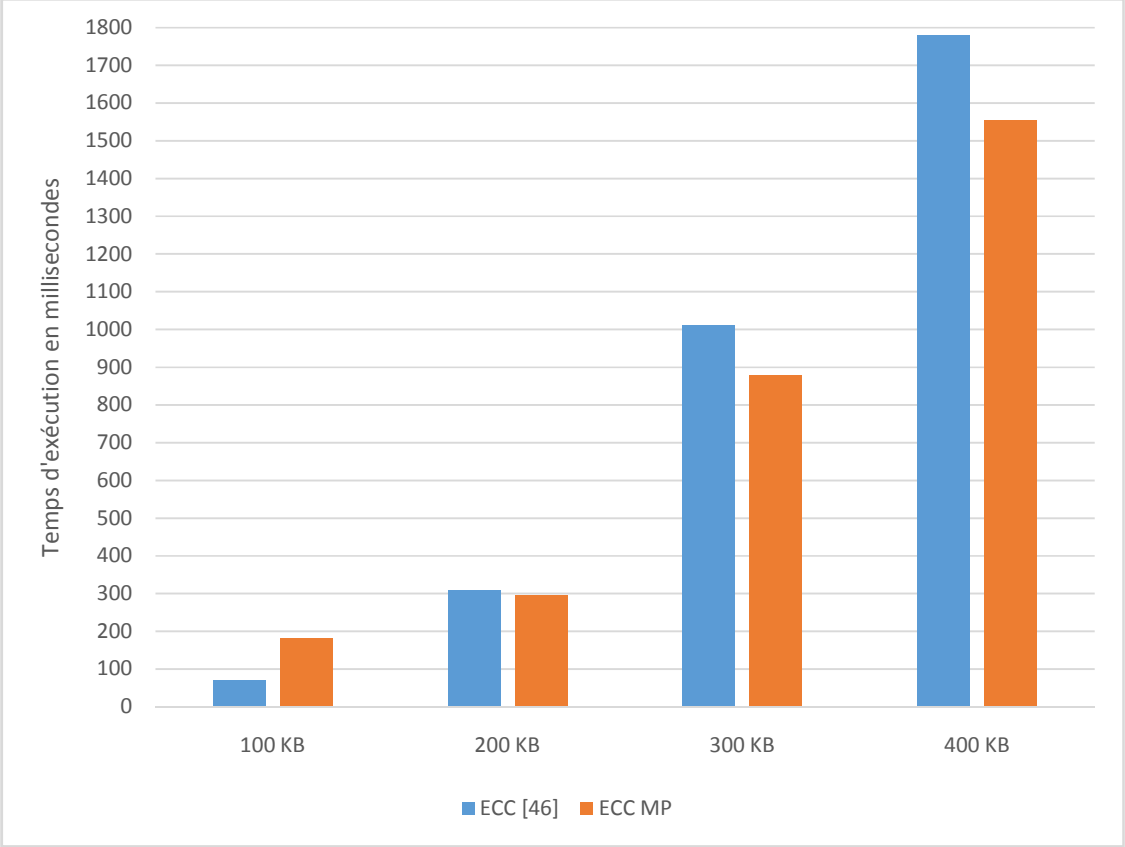



Figure IV. 10: Temps de chiffrement entre ECC standard et ECC proposé.

## 5.2 Temps de déchiffrement

La figure 11 illustre l'évaluation de la performance de temps de décodage entre les normes ECC et ECC mises en œuvre. Le temps de décodage dans les codes ECC et ECC séparés n'indique pas une différence plus importante car le processus de décodage est identique dans les

deux cas. ECC et ECC mis en œuvre. Cependant, le temps de décodage dans les codes ECC implémentés est légèrement inférieur aux codes ECC standard, car la génération de clé est légèrement différente des codes ECC standard et des codes ECC implémentés.

