N° d'ordre : ....

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ MOHAMMED SEDDIK BENYAHIA DE JIJEL
FACULTÉ DE SCIENCES EXACTES ET D'INFORMATIQUE

MÉMOIRE DE MASTER

Présenté pour l'obtention du diplôme de :

MASTER

En **INFORMATIQUE**

Option : INFORMATIQUE LÉGALE ET MULTIMÉDIA

Thème

La Cryptographie à base des Courbes Elliptiques : Étude Comparative

Présenté par :

Lakmiti Imene
Djafri Rania

Encadré par :

Mr. Bouachiba Fouad

Année Universitaire : 2021/2022

À mes très chers parents qui nous ont soutenus et encouragés durant toute notre scolarité

À mes sœurs,

À mes proches,

À mes amis,

À toutes les personnes qui nous ont apportés de l'aide.

Remerciements

Nous tenons à remercier en premier lieu de dieu, qui nous a donné la force et la patience pour accomplir ce Modeste travail.

Mes remerciements s'adressent aussi à Mr Bouachiba Fouad, qui nous a fait orienter notre travail.

Nous tenons à remercier très sincèrement l'ensemble des membres du jury qui nous a fait le grand honneur d'accepter d'examiner notre travail.

Nous remercions nos très chers parents pour leur amour, leurs sacrifices et leurs encouragements.

Enfin, nous adressons mes plus sincères remerciements à tous qui nous ont toujours soutenu et encouragé de près ou de loin.

Résumé

L'humanité a constamment besoin de la cryptographie car la sécurisation des informations sensibles est un problème intéressant.
La cryptographie est couramment utilisée pour protéger les données pendant la transmission et le stockage, Les entreprises s'appuient de plus en plus sur le chiffrement pour protéger les applications et les informations sensibles contre les atteintes à leur réputation en cas de violation de données.
Aujourd'hui, la cryptographie à base des courbes elliptiques (ECC) présente une solution alternative concurrente et de nombreux avantages dont laquelle elle se base sur des problèmes mathématiques difficiles à résoudre tel que logarithme discret dans des corps finis.
Afin de connaître l'efficacité de l' ECC, ses avantages, ses inconvénients et son classement parmi les autres systèmes de chiffrement , nous avons mené cette étude en comparant ECC avec certains cryptosystèmes symétriques et asymétriques les plus connus en termes de la taille de clé et du même niveau de sécurité (génération de clés, chiffrement et déchiffrement) pour donner aux intéressants une image claire sur l'ECC, sa manière de fonctionnement et ses problèmes en comparant avec certains cryptosystèmes connus. Le but est de savoir quand ce système peut être utilisé pour obtenir le meilleur temps d'exécution et la meilleure sécurité possible.
Si l'information arrive un peu tard et en toute sécurité, Mieux que d'arriver rapidement et sans sécurité, La liberté de choisir le système de chiffrement reste entre les mains de l'utilisateur en fonction de ses besoins de cryptage et du domaine dans lequel le système est utilisé. Par exemple, le besoin d'un dossier médical urgent n'est pas le même que le besoin d'un dossier militaire, l'étude que nous avons préparée reste détaillant en facteur de temps et de sécurité pour donner à l'utilisateur une vision plus large pour choisir le system le plus approprié pour leur besoins basé sur les facteurs les plus importants dans le domaine du cryptage.

Mot clés : *Cryptographie, Chiffrement, Courbe elliptique, cryptosystèmes, Déchiffrement, ECC, logarithme discret.*

Abstract

Humanity constantly needs cryptography because securing information sensitive formations is an interesting problem.
Cryptography is commonly used to protect data during transmission and storage, Businesses are increasingly relying on encryption to protect sensitive applications and information from reputational damage in the event of a data breach.
Today, elliptic curve cryptography (ECC) presents a competing alternative solution and many advantages of which it is based on mathematical problems difficult to solve such as discrete logarithm in finite fields. In order to know the effectiveness of ECC, its advantages, disadvantages and its ranking among other cryptosystems, we conducted this study by comparing ECC with some well-known symmetric and asymmetric cryptosystems in terms of key size. and the same level of security (key generation, encryption and decryption) to give interested parties a clear picture of the ECC, its way of operation and its problems by comparing with some known cryptosystems. The goal is to know when this system can be used to obtain the best execution time and the best possible security.
If the information arrives a little late and safe, Better than to arrive quickly and without security, the freedom to choose the encryption system remains in the hands of the user depending on their encryption needs and the domain in which the system is used. For example, the need for an urgent medical file is not the same as the need for a military file, the study that we have prepared remains detailing the time and security factors to give the user a vision to choose the most appropriate system for their needs based on the most important factors in the field of encryption.

Keywords : *Cryptography, Cryptosystems, Decryption, Discrete logarithm, ECC, Elliptical curve, Encryption.*

TABLE DES MATIÈRES

Table des Matières i

Table des figures iv

Liste des tableaux vi

Liste des acronymes vii

Introduction Générale 1

1 la cryptographie 3
- 1.1 Introduction . . . 3
- 1.2 Qu'est-ce que la cryptographie . . . 3
- 1.3 Terminologie de la cryptographie . . . 4
- 1.4 Objectif de la cryptographie . . . 5
- 1.5 Classes de la cryptographie . . . 5
- 1.6 Cryptographie moderne . . . 6
  - 1.6.1 Cryptographie symétrique (à clé privée ou secret) . . . 6
    - 1.6.1.1 Catégories de chiffrement dans la cryptographie symétrique 6
    - 1.6.1.2 Algorithmes les plus connus dans la cryptographie symétrique . . . 12
    - 1.6.1.3 Propriétés de la cryptographie symétrique . . . 16
    - 1.6.1.4 Avantages et les inconvénients de la cryptographie symétrique . . . 16
  - 1.6.2 Cryptographie asymétrique . . . 17
    - 1.6.2.1 Principe de fonctionnement . . . 17
    - 1.6.2.2 Algorithmes les plus connus dans la cryptographie asymétrique . . . 18
    - 1.6.2.3 Fonction de hachage . . . 20
    - 1.6.2.4 Avantages et les inconvénients de cryptographie asymétrique 20
  - 1.6.3 Cryptographie hybride . . . 20
- 1.7 Cryptographie future . . . 22
  - 1.7.1 Cryptographie quantique . . . 22

1.7.1.1 Deux protocoles fondamentaux de la cryptographie quantique . . . 22
1.8 Conclusion . . . 22

2 La cryptographie à base des courbes elliptiques (ECC). 23
2.1 Introduction . . . 23
2.2 Généralités . . . 23
2.2.1 Groups . . . 23
2.2.1.1 Groupe Abélien . . . 24
2.2.1.2 Groupe cyclique . . . 24
2.2.2 Corps . . . 24
2.2.2.1 Corps fini . . . 25
2.3 Présentation des courbes elliptiques . . . 25
2.3.1 Equation de Weierstrass . . . 25
2.3.2 Définition d'une courbes elliptique . . . 26
2.3.3 Problème de logarithme discret sur les courbes elliptiques . . . 27
2.4 Arithmétique sur les courbes elliptiques . . . 27
2.4.1 Multiplication des points . . . 28
2.4.2 Addition des points . . . 28
2.4.3 Doublement de points . . . 29
2.4.4 Nombre des points sur une courbe elliptique . . . 30
2.4.5 Cryptographie à base des courbes elliptiques (ECC) . . . 30
2.4.5.1 Génération des clés ECC . . . 31
2.4.5.2 Chiffrement de ECC . . . 31
2.4.5.3 Déchiffrement de ECC . . . 32
2.5 Cryptosystèmes basé sur les courbes elliptiques . . . 32
2.5.1 Protocole d'échange de Diffie-Hellman ECDH . . . 32
2.5.2 Cryptosystème El Gamal sur des courbes elliptiques . . . 33
2.5.3 Signature numérique . . . 34
2.6 Domaines d'utilisation de la cryptographie à base des courbes elliptiques . . . 36
2.7 Conclusion . . . 37

3 Positionnement d'ECC entre les systèmes de chiffrement 38
3.1 Introduction . . . 38
3.2 L'environnement de développement . . . 38
3.3 Applets de simulation . . . 39
3.4 L'architecture de notre système . . . 40
3.5 Quelques interfaces de notre application . . . 40
3.5.1 Captures d'écran du l'exécution des systèmes . . . 42
3.6 Détails de l'implémentation d'ECC . . . 46
3.6.1 Temps d'exécution du l'ECC . . . 47
3.7 Expérimentation et analyse des résultats . . . 49
3.7.1 ECC vs RSA . . . 49
3.7.1.1 Comparaison de changement du temps d'exécution ECC et RSA sur les tailles de clé . . . 49

3.7.1.2 Comparaison de changement du temps d'exécution ECC et RSA sur le même niveau de sécurité (équivalente des clé) : 52
3.7.2 ECC vs El Gamal . . . 55
3.7.2.1 Comparaison de changement du temps d'exécution ECC et El Gamal sur les taille de clé . . . 55
3.7.2.2 Comparaison de changement du temps d'exécution ECC et El Gamal sur le même niveau de sécurité (équivalente des clé) : . . . 57
3.7.3 ECC vs Diffie Hellman . . . 60
3.7.3.1 Comparaison de changement du temps d'exécution ECC et Diffie Hellman sur les tailles de clé . . . 60
3.7.4 ECC vs AES . . . 61
3.7.4.1 Comparaison de changement du temps d'exécution ECC et AES sur les tailles de clé . . . 61
3.7.4.2 Comparaison de changement du temps d'exécution ECC et AES sur le même niveau de sécurité (équivalente des clé) 64
3.7.5 ECC Vs TripleDES . . . 66
3.7.5.1 Comparaison de changement du temps d'exécution ECC et TripleDES sur les taille de clé . . . 66
3.7.6 ECC vs RC4 . . . 69
3.7.6.1 Comparaison de changement du temps d'exécution ECC et RC4 sur les tailles de clé . . . 69
3.7.7 ECC vs tous les Systèmes étudiés . . . 72
3.7.7.1 Comparaison de changement du temps d'exécution ECC et tous les systèmes étudiés sur les tailles de clé . . . 72
3.7.7.2 Comparaison de changement du temps d'exécution ECC et tous les Systèmes de chiffrement étudiés sur le même niveau de sécurité (équivalente des clé) . . . 74
3.8 Discussion des résultats . . . 77
3.8.1 ECC Vs Les systèmes symétriques . . . 77
3.8.2 ECC Vs Les systèmes Asymétriques . . . 78
3.9 Conclusion . . . 79

Conclusion générale **80**

Bibliographie **vii**

TABLE DES FIGURES

1.1 Principes de base de la cryptographie. 4
1.2 Cryptage et décryptage. 5
1.3 Classes de la cryptographie 6
1.4 Chiffrement symétrique 6
1.5 Electronic code book (ECB) 8
1.6 Cipher FeedBack (CFB) 9
1.7 Cipher Block Chaining (CBC) 9
1.8 Output FeedBack (OFB) 10
1.9 CounTeR (CTR) 11
1.10 Chiffrement par flux 12
1.11 Organigramme de l'algorithme DES 12
1.12 Schéma de DES 13
1.13 Déroulement du chiffrement AES 15
1.14 Chiffrement asymétrique 17
1.15 Principe des fonctions de hachage 20

2.1 Quelques courbes elliptiques. 27
2.2 Algorithme de multiplication scalaire 28
2.3 Addition de point P et Q sur un courbe elliptique 29
2.4 Doublement de point sur un courbe elliptique 30
2.5 Hiérarchie des opérations sur les courbes elliptiques en cryptographie 31
2.6 Algorithme de ECDH 33
2.7 Échange de clé Diffie-Hellman en utilisant les courbes elliptiques 33
2.8 Algorithme de encrypte ECC-El Gamal 34
2.9 Algorithme de décrypte ECC-El Gamal 34
2.10 Algorithme de ECDSA Signature 36
2.11 Algorithme de ECDSA Vérification 36
2.12 Courbe Secp256k1 utilisé par Bitcoin. 37

3.1 L'architecteur de système. 40
3.2 Interface d'accueil de l'application. 41
3.3 Systèmes de chiffrement dans l'application. 41

3.4 Interface de l'application au résultat de l'exécution du système ECC. . . . 42
3.5 Interface de l'application au résultat de l'exécution du système RSA. . . . 43
3.6 Interface de l'application au résultat de l'exécution du système El Gamal. . 43
3.7 Interface de l'application au résultat de l'exécution du système Diffie Hellman. 44
3.8 Interface de l'application au résultat de l'exécution du système AES. 44
3.9 Interface de l'application au résultat de l'exécution du système TripleDES. 45
3.10 Interface de l'application au résultat de l'exécution du système RC4. 45
3.11 Courbe de changement du temps d'exécution (ms) de génération des clés ECC en fonction des tailles des clés. 48
3.12 Courbe de changement du temps d'exécution (ms) de chiffrement ECC en fonction des tailles des clés. 48
3.13 Courbe de changement du temps d'exécution (ms) de déchiffrement ECC en fonction des tailles des clés. 49
3.14 Courbe comparative du temps d'exécution de génération de clé entre ECC et RSA. 50
3.15 Courbe comparative du temps d'exécution de chiffrement entre ECC et RSA. 51
3.16 Courbe comparative du temps d'exécution de déchiffrement entre ECC et RSA. 52
3.17 Courbe comparative du temps d'exécution de génération de clé entre ECC et RSA dans le cas de clés équivalentes. 53
3.18 Courbe comparative du temps d'exécution de chiffrement entre ECC et RSA dans le cas de clés équivalentes. 54
3.19 Courbe comparative du temps d'exécution de déchiffrement entre ECC et RSA dans le cas de clés équivalentes. 54
3.20 Courbe comparative du temps d'exécution de génération de clé entre ECC et El Gamal. 55
3.21 Courbe comparative du temps d'exécution de chiffrement entre ECC et El Gamal. 56
3.22 Courbe comparative du temps d'exécution de déchiffrement entre ECC et El Gamal. 57
3.23 Courbe comparative du temps d'exécution de génération de clé entre ECC et El Gamal dans le cas de clés équivalentes. 58
3.24 Courbe comparative du temps d'exécution de chiffrement entre ECC et El Gamal dans le cas de clés équivalentes. 59
3.25 Courbe comparative du temps d'exécution de déchiffrement entre ECC et El Gamal dans le cas de clés équivalentes. 60
3.26 Courbe comparative du temps d'exécution de génération de clé entre ECC et Diffie Hellman. 61
3.27 Courbe comparative du temps d'exécution de génération de clé entre ECC et AES. 62
3.28 Courbe comparative du temps d'exécution de chiffrement entre ECC et AES. 63
3.29 Courbe comparative du temps d'exécution de déchiffrement entre ECC et AES. 64
3.30 Courbe comparative du temps d'exécution de génération de clé entre ECC et AES dans le cas de clés équivalentes. 65

3.31 Courbe comparative du temps d'exécution de chiffrement entre ECC et AES dans le cas de clés équivalentes. 65
3.32 Courbe comparative du temps d'exécution de déchiffrement entre ECC et AES dans le cas de clés équivalentes. 66
3.33 Courbe comparative du temps d'exécution de génération de clé entre ECC et TripleDES. 67
3.34 Courbe comparative du temps d'exécution de chiffrement entre ECC et TripleDES 68
3.35 Courbe comparative du temps d'exécution de déchiffrement entre ECC et TripleDES. 69
3.36 Courbe comparative du temps d'exécution de génération de clé entre ECC et RC4. 70
3.37 Courbe comparative du temps d'exécution de chiffrement entre ECC et RC4. 71
3.38 Courbe comparative du temps d'exécution de déchiffrement entre ECC et RC4. 72
3.39 Courbe comparative du temps d'exécution de génération de clé entre ECC et tous les Systèmes étudiés. 73
3.40 Courbe comparative du temps d'exécution de chiffrement entre ECC et tous les systèmes étudiés. 73
3.41 Courbe comparative du temps d'exécution de déchiffrement entre ECC et tous les systèmes étudiés. 74
3.42 Courbe comparative du temps d'exécution de génération de clé entre ECC et tous les systèmes de chiffrement dans le cas de clés équivalentes. 75
3.43 Courbe comparative du temps d'exécution de chiffrement entre ECC et tous les Systèmes de chiffrement dans le cas de clés équivalentes. 76
3.44 Courbe comparative du temps d'exécution de déchiffrement entre ECC et tous les Systèmes de déchiffrement dans le cas de clés équivalentes. 77

LISTE DES TABLEAUX

1.1 Avantages et inconvénients de cryptographie symétrique 17
1.2 Avantages et inconvénients de cryptographie asymétrique. 21

3.1 Courbe P-160. 46
3.2 Courbe P-256. 46
3.3 Courbe P-384. 47
3.4 Courbe P-521. 47
3.5 Comparaison des tailles de clés de ECC et du temps d'exécution pour la génération des clés, le chiffrement et le déchiffrement de ECC. 48
3.6 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au RSA. 49
3.7 Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au RSA. 50
3.8 Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au RSA. 51
3.9 Comparaison les taille de clé entre ECC et RSA pour la force de chiffrement. 52
3.10 Comparaison du temps d'exécution entre ECC et RSA en mode équivalent sur les tailles des clés. 53
3.11 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au El Gamal. 55
3.12 Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au El Gamal. 56
3.13 Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au El Gamal. 57
3.14 Comparaison les taille de clé entre ECC et El Gamal pour la force de chiffrement. 58
3.15 Comparaison du temps d'exécution entre ECC et El Gamal en mode équivalent sur les tailles des clés. 58
3.16 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au Diffie Hellman. 60
3.17 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au AES. 61

3.18 Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au AES. 62
3.19 Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au AES. 63
3.20 Comparaison du temps d'exécution entre ECC et AES en mode équivalent sur les tailles des clés. 64
3.21 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au TripleDES. 66
3.22 Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au TripleDES. 67
3.23 Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au TripleDES 68
3.24 Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au RC4. 69
3.25 Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au RC4. 70
3.26 Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au RC4 71
3.27 Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes étudiés pour la génération des clés en mode équivalent les taille de clé . . . 75
3.28 Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes pour le chiffrement en mode équivalent. 75
3.29 Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes pour le déchiffrement en mode équivalent. 76
3.30 Résultat de ECC Vs Symétrique. 77
3.31 Résultat de ECC Vs Asymétrique en terme des tailles de clé. 78
3.32 Résultat de ECC Vs Asymétrique en même niveau de sécurité. 78

LISTES DES ACRONYMES

AES	Advanced Encryption Standard
CBC	Cipher-Block Chaining
CFB	Cipher Feedback mode
CTR	CounTeR
DES	Data Encryption Standard
3DES	Triple Data Encryption Standard
DH	Diffie-Hellman
ECB	Electronic Code Book
ECC	Elliptic curve cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithme
GF(p)	Corps Fini (Galois Field) premier à p éléments
GF(2n)	Corps Fini (Galois Field) binaire à 2n éléments
GnuPG	GNU PrivacyGuard
IDE	Integrated Development Envirenment
IV	Initialization Vector
JDK	JAVA Developement Kit
MD5	Message Digest5
NIST	National Institute of Standards and Technology
NSA	National Security Agency
OFB	Output Feedback mode
PGP	Pretty Good Privacy
RC4	Rivest Cipher 4
RSA	Rivest Shamir Adleman
SHA	Secure Hach Algorithm
TLS	Transport Layer Security
XOR	eXclusive OR

INTRODUCTION GÉNÉRALE

Le mot cryptographie vient des mots en grec ancien «kruptos» qui signifie caché et «graphein» qui signifie écrire, beaucoup des termes de la cryptographie utilisent la racine crypto, ou des dérivés du terme chiffre, la cryptographie utilisé depuis des milliers d'années pour assurer les communications militaires et diplomatiques.
La cryptographie est un domaine dont l'objectif principal est de protéger l'information, de la rendre inintelligible à celles ou ceux à qui elle n'est pas destinée. Elle repose sur des algorithmes solides qui s'appuient eux-mêmes sur des problèmes mathématiques réputés difficiles (logarithme discret, factorisation des grands nombres, etc..) .
Les systèmes de chiffrement symétriques sont basés sur la substitution et la permutation, et les systèmes de chiffrements asymétriques sont basés sur des problèmes mathématiques difficiles tels que la factorisation des nombres premiers et logarithme discret.
une courbe elliptique est un cas particulier de courbe algébrique, munie entre autres propriétés d'une addition géométrique sur ses points, La cryptographie à base des courbes elliptiques (ECC) fait partie des méthodes asymétriques présente une solution alternative concurrente et de nombreux avantages dont laquelle elle se base sur des problèmes mathématiques difficiles à résoudre tel que logarithme discret dans des corps finis, Son principal avantage est que lorsque la courbe est bien choisie, il n'y a pas d'algorithme sous-exponentiel connu qui puisse résoudre le problème du logarithme discret pour ce groupe.
Dans notre projet, nous avons fait une étude plus spécifique sur le fonctionnement et le positionnement de la cryptographie à base des courbes elliptiques (ECC) par rapport à certains systèmes symétriques et asymétriques bien connus, Actuellement, il existe de nombreux systèmes de cryptage qui ont un haut niveau de sécurité et de rapidité, Nous avons donc décidé de faire une étude comparative entre ECC et certains des systèmes bien connus symétrique comme AES, RC4, Triple DES et asymétrique comme RSA, El Gamal et Diffie Hellman pour connaître les performances (efficacité), la classification et localisation d'ECC entre les systèmes de chiffrement décrits précédemment, leurs avantages et leurs inconvénients, et où nous pouvons les utiliser, et y a-t-il une différence avec les autres systèmes en termes du temps d'exécution et du niveau de sécurité.

Organisation du Mémoire

Ce mémoire est structuré en trois chapitres.

Nous présentons dans le première chapitre les principaux concepts et les terminologie de la cryptographiques, leurs classes et les algorithme de cryptographie les plus connue.

Nous présentons dans le deuxième chapitre les courbes elliptiques et la notion de cryptographie à base des courbes elliptiques, son algorithme, quelques protocoles basés sur cette dernière et leurs utilisations.

Dans le troisième chapitre, nous avons présenté l'application comparative que nous avons implémenté et des captures d'écran de celle-ci et les moyens que nous avons utilisés, plus une comparaison analytique d'ECC avec d'autres systèmes de chiffrement les plus connus en termes de taille de clé et en termes de même niveau de sécurité.

CHAPITRE 1

LA CRYPTOGRAPHIE

1.1 Introduction

Le développement croissant des nouvelles technologies et l'utilisation excessive des nouveaux moyens de communication ont donné la naissance à des nouveaux problèmes. ces problèmes liés essentiellement à la sécurité des données. En effet, La sécurité est assurée grâce à la cryptographie qui a comme objectif de permettre à deux entités communicantes d'échanger d'une manière efficace et sécurisée des données confidentielles. la cryptographie est l'étude des principes, méthodes et techniques mathématiques reliées aux aspects de sécurité de l'information telles la confidentialité, l'intégralité des données, authentification d'entités et l'originalité des données. dans ce chapitre, nous ferons une étude qui comprend des sections expliquant la technologie de la cryptographie, ses types (symétrique et asymétrique), ainsi que les algorithmes de cryptographie les plus couramment utilisés.

1.2 Qu'est-ce que la cryptographie

Le terme cryptographie est un terme général qui désigne toutes les technologies permettant de chiffrer des messages [1], c'est-à-dire qu'ils peuvent être rendus incompréhensibles sans opérations spécifiques. Si le but traditionnel de la cryptographie est d'élaborer des méthodes permettant de transmettre des données de manière confidentielle, la cryptographie moderne s'attaque en fait plus généralement aux problèmes de sécurité des communications [2] [3].
Son objectif principal est de permettre à deux personnes de communiquer à travers un canal sécurisé de telle sorte qu'un opposant comme le montre la figure 1.1, le personne tenant un point d'interrogation ne puisse pas comprendre ce qui est échangé, on utilise une clé appelée clé de chiffrement pour le processus de chiffrement. pour rendre l'information à nouveau compréhensible on utilise une clé appelée clé de déchiffrement pour le processus de déchiffrement [4].

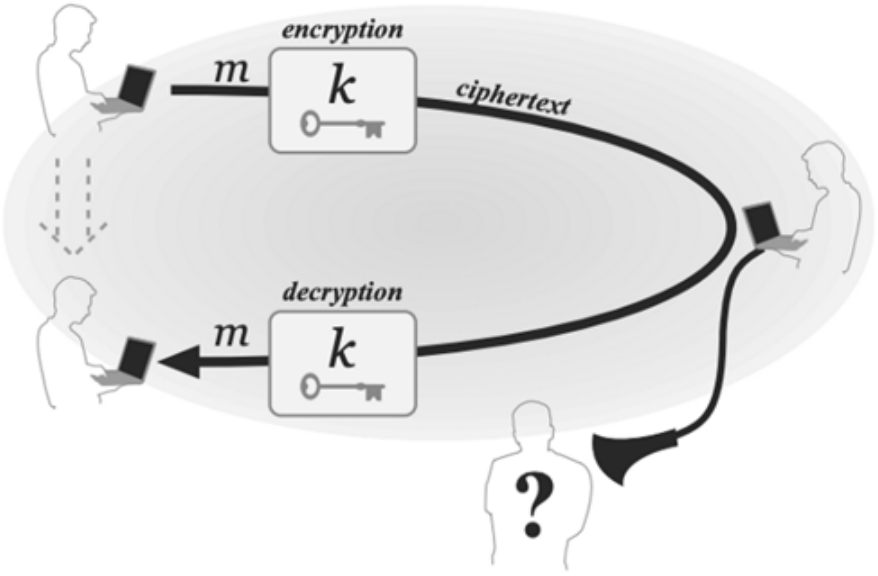

FIGURE 1.1 – Principes de base de la cryptographie [5]

1.3 Terminologie de la cryptographie

Voici quelques notions et terminologie importantes de la cryptographie : [6] [7]
La cryptographie est l'art de masquant contenu d'un message en clair avec une clé.
La cryptologie est la branche des mathématiques comprenant la cryptographie et la cryptanalyse.
Un cryptosystème est un système de chiffrement constitué d'un algorithme cryptographique.
La cryptanalyse l'art de décrypter sans avoir la clé secrète afin de trouver les failles du cryptosystèmes.
La stéganographie C'est l'art qui consiste à dissimuler un message dans un autre message par exemple dans une image ou bien un texte, la différence entre la stéganographie et la cryptographie réside dans l'aspect de sécurité ou la sécurité de la cryptographie consiste à rendre un message incompréhensible, en revanche la sécurité par la stéganographie consiste seulement à cacher un message, plusieurs techniques de la stéganographie existent pour pouvoir cacher un message par exemple cacher du texte dans une image ou bien utiliser la technique de décomposition en plan de bits pour pouvoir modifier le bit de poids faible des pixels codant l'image. [7]
Cryptogramme (Texte chiffré : Ciphertext) est le résultat de l'application d'un chiffrement d'un texte clair.
Texte clair (Plaintext) le message à chiffrer.
Chiffrement est la fonction permettant de transformer une donnée (texte, message ...) afin de la rendre incompréhensible par une personne autre que celui qui créé le message et ainsi que le destinataire. Noté E_k, est l'action de chiffrer un **message en clair**, noté **M**, en un **message chiffré**, noté **C**, et cela de façon à ce qu'il soit impossible de retrouver le message en clair à partir du message chiffré sans la clé.
Déchiffrement Le contraire de chiffrement est la fonction qui vous permet de trouver du

texte clair à partir du texte chiffré.
Clé est un paramètre utilisé en entrée d'une opération cryptographique (chiffrement, déchiffrement), On distingue généralement deux types de clefs :
Clés symétriques il s'agit de clés utilisées pour le chiffrement ainsi que pour le déchiffrement, on parle alors de chiffrement symétrique ou à clé secrète.
Clés asymétriques il s'agit de clés utilisées dans le cas du chiffrement asymétrique ou à clé publique, dans ce cas, une clé différente est utilisée pour le chiffrement et pour le déchiffrement [6].
Le cryptage c'est le chiffrement d'un message claire et **le décryptage** est le déchiffrement d'un message chiffré, comme on a résumé dans la figure 1.2 quelques termes précédents pour expliquer le cryptage et le décryptage [8].