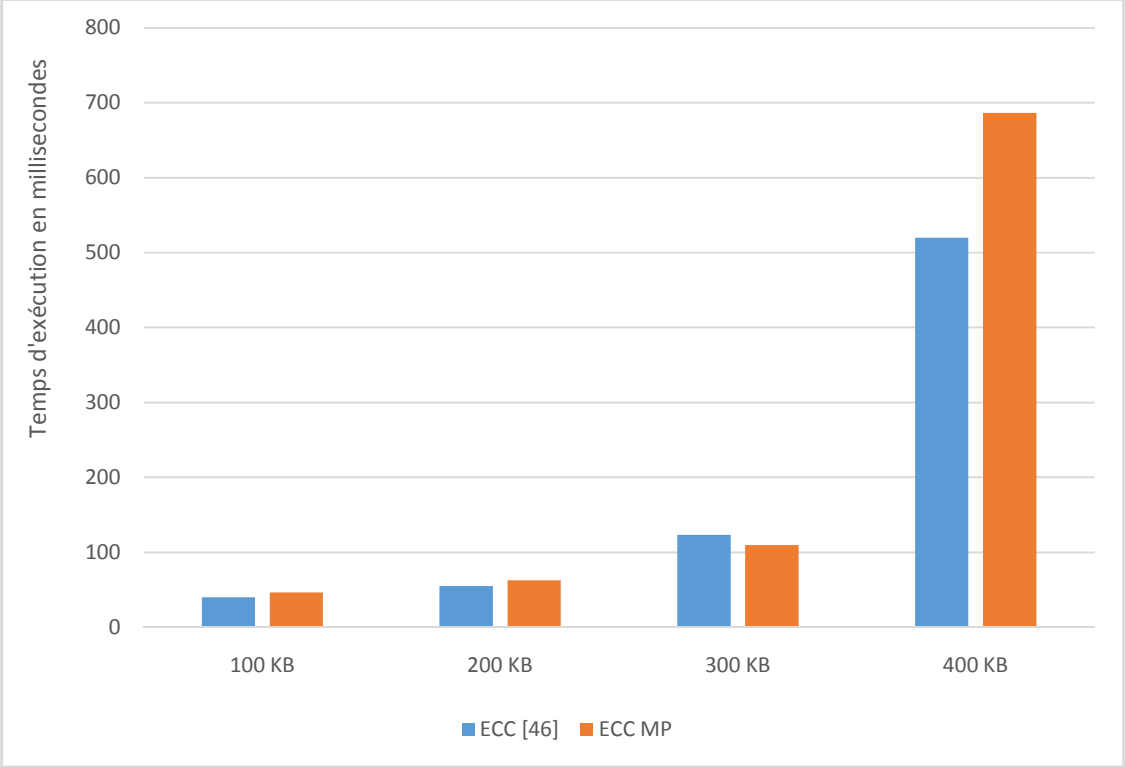



Figure IV. 11:Temps de déchiffrement entre le ECC standard et le ECC proposé.

## 6 Conclusion

Dans cette section, nous avons vu les étapes et les fonctions nécessaires à la mise en œuvre du codage par courbe elliptique (ECC). Bien que ce système de cryptage soit difficile à mettre en œuvre, la manière dont la partie théorique a été introduite nous a permis de coder plus rapidement en suivant simplement l'algorithme de base.

Nous avons également discuté et analysé l'expérience visant à comparer les performances du temps ECC standard avec le temps ECC amélioré pour des critères d'évaluation tels que le temps (ms).

# Conclusion Générale

Dans de ce projet de fin d'études, on s'est intéressé à la cryptographie à base des courbes elliptiques. Malgré que la recherche dans ce domaine est active, il a été difficile de trouver des articles ou des livres qui expliquent bien cette technique de cryptage.

Pour pallier à ce manque, on s'est intéressé à rédiger ce mémoire en commençant par une vue globale sur la cryptographie moderne et les outils utilisés pour assurer l'intégrité, l'authenticité et la confidentialité des données. Puis, on a présenté une explication plus ou moins détaillée de la théorie des courbes elliptiques.

Afin de développer cette application, nous nous intéressons à l'amélioration de la méthode de multiplication scalaire. Ce qui nous fournit des ECC plus rapides et plus efficaces que les autres ECC standards. Malgré le fonctionnement parfait de cette application, l'optimisation et l'amélioration restent toujours possibles. On peut par exemple utiliser les threads pour le chiffrement et le déchiffrement parallèles. On peut aussi ajouter de l'aléatoire dans le code pour rendre l'attaque par canaux caché plus difficile. Le codage des points chiffrés en hexadécimale pour réduire leurs tailles est aussi une amélioration possible.

# Bibliographies

[1]GHISLAINE Labour et HERVE Schauer,Introduction à la cryptographie consultants, 1999-2001, page 3.

[2] Alice Lan, Benoit Vandervelde, cours : « Panorama des algorithmes de Cryptographie », université de Nantes, 13 mars 2011.

[3] Https://support.microsoft.com/fr-fr/kb/246071 (22/06/2016**).**

[4] Cryptographie et Sécurité informatique, Université de Liège, 2009 - 2010.

[5] Nadia El Merabt : « les concepts fondamentaux de la cryptographie», mémoire de master, université de Caen, France, mars 2014.

[6] GaelThomas : « design et analyse de sécurité pour les constructions pour la cryptographie symétrique », mémoire de master, Université de limoges, juin 2015.

[7] Bayart, Frederic. La saga du DES. http://www.bibmath.net/crypto. [En ligne] [Citation : 09 03 1013]

[8] Nicolas Estibals : «algorithme et arithmétiques pour l'implémentation de couplages cryptographiques », thèse de magister, université de Lorraine, octobre 2013.

[9] Cours sécurité informatique, Prof GUEZOURI Mustapha, Université d'Oran 1 Ahmed Ben Bella, 2016.

[10]Dennis J Baker and Anthony Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. Communications, IEEE Transactions on, 29(11):1694 1701, 1981.