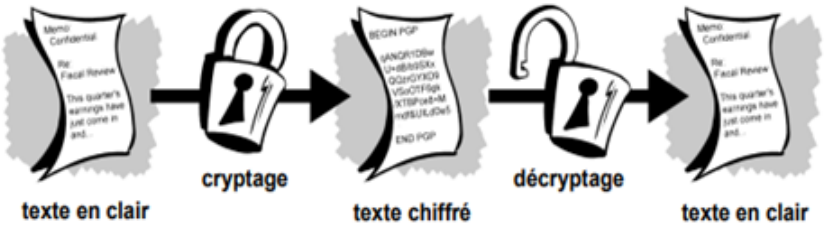

FIGURE 1.2 – Cryptage et décryptage

1.4 Objectif de la cryptographie

Cette partie introduira les mécanismes de base de la cryptographie moderne qui permettent de réaliser quatre services de sécurité fondamentaux : la confidentialité, l'intégrité, l'authentification, la non répudiation [9] [10] [11].

- **La confidentialité** est l'attribut qui garantit que les informations ne sont pas comprises par des personnes, entités et processus non autorisés.
- **L'intégrité** C'est la propriété qui permet de vérifier qu'une donnée n'a pas été modifiée par une entité tierce (accidentellement ou intentionnellement).
- **L'authentification** C'est l'attribut qui vérifie que la source de données est bien l'identité revendiquée.
- **La non répudiation** de la source garantit que l'expéditeur du message ne peut pas nier que le message a été envoyé à l'avenir.

1.5 Classes de la cryptographie

La figure 1.3 présente les différentes classes de la cryptographie : cryptpgraphie classique (chiffrement par substitution et chiffrement par transposition), cryptographie moderne (chiffrement à clé publique, chiffrement à clé privée, chiffrement hybride) et la cryptographie future (chiffrement quantique) [8].

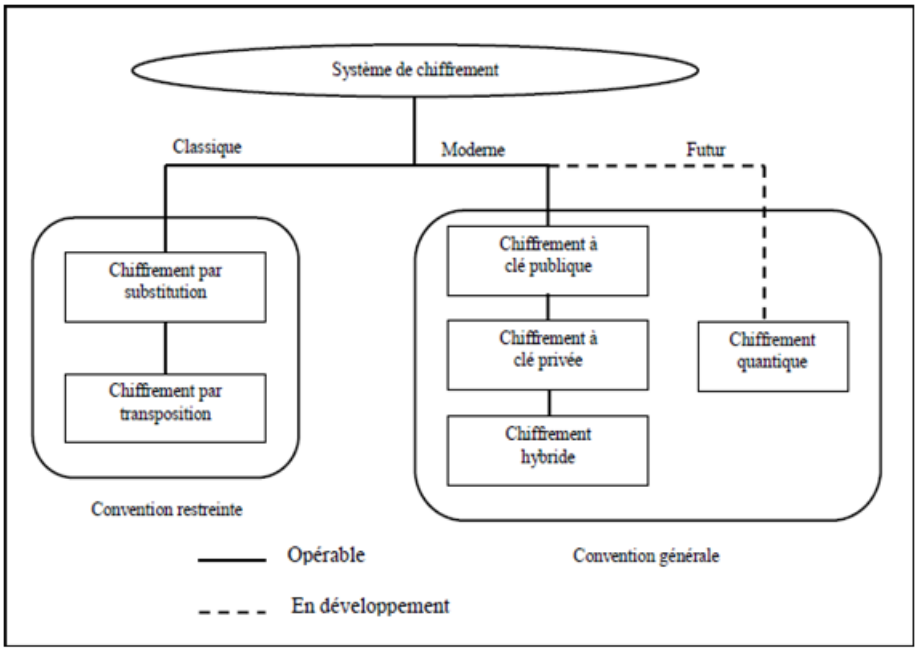

FIGURE 1.3 – Classes de la cryptographie

1.6 Cryptographie moderne

1.6.1 Cryptographie symétrique (à clé privée ou secret)

La cryptographie à clé privée ou (symétrique/secrète), est un mécanisme utilisé le même clé pour le chiffrement et le déchiffrement, la clé doit être unique et elle ne peut plus être utilisée après la fin de communication (figure 1.4) [12].
La transmission sûre de la clé reste le défi majeur de ce mode de cryptage [9].

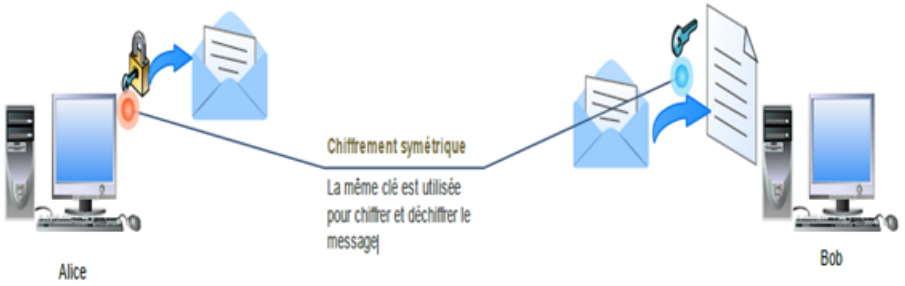

FIGURE 1.4 – Chiffrement symétrique

1.6.1.1 Catégories de chiffrement dans la cryptographie symétrique

Il existe deux catégories de chiffrement : chiffrement par bloc et chiffrement par flux [8] [10] [13].

1.6.1.1.1 Chiffrement par bloc

Division du texte clair en blocs fixe, puis chiffrement bloc par bloc, est le plus répandu et possède d'une meilleure réputation que le chiffrement par flux facile à

analyse mathématique, les étapes générales du chiffrement par blocs symétrique ou à clef secrète est le suivant : [8] [10] [13]

(a) Coder l'information source en binaire. on obtient ainsi une chaine de caractères composée 0 et de 1.

(b) Découper cette chaine en blocs de longueur donnée (par exemple 64 bits ou 128 bits ou 256 bits).

(c) Chiffrer un bloc en faisant un OU exclusif (ou XOR) bit à bit avec une clé secrète, k, qui est une suite de 0 et de 1 de même longueur, (un XOR est donc l'addition sans retenu en base deux).

(d) Déplacer et permuter certains bits du bloc.

(e) Recommencer un certain nombre de fois l'étape précédente, on appelle cela une ronde.

(f) Passer au bloc suivant et retourner à l'étape (c) jusqu'à ce que tous les blocs soient chiffres.

Exemple des algorhithmes de mode de fonctionnement :

— DES : IBM, Standard NIST 1976.

— 3DES : W. Diffie, M. Hellman, W. Tuchmann 1978.

— IDEA : Xuejia Lai et James Massey en 1992.

— Blowfish : Bruce Schneier en 1993.

— AES (Rijndael) : Joan Daemen et Vincent Rijmen 2000.

Dans le chiffrement symétrique l'algorithme opère sur un bloc. Pour chiffrer un ensemble de blocs constituants le message à chiffrer il est nécessaire de définir une stratégie d'opération sur la succession des blocs à chiffrer.
Il existe plusieurs modes définis par FIPS 81 (1980) : ECB, CFB, CBC, OFB, CTR. nous adoptons la notation suivante dans la description des quatre modes d'opération :

— mi : n-ième bloc du texte clair

— Ci : n-ième bloc du texte chiffré

— E (mi) : fonction de chiffrement

— D (ci) : fonction de déchiffrement

— $\bigoplus$: XOR

(a) **Mode Electronic Code Book (ECB)** :
Le mode ECB est le plus simple, Le message M est découpé en blocs m_i, $i \geq 1$, et chaque bloc est crypté séparément par $c_i = E(m_i)$ où $E = E_k$ dépend de la clé secrète k et c_i est le bloc crypté correspondant, comme nous l'avons illustrer dans la figure 1.5 : [8] [13]

— Chiffrement : $C_i = E(m_i)$

— Déchiffrement : $m_i = D(C_i)$

— Le même texte clair et clé de chiffrement donnent le même texte chiffré.

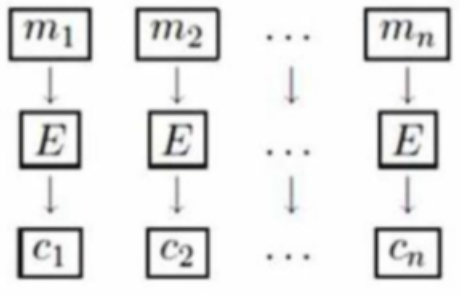

FIGURE 1.5 – Electronic code book (ECB) mode [8]

(b) **Mode CFB, Cipher FeedBack :**
Le mode CFB est introduit pour qu'il ne soit pas nécessaire de calculer la fonction inverse D_k de la fonction suivante chiffrage E_k, le message M est divisé en blocs $(mi), i \geq 1$, et chaque bloc est chiffré comme suit. nous choisissons d'abord un bloc initial m_0, selon le schéma qui representé dans la figure 1.6, chaque bloc de texte en clair m_i est XORé avec le chiffrement du bloc de sortie précédent c_{i-1} [8] [13].

— Chiffrement :
  — $C_1 = m_1 \bigoplus E_k(c_0)$
  — $C_2 = m_2 \bigoplus E_k(c_1)$
  — $C_i = m_i \bigoplus E_k(C_{i-1}), si(i > 0)$
— Déchiffrement :
  Ce mode est moins sûr que le CBC et est utilisé par exemple pour les cryptages réseaux, l'intérêt est que le déchiffrement ne nécessite pas de calculer D_k : [8]
  — $m_i = C_i \bigoplus E_k(c_{i-1})$

IV est envoyé en clair avec le message chiffré (se souvenir que l'on calcule modulo 2 sans retenue, bit à bit) d'où un gain de temps.

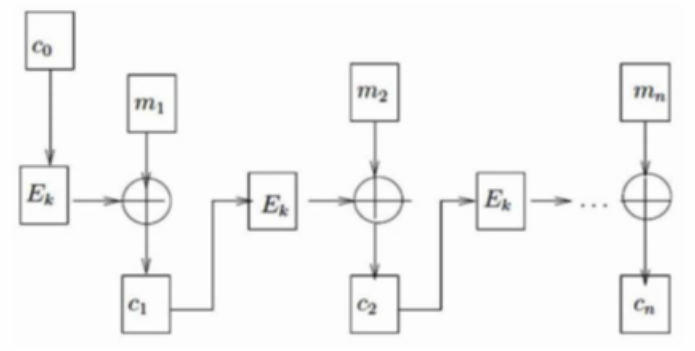

FIGURE 1.6 – Cipher FeedBack (CFB) [8]

(c) **Mode CBC, Cipher Block Chaining :**
Le mode CBC est introduit, donc si le bloc apparaît dans deux messages différents, ou il apparaît deux fois dans un message, le message M est divisé en blocs (m_i) $i \geq 1$, et chaque bloc est chiffré comme suit. nous sélectionnons d'abord un bloc initial c_0. chaque bloc clair m_i est d'abord modifié en effectuant XOR, ce bloc et le bloc chiffré précédent, c_{i-1}, puis le résultat de XORization avec la clé, selon le schéma qui representé dans la figure 1.7 [8] [13].

- Chiffrement :
  - $C_1 = E_k(m_1 \bigoplus C_0)$
  - $C_2 = E_k(m_2 \bigoplus C_1)$
  - $C_i = E_k(m_i \bigoplus C_{i-1})$
- Déchiffrement : Le déchiffrement nécessite de connaitre la fonction inverse de la fonction de codage $D_k = E_k^{-1}$ pour décrypter
  - $m_i = C_{i-1} \bigoplus D_k(c_i)$

Offre une sécurité plus élevée et n'affaiblit pas le cryptosystèmes, mais il nécessite de connaitre la fonction inverse D_k de E_k.

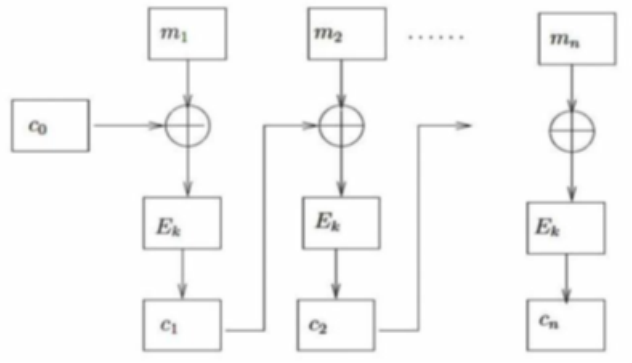

FIGURE 1.7 – Cipher Block Chaining (CBC) [8]

(d) **Mode Output FeedBack (OFB) :**
Le mode OFB montré sur le shéma dans la figure 1.8 est une variante de CFB

qui permet d'avoir un cryptage et un décryptage totalement symétrique : [8] [13]
- I[n] =nième bloc temporaire
- R[n] =nième bloc temporaire second

— Chiffrement :

— $I[0] = C_0$

— $I[n] = R[n-1], si(n > 0)$

— $R[n] = E(I[n])$

— $C_i = m_i \bigoplus R[n]$

— Déchiffrement :

— $I[0] = C_0$

— $I[n] = R[n-1], si(n > 0)$

— $R[n] = E(I[n])$

— $m_i = C_i \bigoplus R[n]$

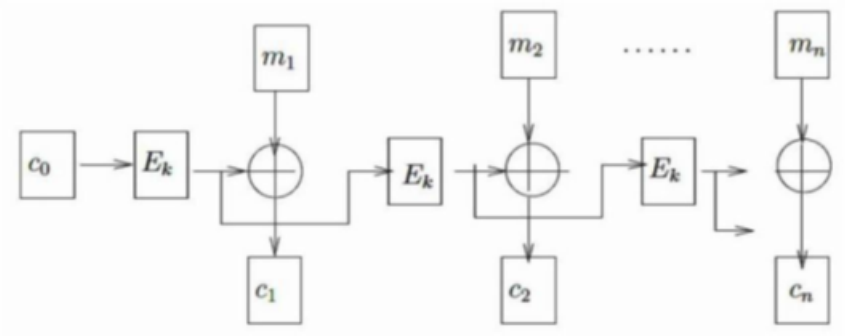

FIGURE 1.8 – Output FeedBack (OFB) [8]

Sa sécurité est équivalente à celle du mode CFB.

(e) **CounTeR (CTR)**
Le mode CTR est lui aussi totalement symétrique, mais en outre facilement parallélisable, il utilise pour le chiffrement un compteur de valeur initiale T, selon le schéma qui representé dans la figure 1.9 [8] [13].
$C_i = m_i \bigoplus E_k(T + i)$

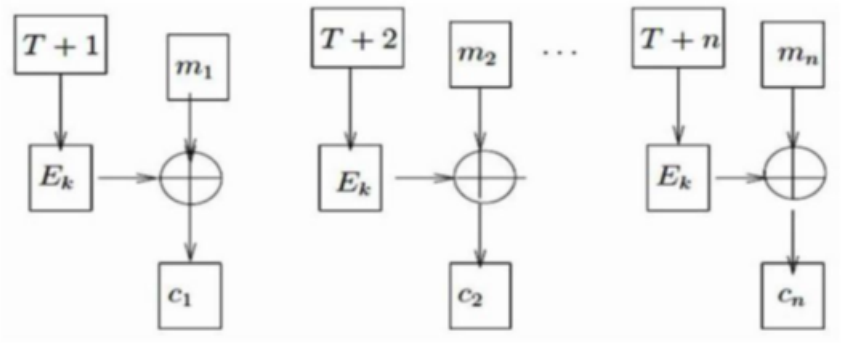

FIGURE 1.9 – CounTeR (CTR) [8]

Le déchiffrage est identique, et sa sûreté est équivalente à celle du mode CFB.

1.6.1.1.2 **Chiffrement par flux**

Cette partie est extraite de plusieurs références [13] [14] [15] [16] dont l'objectif d'exprimer le chiffrement par flot. ce dernier arrive à traiter les données de longueur quelconque et n'a pas besoin de les découper, les systèmes de chiffrement à flux sont aussi appelés chiffrement à la volée. dans un système de chiffrement à flot, la clé utilisée est une suite chiffrant que l'on additionne bit à bit au message clair. la sécurité des systèmes de chiffrement à flot repose alors sur le comportement aléatoire de la suite chiffrant. les systèmes de chiffrement à flot sont des systèmes de chiffrement rapides et bien adaptés pour les implémentations matérielles courantes, le chiffrement effectue une opération booléenne, connue sous le nom de OU exclusif, entre la clé et le texte en clair pour produire un texte chiffré.

Exemples d'algorithmes de chiffrement par flux :

- RC4 : le plus répandu, conçu en 1987 par Ronald Rivest, utilisé notamment par le protocole WEP du Wifi.
- Py : un algorithme récent de Eli Biham .
- SEAL : Don Coppersmith et Phillip Rogaway pour IBM 1993.
- E0 : utilisé par le protocole Bluetooth, pour protéger la confidentialité des communications dans le protocole de transmission sans fil Bluetooth à courte portée.
- A5 : Un algorithme publié en 1994 pour une utilisation dans les téléphones mobiles de type GSM pour crypter les communications radio entre le téléphone mobile et la station de base la plus proche

chiffrement par flux utilisé la clé k et un initialisation vecteur IV pour générer des bits pseudo-aléatoires XORé avec un bit de message claire, le résultat est le bit de message chiffré comme on a présenté dans la figure 1.10 .

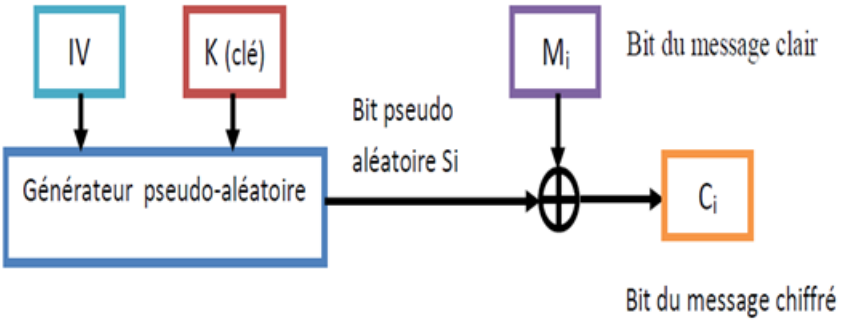

FIGURE 1.10 – Chiffrement par flux [17]

1.6.1.2 Algorithmes les plus connus dans la cryptographie symétrique

1. **DES**
   Aperçu général DES (Data Encryption Standard) est l'un des premiers algorithmes de chiffrement symétrique, il est basé sur un ensemble de permutations et de substitutions, comme indiqué ci-dessous, c'est un algorithme qui s'exécute sur des blocs de 64 bits et utilise une clé de 56 bits suffisante à l'époque, il a été formellement défini dans FIPS46-3, il se compose de l'arrangement initial, du calcul basé sur la valeur médiane de la clé, et de l'arrangement final. la figure 1.11 illustre la disposition initiale des blocs de 64 bits pour les opérations DES [10] [18].

FIGURE 1.11 – Organigramme de l'algorithme DES [5]

1.1 Algorithme DES :
L'algorithme DES est relativement simple puisqu'il combine des permutations et des substitutions, on donne tout d'abord une description générale de ce système qui se déroule en trois étapes : [8] [18]

(a) Etant donné un bloc de texte clair x. une chaîne de bits x0 est construite en changeant l'ordre des bits de x suivant une permutation initiale IP fixée. On écrit : $x_0 = IP(x) = L_0R_0$
Où L_0 contient les 32 premiers bits de la chaîne x_0 et R_0 contient les 32 bits restants.

(b) 16 itérations (ou 16 tours) d'une certaine fonction sont effectuées. chaque tour suit le même schéma qui consiste à prendre en entrée 32 bits : L_{i-1} et R_{i-1} du tour précédent et de produire de nouveau 32 bits L_i et R_i de la manière suivante : $L_i = R_{i-1}$ et $R_i = L_{i-1} \bigoplus f(R_{i-1}, k_i)$

(c) Après le dernier tour, les moitiés gauche et droite (L_{16} et R_{16}) sont échangées puis le texte sera permuté bit à bit par IP^{-1} pour obtenir le bloc de texte chiffré y. Plus formellement, y s'obtient comme suit : $y = IP^{-1}(R_{16}L_{16})$.

Cette description peut être résumée par le schéma général illustré par la figure 1.12, qui résume les 16 itérations de l'algorithme DES.

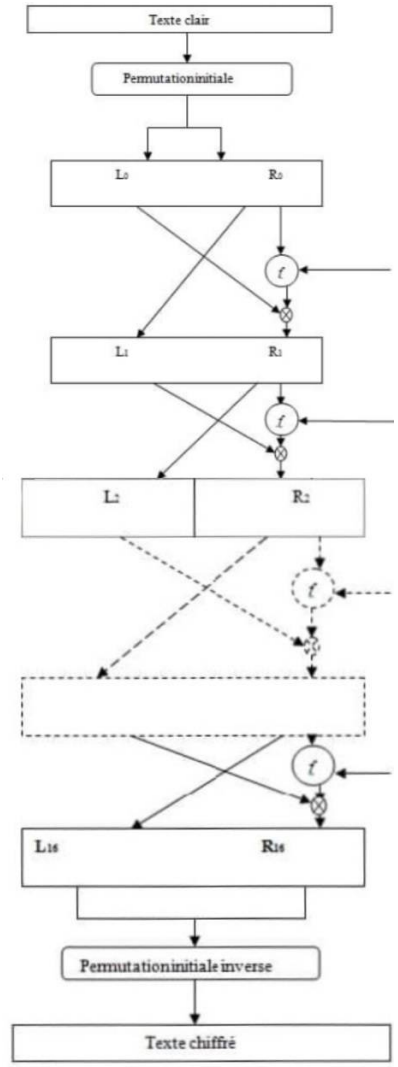

FIGURE 1.12 – Schéma de DES [10]

1.2 Performances de DES

— Pour casser le DES, il faut $2^{56} = 7,2 * 10^{16}$ clés possibles pour trouver la clé.

— Une longueur de 128 bits augmenterait considérablement la résistance du DES.

— La sécurité du DES dépens de la sécurité des clés utilisées [19].

2. **TripleDES** venu pour améliorer les performances de DES, car la faiblesse de DES est sa courte longueur de clé, 3DES combine plusieurs chiffrements DES utilisant 2 ou 3 clés différentes pour obtenir un système de chiffrement global avec des clés longues de 112 bits et 168 bits, triple DES généralement utilisé avec deux clés différentes, le schéma d'utilisation standard est à utiliser en mode EDE (Encryption, Décryptions, Encryption), ce qui le rend compatible avec DES lorsque la même clé est utilisée 3 fois, par conséquent, la clé triple DES se compose de deux DES et d'une longueur de 112 bits, ce qui rend 3DES impossible pour les attaques exhaustives, nous pouvons définir une variante avec 3 clés différentes et être toujours vulnérables aux attaques basées sur l'un des messages intermédiaires [19].

3. **AES** est le sigle d'Advanced Encryption Standard (en français, standard de chiffrement avancé), c'est l'algorithme Rijndael, du nom de leurs concepteurs Belges Joan Daemen et Vincent Rijmen, il a été retenu par le NIST en octobre 2000 pour être l'algorithme AES, le nouveau standard de chiffrement pour les organisations du gouvernement des États-Unis, et ce, principalement pour des raisons de sécurité, performance, efficacité, facilité d'implémentation et flexibilité, de plus son utilisation est très pratique car il consomme peu d'espace de mémoire [8].
   Contrairement à la plupart des chiffrements asymétriques dont la sécurité repose sur des problèmes mathématiques difficiles (tels que le logarithme discret dans ECC ou la factorisation d'entiers dans RSA), la force de l'AES vient de la combinaison de la permutation et du remplacement, généralement appelé réseau de remplacement-permutation (SPN), nous pouvons dire que AES lui-même est un problème difficile, car de nombreux techniciens ont essayé de déchiffrer le cryptage AES et ont échoué [17]. AES est un chiffrement symétrique par bloc prend une taille de bloc de texte en clair de 128 bits ou 16 octets, la longueur de clé peut être de 128, 192 ou 256 bits [20].

   3.1 Détails techniques :
   L'AES opère sur des blocs de 128 bits (Plaintext P) qu'il transforme en blocs cryptés de 128 bits par une séquence de nombre opérations ou 'rounds', à partir d'une clé de 128, 192 ou 256 bits, suivant la taille de celle-ci, le nombre de rounds diffère : respectivement 10, 12 et 14 rounds [17] [21].
   le schéma suivant qui représenté par la figure 1.13 décrit brièvement le déroulement du chiffrement :

   — BYTE-SUB (Byte Substitution) est une fonction non-linéaire opérant indépendamment sur chaque bloc à partir d'une table dite de substitution.