[11]Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. Guide to elliptic curve cryptography.
Springer Science & Business Media, 2006.

[12]Lawrence C Washington. Elliptic curves: number theory and cryptography. CRC press, 2008.

[13]. **C**ourbes elliptiques et cryptographie, Hägler Michael, 2006.

[14]Neal Koblitz. Elliptic curve cryptosystems. Mathematics of computation, 48(177) :203209, 1987.

[15]Cryptographie sur les courbes elliptiques, Nicolas Papernot, 2012.

[16]Cryptographie Avancée : Courbes elliptiques, Cécile GONÇALVES, UPEC, 2015.

[17].Brown L Daniel R. Elliptic curve cryptography, May 2009. Certicom Research.

[18]Encryption of data using Elliptic Curve over finite fields, D.Sravana Kumar, CH Suneetha, A.Chadrasekhar, international Journal of Distributed and Parallel Systems, January 2012.

[19]A course in Number Theory and Cryptography, Kobliz, 1994.

[20].Cryptographie et Sécurité informatique, Université de Liège, 2009 - 2010.

[21].Courbes Elliptiques, Application à la Cryptographie, Stéphane Ballet et Alexis Bonecaze, Ecole Poly Tech de Marseille.

[22] Parallel Collision Search with Cryptanalytic Applications, P.C. van Oorschot, M.J. Wiener, Journal of Cryptology, 1996.

[23]Nicholas Jansma and Brandon Arrendondo. Performance comparison of elliptic curve and rsa digital signatures, April 2004.

[24] Arjen K Lenstra and Eric R Verheul. Selecting cryptographic key sizes. *Journal of cryptology*, 14(4):255–293, 2001.

[25] Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and SheuelingChangShantz. Comparing elliptic curve cryptography and rsa on 8-bit cpus. In Marc oye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 119–132. Springer Berlin Heidelberg, 2004.

[26] Priit Karu and Jonne Loikkanen. Practical comparison of fast public-key crypto systems. In Telecommunications Software and Multimedia Lab. at Helsinki Univ. of Technology, Seminar on Network Security. Citeseer, 2001.

[27]Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2): 120–126, 1978.

[28] RSA Laboratories. Pkcs #1: rsa encryption standard, November 1993. URL ftp: //ftp.rsasecurity.com/pub/pkcs/ascii/pkcs-1.asc.

[29] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177): 203–209, 1987.

[30]Victor S Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology—CRYPTO'85 Proceedings*, pages 417–426. Springer, 1986.

[31]Certicom. Certicom intellectual property. https://www.certicom.com/index.php/ \licensing/certicom-ip.

[32]NIST. Recommended elliptic curves for federal government use, 1999. Available at http://csrc.nist.gov/encryption.

[33] D. R. Stinson, Cryptographic: theory and practice, CRC Press, Ibc, 1995

[34]Menezes, A.: Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers, Dordrecht (1993).

[35] Franc¸ois Morain and Jorger Olivos. Speeding up the computation on an elliptic curve using addition-subtraction chains. *Information Theory Appl.*, 24:531–543, 1990.

[36].J.Solinas, *Efficient Arithmetic on Koblitz Curves*, Designs, Codes a Cryptography, 2000.

[37]J.Solinas, *Efficient Arithmetic on Koblitz Curves*, Designs, Codes an Cryptography, 2000.

[38].Joseph. H. Silverman, *The Arithmetic of Elliptic Curves*, GTM 106, Springer-Verlag, New York, 1986.

[39].K. Koyama and Y. Tsuruoka, "Speeding up elliptic cryptosystems by using a signed binary window method", *Advances in Cryptology-Proceedings of Crypto'92*, Lecture Notes in Computer Science, 740(1993), Springer-Verlag,345-357.

[40].D. V. Chudnovsky and G. V. Chudnovsky \Sequences of numbers generated by addition in formal groups and new primality and factorization tests" *Advances in Applied Math* December *1986*, Pages *385-434*.

[41]H. Cohen, A. Miyaji and T. Ono, \Efficient elliptic curve exponentiation", *Advances in Cryptology-Proceedings of ICICS'97*, Lecture.Notes.in.Computer.Science, 1334(1997), Springer-Verlag, 282-290.

[42]Menezes, A.: Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers, Dordrecht (1993).

[43].Cohen, H. and G. Frey, Handbook of Elliptic and Hyper elliptic CurveCryptography2006: Chapman and Hall/CRC.

[44] N. F. H. Al Saffar and M. R. Md Said / International Journal of Computer Networks and Communications Security, 2 (12), December 2014

[45]Pritam Gajkumar Shah., Xu Huang, Dharmendra Sharma, "sliding window method with flexiblewindow size for scalarmultiplication on wireless sensor network nodes." . 2010.

[46]. Xiong H, Hu JB, Chen Z. Security Flaw of an ECC-based Signcryption Scheme with Anonymity. International Journal of Network Security. 2013; 4:317–20.