   — SHIFT-ROW est une fonction opérant des décalages (typiquement elle prend l'entrée en 4 morceaux de 4 octets et opère des décalages vers la gauche de 0,

1, 2 et 3 octets pour les morceaux 1, 2, 3 et 4 respectivement).

— MIX-COL est une fonction qui transforme chaque octet d'entrée en une combinaison linéaire d'octets d'entrée et qui peut être exprimée mathématiquement par un produit matriciel sur le corps de Galois (2^8).

— Le + entouré d'un cercle désigne l'opération de OU exclusif (XOR).
K_i est la ième sous-clé calculée par un algorithme à partir de la clé principale K.

— Le déchiffrement consiste à appliquer les opérations inverses, dans l'ordre inverse et avec des sous-clés également dans l'ordre inverse.

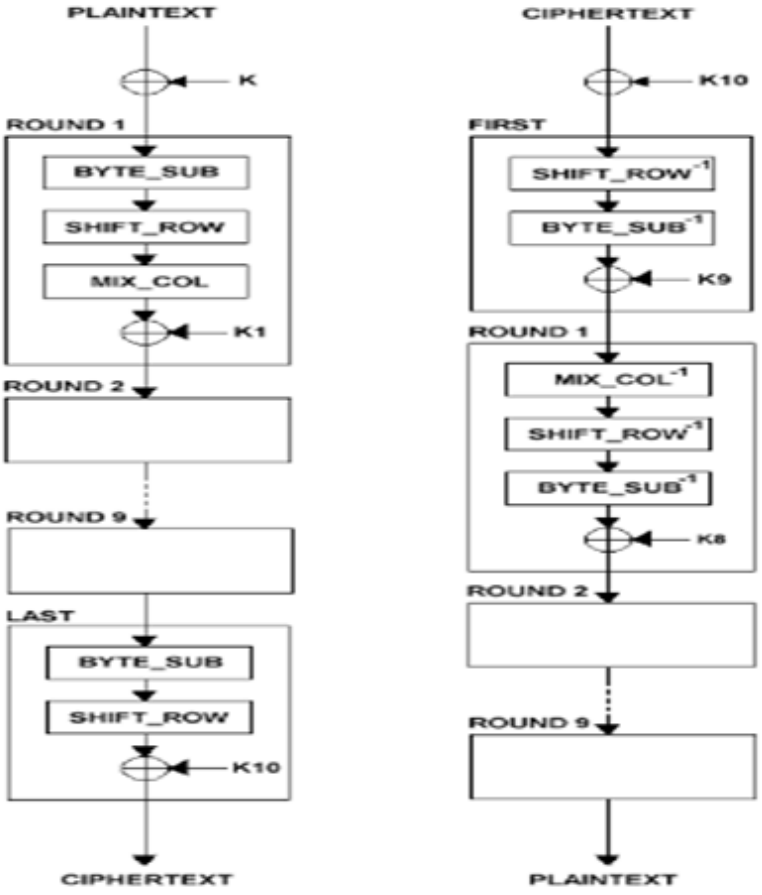

FIGURE 1.13 – Déroulement du chiffrement AES [21]

3.2 Avantages de l'AES

Au-delà du haut niveau de sécurité garanti, l'AES présente d'autres avantages : [19]

— Rapidité de traitement et haute performance, le système AES peut chiffrer et déchiffrer rapidement une grande quantité de données.

— Faibles besoins en ressources et en mémoire, l'algorithme AES peut donc être utilisé pour des appareils grand public tels que des ordinateurs portables.

— Large possibilité d'implémentation sur matériel (hardware) ou sur logiciel (software).

4. **RC4** a été conçu en 1987 par Ron Rivest, il fait partie des algorithmes dits de chiffrement en continu, il est basé sur des permutations aléatoires, avec des opérations sur des octets, le RC4 est employé dans la sécurité des réseaux sans fil (WEP dans le WIFI) ainsi que dans plusieurs applications commerciales, par exemple dans le protocole SSL et dans Oracle Secure SQL, il s'exécute très rapidement dans les logiciels.
   L'algorithme RC4 a une longueur de clé variable (de 1 à 256 octets), la clé est utilisée pour initialiser une "table d'états" de 256 octets, la table d'état est employée pour la génération d'octets pseudo-aléatoires et ensuite pour produire le flux pseudo-aléatoire avec lequel le texte clair sera transformé avec l'opération de l'OU-Exclusif. Pour le chiffrement deux étapes sont nécessaires : l'initialisation à l'aide de la clé et le chiffrement du texte clair [19].

   — La première étape génère deux tableaux de 256 octets en fonction de la clé : un tableau K initialisé avec les octets de la clé et un tableau P (appelé table d'états, laquelle sera le flux appliqué sur le texte clair) initialisé avec les nombres de 0 à 255 permutés pseudo-aléatoirement selon le tableau K.
   Exemple des tableaux (clé de 40 bits) :
   $K = clé(0), clé(1), clé(2), .., clé(39), clé(0), clé(1), .., clé(39)$
   et $P = 34, 55, 228, 0, .., 4$

   — La deuxième étape consiste à des permutations pour effectuer le chiffrement, à noter que les additions sont toutes exécutées modulo 256, le chiffrement est relativement simple.
   le RC4 est populaire en étant extrêmement rapide (environ dix fois plus rapide que le DES), et relativement sûr [19].

1.6.1.3 Propriétés de la cryptographie symétrique

— La clé de transmission doit être envoyée via un canal sécurisé.

— Dans certaines méthodes de chiffrement, un espion peut appliquer une méthode dite force brute qui consiste à essayer toutes les clés jusqu'à l'obtention d'un message ayant un sens.

— La même clé est utilisée pour le chiffrement et le déchiffrement.

— La difficulté de générer des clés aléatoirement réellement [22].

1.6.1.4 Avantages et les inconvénients de la cryptographie symétrique

Dans ce tableau 1.1 on a résumé quelques avantages et inconvénients de la cryptographie symétrique [22].

Cryptograohie	Avantages	Inconvénient
Symétrique	• Système rapide de chiffrement et déchiffrement. • Clés relativement courtes (128 ou 256 bits). • Primitive de mécanismes cryptographiques, et bonne performances et sécurité bien étudié assure la confidentialité des données.	• Gestion des clés difficiles (nombreux clés). • Point faible : l'échange de la clé secrète. • Dans un réseau de N entités susceptibles de communiquer secrètement il faut distribuer $N * (N-1)/2$ clés.

TABLE 1.1 – Avantages et inconvénients de cryptographie symétrique

.

1.6.2 Cryptographie asymétrique

En 1976, Whitfield Diffie et Martin E. Hellman décrit la possibilité de développer un algorithme de chiffrement basé sur deux clés différentes [18].
Dans un système asymétrique, le récepteur génère une paire de clés asymétrique : une clé publique diffusée à tout le monde et une clé privée gardée secrète chez le récepteur, la particularité de cette paire de clés est que tout message chiffré avec la clé publique ne peut être déchiffré qu'avec la clé privée correspondante, par conséquent la confidentialité du message crypté à l'aide de la clé publique du destinataire (figure 1.14), c'est le mécanisme utilisé par la signature numérique pour authentifier l'auteur d'un message [10].

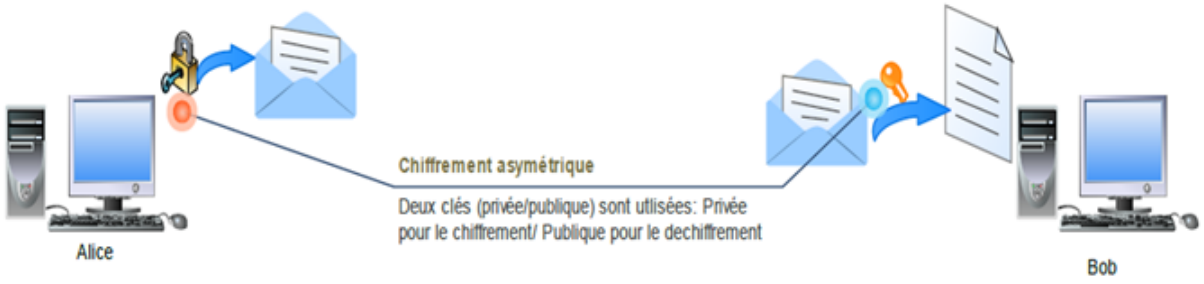

FIGURE 1.14 – Chiffrement asymétrique

1.6.2.1 Principe de fonctionnement

Le fonctionnement de la cryptographie asymétrique est illustré sur la figure 1.14 tel que : [17] [18]

1. L'entité A(Alice) génère une paire de clés l'une publique notée P(A), l'autre privée notée S(A).
2. L'entité B(Bob) désire communiquer avec A, donc elle récupère la clé publique de P(A).
3. B(Bob) chiffre le message avec la clé publique P(A), et envoie le message chiffré à A.

4. A déchiffre le message reçu en utilisant sa clé privée P(S).

Les algorithmes de chiffrement asymétrique se basent sur des concepts mathématiques tels que l'exponentiation de grands nombres premiers (RSA), le problème des logarithmes discrets (El Gamal, courbe elliptique), ou encore le problème du sac à dos (Diffie-Hellman).

1.6.2.2 Algorithmes les plus connus dans la cryptographie asymétrique

1. **RSA** L'algorithme à clé publique RSA (Rivest, Shamir et Adleman) a été inventé en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman, sa sécurité réside dans la difficulté à factoriser le produit de deux grands nombres premiers [8] [18].
   RSA est fondé sur la difficulté de factoriser des grands nombres qui sont le produit de deux grands nombres premiers. graphiquement, on peut dire que multiplier deux grands nombres premiers est une fonction à sens unique : il est facile de multiplier deux nombres pour obtenir un produit, mais il est difficile d'analyser ce produit et de trouver deux grands nombres premiers [10].

   1.1 Initialisation

   — Choisir deux nombres premiers, p et q, les deux étant plus grands que 10^{100}.

   — Calculer $n = p * q$ (n est le module)

   — Choisir e aléatoire tel que e et $((p-1)*(q-1))$ n'aient aucun facteur commun excepté 1.

   — Trouver d tel que : $e * d = 1 \quad Mod \quad ((p-1)(q-1))$.

   — Clé publique : (n, e).

   — Clé privée : (n, d) ou (p, q, d) si on désire garder p et q.

   1.2 Chiffrement/Déchiffrement

   — L'expéditeur crée le texte chiffré c à partir du message m : $c = m^e \quad mod(n)$, où (n, e) est la clé publique du destinataire

   — Le destinataire reçoit c et effectue le déchiffrement : $m = c^d \quad mod(n)$, où (n, d) est la clé privée du destinataire.

   1.3 Sécurité du système

   Pour craquer le code, il faut trouver d (exposant privé) à partir de n et e (clé publique) [3] [11].

   — Eve sait que $d * e \quad mod \quad m = 1$

   — Pour résoudre cette équation, vous devez connaître m ...

   — ... Autrement dit, déterminer les nombres premiers p et q tels que $pq = n$ (parce que $m = (p-1)(q-1)$)

   — Trouvez donc la factorisation de n en deux nombres premiers p et q.

   — Factorisation d'entiers (très grands) en facteurs premiers très difficiles, car cette opération doit pouvoir Calcul très important.

   — Il est recommandé d'utiliser régulièrement des tailles de clé de plus en plus grande (actuellement de 2048 bits).

2. **Diffie Hellman** Le plus simple pour comprendre El Gamal, c'est encore de partir de l'échange de clé Diffie-Hellman, voici les étapes du protocole : [23]
   - Alice et Bob choisissent un nombre premier n et un nombre g tels que : $1 \leq g \leq n-1$.
   - Alice choisit un nombre secret a et calcule $X = g^a \bmod n$ et Bob fait la même chose avec le nombre secret b calcule $Y = g^b \bmod n$.
   - Alice envoie le nombre X à Bob et il lui envoie le nombre Y.
   - Les deux calculent le nombre k de sorte que : Alice calcule $k = Y^X \bmod n$ et Bob calcul $k = X^Y \bmod n$.
   - La clé publique est (g,n) et la clé secrète est (a pour Alice) et (b pour Bob).
3. **El Gamal** est un algorithme de cryptographie asymétrique basé sur les logarithmes discrets, il a été créé par Taher Elgamal, cet algorithme est utilisé par le logiciel libre GNU Privacy Guard, de récentes versions de PGP, et d'autres systèmes de chiffrement, et n'a jamais été sous la protection d'un brevet contrairement à RSA, il peut être utilisé pour le chiffrement et la signature électronique, l'algorithme DSA du NIST est basé sur El Gamal, voici les étapes du protocole : [23]
   - Choisir un grand nombre premier p et deux nombres a et g tels que a < p et g < p .
   - Calculer $A = g^a \bmod p$, la clé public (A, g, p) et la clé privé (a).
   - Pour chiffrer le message claire M choisir un nombre aléatoire k $\in Z_{p-1}$.
   - Calcule : $C1 = g^k \bmod p$ et $C2 = M * A^b \bmod p$, le message chiffré est le couple (C1,C2).
   - Pour déchiffrer le message calcule :$C1 * C2^{p-a-1} \bmod p$.
4. **ECC** est un sujet très populaire en mathématiques, c'est à la fois très compliqué et très riche, ils sont à l'origine de nouveaux algorithmes cryptographiques très sécurisés, la théorie des courbes elliptiques est très ancienne, mais son utilisation en cryptographie date de 1985 qui a été proposée par Neal Koblitz et Victor S. Miller, l'idée était de transformer les cryptosystèmes utilisant des groupes finis du type F_p en cryptosystèmes utilisant l'arithmétique des courbes elliptiques, l'opération de base est la multiplication de points d'une courbe avec de grands nombres entiers : k_P, qui revient en effet à une addition du point P, k fois , la cryptographie par courbe elliptique pose l'épineux problème de la résolution du logarithme discret, la sécurité de cette méthode cryptographique réside dans le choix de la courbe elliptique et de la difficulté que celle-ci pose à résoudre le problème du logarithme discret [24].
   En cryptographie, les courbes elliptiques peuvent être utilisées pour des opérations asymétriques, telles que l'échange de clés ou le cryptage asymétrique sur des canaux non sécurisés, les clés utilisées pour les chiffrements à courbe elliptique sont plus courtes que les systèmes basés sur des problèmes de décomposition, tels que la cryptographie à clé publique RSA (clés de 160 bits lorsque RSA utilise une clé de 1024 bits, à un niveau de sécurité équivalent), ce qui signifie un calcul plus rapide et une puissance inférieure, consommation ainsi que des économies de mémoire et de bande passante, de plus, ECC offre un niveau de sécurité égal ou supérieur à celui des autres méthodes [24], dans le prochain chapitre, nous allons bien expliquer la cryptographie à base des courbes elliptique.

1.6.2.3 Fonction de hachage

Une fonction de hachage est une fonction qui prend en entrée une chaîne de longueur arbitraire finie et qui retourne un haché de longueur fixe (figure 1.15), les fonctions de hachage classiques le haché est compris entre 128 et 512 bits, Soit h une fonction de hachage : [2] [14]
$h : \{0,1\}* \rightarrow \{0,1\}^m$ où m est la taille du haché et $\{0,1\}*$ signifie ensemble des chaînes binaires de taille quelconque. Les fonctions de hachage ne sont pas des systèmes de chiffrement puisqu'elles ne nécessitent pas l'utilisation d'une clé. et pourtant, on les classe naturellement dans la famille des algorithmes symétriques car la plupart du temps, elles sont construites à partir de primitives dérivées de systèmes de chiffrement par bloc ou de systèmes de chiffrement à flot [14].
Les propriétés de ces fonctions de hachage sont les suivantes :

— La fonction de hachage doit être telle qu'elle associe un et un seul hachage au texte en clair (Cela signifie que la moindre modification d'un document entraînera une modification de sa valeur de hachage).

— Un résultat sur un nombre limité d'octets.

— L'impossibilité de retrouver le message original à partir de condensé (sens unique). Les algorithmes de hachage les plus populaires sont : MD5 (condensé de message) et SHA.

FIGURE 1.15 – Principe des fonctions de hachage [17]

1.6.2.4 Avantages et les inconvénients de cryptographie asymétrique

Dans ce tableau 1.2 on a résumé quelques avantages et inconvénients de la cryptographie asymétrique [22].

1.6.3 Cryptographie hybride

La cryptographie hybride est un système de cryptographie impliquant les deux grandes familles de systèmes cryptographiques : la cryptographie asymétrique et la cryptographie symétrique, elle a basé sur des logiciels tels que PGP et GnuPG s'appuient sur ce concept qui permet de combiner les avantages des deux systèmes [25].

Cryptographie	Avantages	Inconvénient
Asymétrique	• Asymétrique pas de secrète à transmettre. • Nombre clés à distribuer est réduit par rapport aux clés symétriques. • Très utile pour échanger des messages. • La distribution est simplifiée :La clé privée n'est jamais révélée ou transmise et la clé publique est disponible à tous les utilisateurs.	• lenteur des algorithmes de déchiffrement. • La taille de clés généralement grand • sécurité conditionnel publiques. • Lenteur de calcul pas d'authentification de la source.

TABLE 1.2 – Avantages et inconvénients de cryptographie asymétrique.

1.6.3.1 **GnuPG GNU (Privacy Guard)** est un logiciel de cryptage hybride qui vous permet de crypter et de signer des données et des communications, il dispose d'un système général de gestion des clés et de modules d'accès pour divers répertoires de clés publiques, GnuPG également connu sous le nom de GPG, est un outil en ligne de commande avec une intégration facile avec d'autres applications, il existe un grand nombre d'applications et de bibliothèques frontend disponibles. GnuPG prend en charge les algorithmes suivants : [17]
Algorithmes asymétrique : RSA, El Gamal, DSA.
Algorithmes symétrique : 3DES, IDEA, CAST5, Blowfish, AES, Camellia.

1.6.3.2 **PGP** offre des services d'authentification, confidentialité, compression et de segmentation, tout en étant compatible avec de nombreux systèmes de messagerie.
Le système de cryptage hybride PGP crypte le texte à l'aide d'un cryptage à clé publique et d'un cryptage symétrique, les données sont d'abord compressées, pour réduire le temps de transmission de ces données et économiser de l'espace disque, le plus important est de renforcer la sécurité du cryptage, l'analyseur de chiffrement utilise les modèles de texte en clair pour déchiffrer le chiffrement, alors que la compression réduit ces motifs en texte brut, par conséquent, la résistance à la cryptanalyse sera grandement améliorée. Ensuite le processus de cryptage s'effectue principalement en deux étapes, résumées ci-dessous

— PGP crée une clé secrète IDEA de manière aléatoire, et chiffre les données avec cette clé [25].

— PGP crypte la clé secrète IDEA et la transmet au moyen de la clé RSA publique du destinataire [25].

Les trois chiffrements par bloc symétriques offerts par PGP sont CAST, Triple DES et IDEA, qu'ils sachent que les trois chiffrements fonctionnent sur des blocs de texte en clair et chiffré de 64 bits, la taille des clés CAST et IDEA atteint 128 bits et celle de DES triple, 168 bits [17].

1.7 Cryptographie future

1.7.1 Cryptographie quantique

La cryptographie quantique consiste à utiliser les propriétés de la physique quantique pour établir des protocoles cryptographiques afin d'atteindre un niveau de sécurité prouvé ou incommensurable en utilisant uniquement des phénomènes classiques (c'est-à-dire non quantiques), un exemple important de cryptographie quantique est la distribution de clé quantique, qui permet de répartir une clé de chiffrement secrète entre deux interlocuteurs distants, tout en assurant la sécurité de la transmission grâce aux lois de la physique quantique et de la théorie de l'information, cette clé peut ensuite être utilisée dans un algorithme de chiffrement symétrique (technique de masquage unique) pour chiffrer et déchiffrer des données confidentielles [27].

1.7.1.1 Deux protocoles fondamentaux de la cryptographie quantique

— Le Protocole **BB84** bidimensionnel, le premier protocole de distribution de clés quantiques, Décrit à l'origine par Bennett et Brassard et appelé BB84, il permet alice envoie une clé aléatoire à bob en encodant les bits secrets suivants la direction de polarisation du photon.

— Le protocole **Ekert91(E91)** en dimension 2, décrit par Arthur Ekert en 1991 et abrégé en Dans E91, Alice et Bob sont autorisés à utiliser la propriété d'intrication quantique pour en mesurant les paires de qubits intriqués, une clé aléatoire est partagée [27].

1.8 Conclusion

Dans ce chapitre, nous avons abordé en détail la notion de système de cryptographiques, nous avons détaillé les différents types d'algorithmes de chiffrement et leur fonctionnement, lorsque nous avons examiné deux catégories principales de méthodes de chiffrement, le chiffrement symétrique et asymétrique, l'utilisation de la clé secrète et de la clé publique, nous avons découvert qu'il existe des problèmes avec les systèmes de cryptographie à clé secrète et à clé publique et qu'ils ont tous deux leurs propres caractéristiques.
la force des algorithmes à clé publique réside dans la distribution des clés, tandis que les algorithmes à clé secrète sont très efficaces en termes de vitesse de chiffrement.
Dans le chapitre suivant, nous expliquerons en détail le chiffrement basé sur les courbes elliptiques.

CHAPITRE 2

LA CRYPTOGRAPHIE À BASE DES COURBES ELLIPTIQUES (ECC).

2.1 Introduction

Au cours des dernières années, un sujet sur la théorie des nombres et la géométrie algébrique appelé courbes elliptiques a trouvé un champ d'application en cryptographie, la cryptographie à courbe elliptique est un mécanisme à clé publique concurrent des autres cryptosystèmes et qui fournit les mêmes fonctions de chiffrement, sa sécurité est basée sur le problème du logarithme discret de la courbe elliptique.
Nous allons présenter dans ce chapitre un rappel sur les groupes, les définitions de base, les règles sur les courbes elliptiques et la cryptographie à base des courbes elliptiques qui se base sur le problème du logarithme discret.

2.2 Généralités

Dans cette section, nous prendrons un aperçu des courbes elliptiques, du concept de groupe abélien et de groupe cyclique, du concept de champs finis premiers et binaires, et des courbes elliptiques spécifiques aux champs limités. enfin, on peut étudier des Chiffrements avec une courbe elliptique.

2.2.1 Groups

En mathématique, un groupe est un couple $(E, *)$ où E est un ensemble et $*$ est une loi de composition interne qui combine deux éléments a et b de E pour obtenir un troisième élément $a * b$, il faut que la loi satisfasse les quatre axiomes ci-dessous [29].

- **Fermeture :** $\forall (a, b) \in E \mid (a * b) \in E$
- **Associativité :** $\forall (a, b) \in E \mid (a * b) * c = a * (b * c)$.

— **Elément neutre :** $\exists e \in E \mid a * e = e * a = a.$

— **Symétrique :** $\forall a \in E, \exists b \in E \mid a * b = b * a = e.$

2.2.1.1 Groupe Abélien

Un groupe abélien, ou un groupe commutatif, est un groupe dont la loi de composition interne est commutative, un ensemble E est un groupe commutatif lorsque : [29]

$$\forall (a, b) \in E \mid a * b = b * a$$

2.2.1.2 Groupe cyclique

Un groupe fini G est cyclique si tous les élément du groupe peut s'exprime sous forme d'une puissance ou d'un multiple d'un élément particulier g, appelé le générateur du groupe, c'est-à-dire $G =< g >= \{g^n \mid n \in \mathbb{Z}*\}$.
Par exemple si $G = \{g^0, g^1, g^2, g^3, g^4, g^5\}$ et $g^6 = g^0$, alors G est un groupe cyclique.
Tout groupe cyclique est abélien car $g^n g^m = g^{n+m} = g^{m+n} = g^m g^n$.
L'ordre d'un élément e dans un groupe cyclique est le nombre entier n positif le plus petit tel que $ne = 0$ (en notation additive) ou $e^n = 1$ (en notation multiplicative).
Reprenons le même groupe G du paragraphe précédent, par exemple l'ordre de l'élément g^2 est 3 car l'élément neutre du groupe est $g^0 = 1$ et $(g^2)^3 = g^6 = 1$ [29].

2.2.2 Corps

Un corps est un ensemble E muni de deux lois de composition, notée respectivement $+$ et $*$, il faut que les deux lois satisfassent les conditions suivantes : [29]

— Le couple $(E, +)$ forme un groupe abélien, il existe un élément neutre, noté 0, tel que

$$\forall a \in E \mid a + 0 = 0 + a = a.$$

— Le couple $(E \setminus \{0\}, *)$ forme aussi un groupe abélien dont l'élément neutre est 1

$$\forall a \in E \mid a * 1 = 1 * a = a.$$

— La multiplication * est distributif pour l'addition, c'est-à-dire

$$\forall (a, b, c) \in E \mid \quad a * (b + c) = a * b + a * c \quad \text{et} \quad (b + c) * a = b * a + c * a.$$

Autrement dit, un corps est un anneau dont les éléments non nuls forment un groupe abélien pour la multiplication.

2.2.2.1 Corps fini

Les courbes elliptiques les plus couramment utilisées sont des courbes elliptiques définies sur un corps élémentaire de grandes propriétés. ainsi, les opérations fondamentales sur les courbes elliptiques sont les opérations dans le domaine Fp, ces opérations sont très importantes car ils doivent être très efficaces, il est donc bien étudié tant au niveau algorithmique qu'au niveau de l'implémentation logicielle ou matérielle [30].
Un corps fini F est un corps dont le nombre d'éléments est fini, le nombre d'éléments est l'ordre du corps, noté q, qui peut être représenté par la puissance d'un nombre premier $q = p^n$, où p est un nombre premier, appelé la caractéristique du corps, et $n \in \mathbb{Z}+$, pour étudier la cryptographie sur les courbes elliptiques, il faut que nous comprenions les deux types de corps suivants : [29]

1. **Corps premier**
   Un corps est un corps premier, noté Fp lorsque l'ordre du corps $q = p^n$ et p est un nombre premier, le corps est constitué des nombres entiers $\{0, 1, 2, ..., p-1\}$, et $\forall a \in \mathbb{Z}, a \mod p$ donne le reste unique r qui est compris entre $[0, p-1]$ [29].

2. **Corps binaire**
   Un corps fini de l'ordre 2^n est un corps binaire, noté F_{2^n} , qui peut être construit en utilisant une représentation polynomiale, les éléments du corps sont des polynômes binaires dont les coefficients $a_i \in \{0, 1\}$ et les degrés sont inférieurs à n. c'est-à-dire : [29]

$$F_{2^n} = \{a_{n-1}z^{n-1} + a_{n-2} \quad z^{n-2} + ... + a_1 z + a_0 : a_i \in \{0, 1\}\}.$$

2.3 Présentation des courbes elliptiques

Dans cette section, nous montrerons la définition des courbes elliptiques ainsi que l'ensemble des opérations que nous pouvons effectuer sur elles.

2.3.1 Equation de Weierstrass

Un courbe elliptique sur K, définie comme l'ensemble des solutions de l'équation de Weierstrass suivante :

$$Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3$$

Pour simplifier, nous observerons ces équations sous une forme non homogène, c'est-à-dire en posant $x = X/Z$ et $y = Y/Z$.
Dans ce cas, l'équation est de la forme :

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

Toutefois, il faut se rappeler qu'il existe un point de base que nous considérons comme étant l'infini. Si $a_1, ..., a_6 \in K$, , on dit que E est définie sur K. Supposons que $Car(\bar{K}) \neq 2$. Alors, on peut simplifier l'équation en complétant le carré. En effectuant le changement de variable $y \to \frac{(y-a_1x-a_3)}{2}$, on obtient l'équation suivante : [31]

$$y^2 = 4x^3 + b_2x^2 + 2b_4x + b_6$$

2.3.2 Définition d'une courbes elliptique

Les courbes elliptiques ont de nombreuses applications dans des domaines très différents des mathématiques qu'en cryptographie, dans le problème de la factorisation des entiers ou pour fabriquer des codes performants.
La courbe elliptique E est une courbe algébrique, munie entre autres propriétés d'une addition géométrique sur ses points, qui peut être représentée par l'équation Weierstrass :

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

où $a_1, a_2, a_3, a_4, a_6 \in K$ et $\Delta \neq 0$, étant Δ le discriminant de E qui peut être calculé de la manière suivante : [29]

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

La condition de la equation de weierstrass sur les dérivées partielles d'une courbe elliptique est équivalente à la condition que $\Delta \neq 0$ [32].

L'équation peut être grandement simplifiée à cette expression :

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

Le discriminant de la courbe $\Delta = (4a^3 + 27b^2)$, et $\Delta \neq 0$ [29].

La figure 2.1 montre quelques types de courbes elliptiques selon les différents facteurs a et b dans l'équation de courbe [33] .

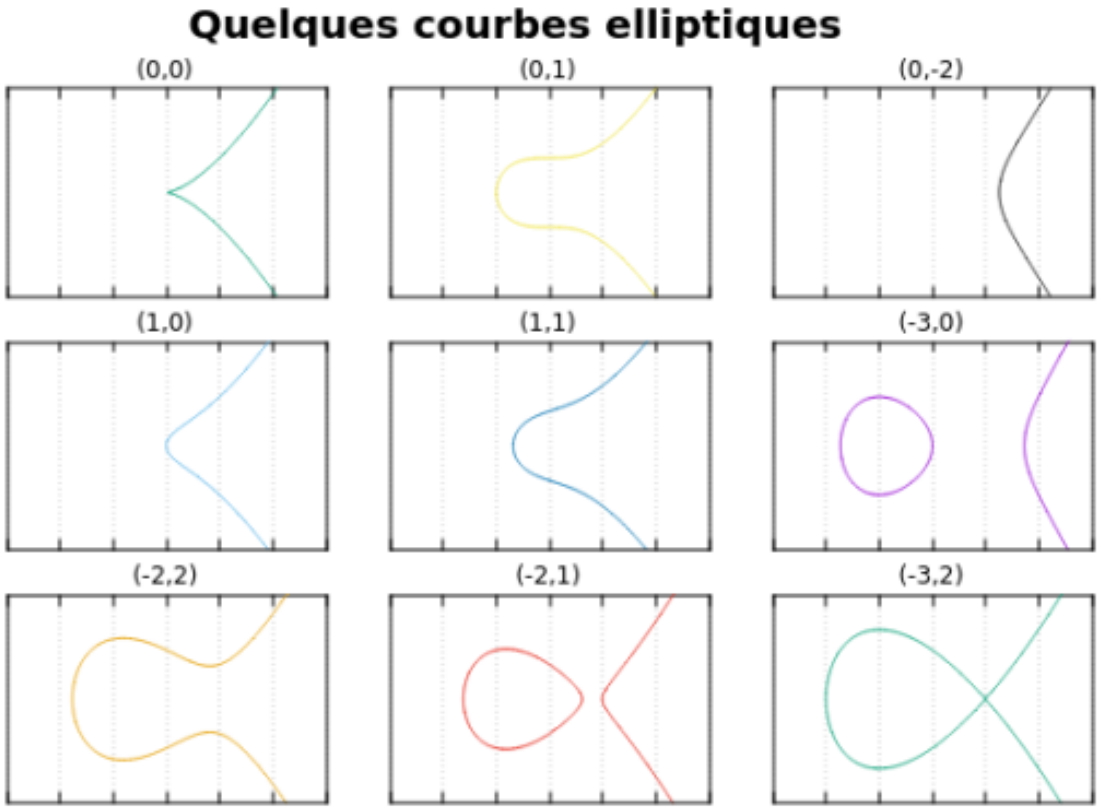

FIGURE 2.1 – Quelque courbe elliptique.

2.3.3 Problème de logarithme discret sur les courbes elliptiques

Soit $H =< P >$ le sous group engendré par P, alors :

$$\forall Q \in H, \exists n \in \mathbb{N} : Q = nP$$

Cet entier n est appelé le logarithme discret de Q en base P et nous le noterons $\log_P Q$.
Pour chiffrer un message via une courbe elliptique, il est nécessaire de calculer l'entier n à partir de les donné public (H, P, Q)
Cet algorithme consistant à calculer $P, 2P, 3P...$ nécessite de faire n additions dans $(E, +)$, ce qui n'est pas efficace [34] [35].
Donc le problème du logarithme discret dans un ensemble est de trouver l'entier n à partir des données globales, la sécurité des protocoles basés sur des courbes elliptiques dépend de la résolution de ce problème.

2.4 Arithmétique sur les courbes elliptiques

Dans cette section, nous verrons des opérations arithmétiques sur des points dans des courbes elliptiques.

2.4.1 Multiplication des points

Soit P un point d'une courbe elliptique, il est possible d'ajouter P à lui-même en réalisant une opération de doublage pour obtenir le point $2 * P$ On peut ensuite calculer $3 * P$, $4 * P$.
C'est la multiplication scalaire : une multiplication entre un scalaire et le point d'une courbe elliptique dont le résultat est un point.
Il est possible de réaliser une multiplication scalaire $k * P$ en ajoutant P à lui-même $k - 1$ fois : $P + P + P + P$... [36].
Le premier algorithme de multiplication scalaire est celui de « double and add », comme on a montré dans la figure 2.2 .

Algorithme 1 Algorithme *double and add* de multiplication scalaire d'un point P sur une courbe elliptique E

Entrées k un scalaire, P un point de E
Sorties: kP
  $Q \leftarrow \mathcal{O}$
  $n = \log_2(k)$
  Pour $i = n - 1$ **à** 0 **Faire**
    $Q \leftarrow 2Q$
    Si $k_i = 1$ **alors**
      $Q \leftarrow Q + P$
    Fin Si
  Fin Pour
  Retourner $Q = kP$

FIGURE 2.2 – Algorithme de multiplication scalaire [32]

2.4.2 Addition des points

L'opération fondamentale sur une courbe elliptique E est l'addition de deux points P et Q (figure 2.3), cette opération va grandement dépendre du système de coordonnées considérées pour représenter un point sur E [32].
Nous travaillons avec des courbes qui sont définies dans un corps premier F_p, les calculs modulaires sont donc indispensables pour limiter les coordonnées des points dans l'intervalle $[0, p - 1]$, Pour $P(x_1, y_1)$, $Q(x_2, y_2)$ deux points distants de la courbe E, nous définissons $P + Q = (x_3, y_3)$, L'unité d'addition de points est utilisée par l'opérateur de multiplication scalaire : [29] [32]
Si $Q \neq P$ et $Q \neq -P$, alors le point $R = (P + Q) = (x_3, y_3)$:

$$x_3 = (\frac{y_2 - y_1}{x_2 - x_1})^2 - x_1 - x_2 \quad (mod \quad p).$$

$$y_3 = (\frac{y_2 - y_1}{x_2 - x_1})(x_1 - x_3) - y_1 \quad (mod \quad p).$$

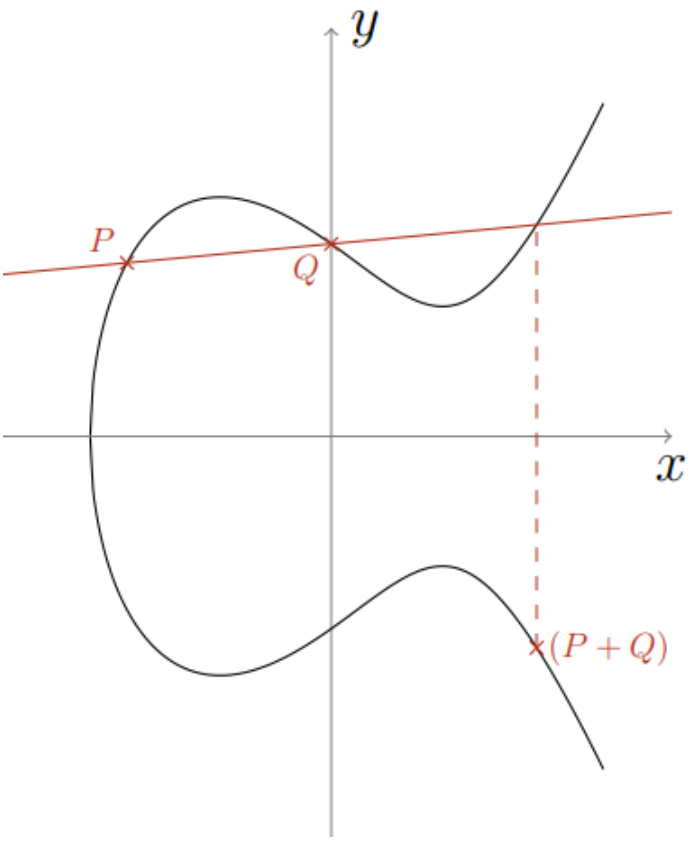

FIGURE 2.3 – Addition de point P et Q sur un courbe elliptique. [32]

2.4.3 Doublement de points

Le calcul de doublement de point est montré dans les formules suivantes : [29]
Si $P = Q$, alors le point $2P = (x_3, y_3)$.

$$x_3 = (\frac{3x_1^2 + a}{2y_1})^2 - 2x_1 \quad (mod \quad p).$$

$$y_3 = (\frac{3x_1^2 + a}{2y_1})(x_1 - x_3) - y_1 \quad (mod \quad p).$$

Si la coordonnée du point P n'est pas nulle, alors la ligne tangente en P coupera la courbe elliptique en exactement un autre point symétrie point 2p, comme le montre la figure 2.4 :

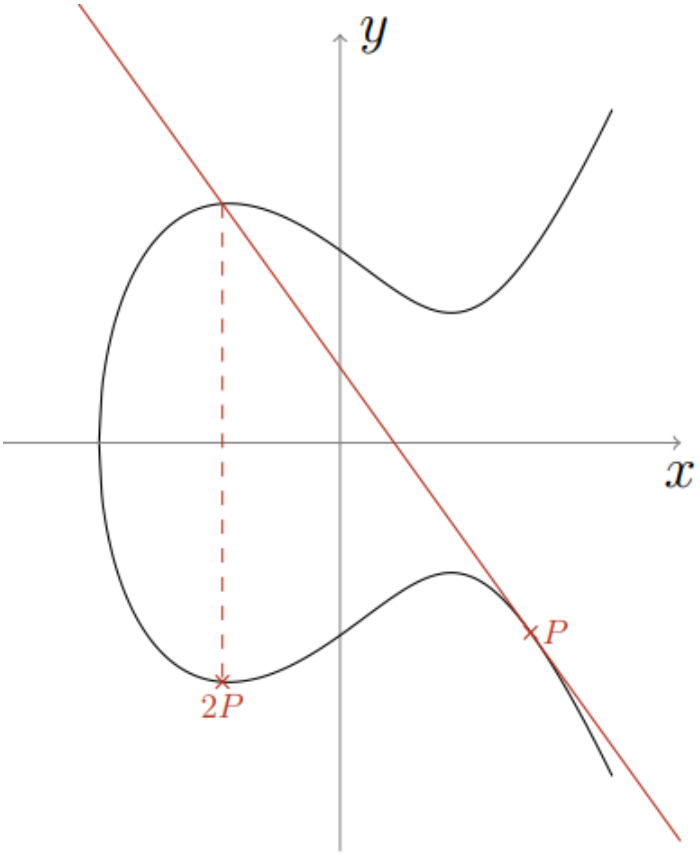

FIGURE 2.4 – Doublement de point sur un courbe elliptique . [32]

2.4.4 Nombre des points sur une courbe elliptique

Il existe une méthode naïve pour observer que le nombre de points d'une courbe elliptique a une borne maximale, pour un x donné il existe au maximum 2 points avec des cordonnées opposées solutions de l'équation (p), nous pouvons donc affirmer qu'il y aura $2p + 1$ points au maximum sur la courbe [36].
Soit une courbe elliptique E définie sur Fp, il existe de nombreuses méthodes pour calculer les points d'une courbe elliptique sur un corps fini qui d'épandent de la caractéristique du corps sur lequel est défini la courbe E [32].

$$|E| = p + 1 - t \quad \& \quad |t| \leq \quad 2 * \sqrt{p}$$

D'autre part, deux éléments suffisent pour engendrer le groupe d'une courbe elliptique et, bien souvent, il est même cyclique.

2.4.5 Cryptographie à base des courbes elliptiques (ECC)

La cryptographie basée sur les courbes elliptiques fait appel à différentes opérations. La figure 2.5 montre que ces opérations peuvent être hiérarchisées et met en avant les liens de dépendance, les protocoles cryptographiques sont basés sur l'exponentiation, dans la suite, je nomme cette opération multiplication scalaire, elle fait appel aux opérations de

doublage et d'addition de points, ces deux opérations basiques sur les points sont des calculs sur leurs coordonnées, qui sont des éléments d'un corps [36].

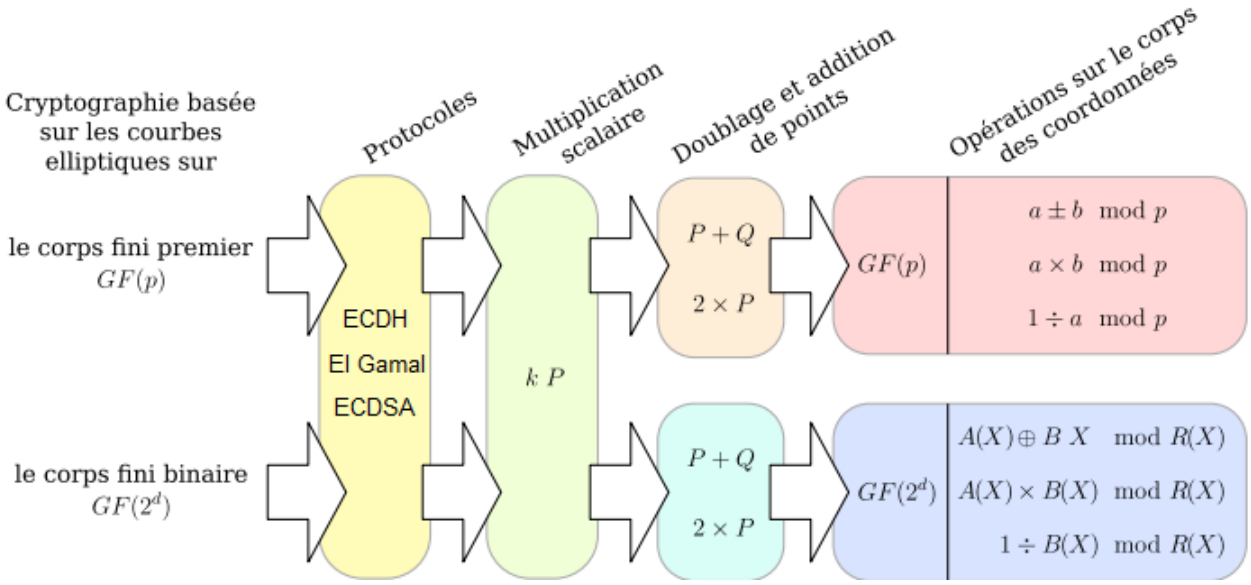

FIGURE 2.5 – Hiérarchie des opérations sur les courbes elliptiques en cryptographie. [36]

2.4.5.1 Génération des clés ECC

Pour génération les clés de ECC, Alice et Bob doivent effectuer les étapes suivantes :

— Alice et Bob choisissent une courbe elliptique et le point P de la courbe (point commun).

— Alice choisit un grand nombre premier k_a (clé privé) et calcule les coordonnées du point Q_a (clé public) tel que $Q_a = k_a P$ et le même processus pour Bob $Q_b = k_b P$.

— Alice et Bob échangent des clés publiques Q_a et Q_b.

2.4.5.2 Chiffrement de ECC

Pour chiffré un message claire M les étapes suivantes doivent être suivies :

— Alice convertir le message clair M en une liste de points à partir de la courbe utilisée.

— Créez un grand nombre entier k et une liste de messages chiffré.

— Ajouter le premier élément de la liste de message chiffré : $P * k$.

— Pour toutes les lettres du message clair on ajoute l'élément $L + Q_b * k$ à la liste de messages chiffrés tel que L c'est le point de chaque letter .

— Envoyez la liste de message chiffré à Bob.

2.4.5.3 Déchiffrement de ECC

Pour déchiffré le message chiffré (la liste des point) les étapes suivantes doivent être suivies :

- Bob calcule le point S tel que S c'est le premier élément du message chiffré * la clé privé b (message chiffré [0] * b).
- Créez une liste de messages déchiffré.
- Pour toutes les point du message chiffré on ajoute l'élément $Lc - S$ à la liste de messages déchiffrés tel que Lc c'est le point chiffré de chaque letter .
  $Lc - S = L + (Q_b * k) - (P * b * k) = L$

2.5 Cryptosystèmes basé sur les courbes elliptiques

L'utilisation d'outils mathématiques tels que les courbes elliptiques à des fins cryptographiques n'a de sens que si elles sont intégrées dans des protocoles sécurisés.
Dans cette section, nous montrons quelques cryptosystème basés sur des courbes elliptiques.

2.5.1 Protocole d'échange de Diffie-Hellman ECDH

La majeure partie de la cryptographie à base des courbes elliptiques d'aujourd'hui repose sur les idées de Diffie-Hellman et utilise le fait que le problème du logarithme discret est réputé difficile sur ces groupes.
L'un des buts principaux de la cryptographie à clé publique est de pouvoir échanger une clé privée qui servira à crypter les messages envoyés par la suite, nous allons décrire le protocole d'échange de clés de Diffie-Hellman adapté aux courbes elliptiques.
Un des moyens pour sécuriser les données transitant entre l'émetteur et le récepteur est qu'ils établissent une clé privée entre eux, la méthode de Diffie-Hellman permet justement de faire cela (figure 2.6, 2.7).

- Alice et Bob choisissent une courbe elliptique E définie sur un corps fini F_q tel que le logarithme discret soit difficile à résoudre, ils choisissent aussi un point $P \in F_q$ tel que le sous-groupe généré par P ait un ordre de grande taille. (E et P sont choisis dont l'ordre soit un grand nombre premier.)
- Alice choisit un nombre entier secret a, calcule $P_a = aP$ et envoie P_a à Bob.
- Bob choisit un nombre entier secret b, calcule $P_b = bP$ et envoie P_b à Alice.
- Alice calcule $aP_b = abP$ et Bob calcule $bP_a = baP$.
- Alice et Bob utilisent une méthode quelconque connue pour extraire une clé secrète de abP, par exemple, ils peuvent utiliser les derniers 256 bits de la première coordonnée de abP comme clé, ou ils peuvent hacher une des coordonnées de abP avec une fonction de hachage pour laquelle ils se sont mis d'accord [37].

L'espion observant le canal de communication pour capter les valeurs ponctuelles de P_a et P_b lors de l'échange, mais s'il veut déduire P_ab , il se heurte au problème du logarithme discret pour extraire a de P_a ou b de P_b .

Algorithm 2 ECDH

Input: A's private key d_A, B's public key P_B
Output: Secret Point S

$S \leftarrow [d_A]P_B$
if $S = \mathcal{O}$ **then**
  return ERROR
return S

FIGURE 2.6 – Algorithme de ECDH [30]

Génération des paramètres publics	
Quelqu'un de confiance choisit et publie un (grand) nombre premier $\boldsymbol{p}$*, une courbe elliptique* $\boldsymbol{E}$ *sur* F_p *et un point* $\boldsymbol{P}$ *dans* $E(F_p)$.	
Calcul privé	
Alice	**Bob**
Elle choisit un entier secret n_A *Elle calcule le point* $Q_A = n_A P$	*Il choisit un entier secret* n_B. *Il calcule le point* $Q_B = n_B P$.
L'échange public des valeurs	
Alice envoie Q_A *à Bob* → Q_A	
Q_B ← *Bob envoie* Q_B *à Alice*	
D'autres calculs privés	
Alice	***Bob***
Elle calcule le point $n_A Q_B$.	*Il calcule le point* $n_B Q_A$.
La valeur du secret partagé est: $n_A Q_B = n_A(n_B P) = n_B(n_A P) = n_B Q_A$.	

FIGURE 2.7 – Échange de clé Diffie-Hellman en utilisant les courbes elliptiques. [38]

2.5.2 Cryptosystème El Gamal sur des courbes elliptiques

C'est un protocole de chiffrement basé sur des courbes elliptiques, sur le problème du logarithme discret, il est facile de créer un analogue direct du cryptosystèmes El Gamal, comme on a présenté dans les deux figues 2.8 et 2.9 les algorithme de encrypte et de décrypte dans ECC-ElGamal, brièvement [38]

— Alice et Bob se mettent d'accord à utiliser un nombre premier p particulier, une courbe elliptique E et un point $P \in E(F_p)$.
— Alice choisit un multiplicateur secret n_A et publie le point $Q_A = n_A P$ comme sa clé publique.
— Le texte en clair de Bob est un point $M \in E(F_p)$. Il choisit un entier k pour être sa

clé éphémère et calcule :

$$C_1 = kP \quad et \quad C_2 = M + kQ_A$$

— Il envoie les deux points (C_1, C_2) à Alice, qui calcule :

$$C_2 - n_A C_1 = (M + kQ_A) - n_A(kP) = M + k(n_A P) - n_A(kP) = M$$

Pour récupérer le texte en clair.

À notre point de vue nous voyons que grâce à la difficulté de résoudre le problème du logarithme discret, Alice et Bob peuvent s'envoyer des messages sans risque.

Algorithm EC-ELGAMAL Encryption

Input: Public Key P, an encoded integer $m \in [0, p-1]$ representing a message
Output: (x_1, y_1, c)

$k \xleftarrow{\mathcal{R}} \{1, \ldots, t-1\}$
$(x_1, y_1) \leftarrow [k]G$
$(x_2, y_2) \leftarrow [k]P$
$c \leftarrow x_2 + m \bmod p$
return (x_1, y_1, c)

FIGURE 2.8 – Algorithme de encrypte ECC-El Gamal [30]

Algorithm EC-ELGAMAL Decryption

Input: Private Key d, an encrypted message (x_1, y_1, c)
Output: an encoded integer $m' \in [0, p-1]$ representing a message

$(x'_2, y'_2) \leftarrow [d](x_1, y_1)$
$m' \leftarrow c - x'_2 \bmod p$
return m'

FIGURE 2.9 – Algorithme de décrypte ECC-El Gamal. [30]

2.5.3 Signature numérique

L'algorithme le plus connu de signature électronique est l'algorithme ECDSA (Elliptic Curve Digital Signature Algorithme), sa sécurité repose sur l'impossibilité de résoudre le logarithme discret dans un sous-groupe d'un courbe elliptique, il a été proposé en premier par Scott Vanstone en réponse à un appel d'offres pour les signatures numériques du National Institute of Standards and Technology (NIST).
C'est un standard ISO depuis 1998, un standard ANSI depuis 1999 et un standard IEEE

et NIST depuis 2000 [40].
Il est important de pouvoir signer un message que l'on envoie c'est-à-dire de pouvoir s'authentifier documenter et assurer son intégrité. pour cela, nous présenterons les étapes de l'algorithme de signature numérique à courbe elliptique (ECDSA).

Nous supposons qu'Alice et Bob utilisent la même courbe elliptique E(Fp) pour sécuriser la communication entre eux, nous supposons que la clé publique d'Alice est $Q_a = k_a * P$ où k_a est sa clé privée et P est le point de générateur de l'ordre n.
Pour signer un message M (figure 2.10), Alice doit suivre les opérations suivantes : [29] [31]

- Choisir un nombre entier aléatoire $k \in [1, n-1]$.
- Calculer $k * P = (x_1, y_1)$ et $r = x_1 \bmod n$, Si r = 0, retourner à l'étape 1.
- Calculer $s = k^{-1}(H(M) + k_a r) \pmod n$ où H est une fonction de hachage, Si $s = 0$ retourner à l'étape 1.
- Envoyer à Bob la signature du message m : le couple (r, s).

Après avoir reçu le message signé, Bob vérifie la signature du message :

- Vérifier si $Q_a \neq \infty$ (point à l'infini) et $Q_a \in E(F_p)$.
- Vérifier si $n * Q_a = \infty$ Car $n * Q_a = n * k_a * P$ et $ord_p(P) = n$.
- Vérifier que (r, s) soient dans l'intervalle $[1, n-1]$.
- Calculer $w = s^{-1} \pmod n$ et $H(M)$.
- Calculer $u1 = H(M)w \pmod n$ et $u2 = rw \pmod n$.
- Calculer $u1 * P + u2 * Q_a = (x_0, y_0)$ et $v = x_0 \pmod n$.
- La signature est acceptée si $r = v$.

Pour vérifions que les signatures authentiques (figure 2.11)seront toujours acceptées : En effet,

$$u1 + u2k_a \equiv H(M)w + rwk_a \equiv w(H(m) + rk_a) \equiv k \pmod n.$$

De plus,

$$(x_0, y_0) = u1P + u2Q_a = u1P + u2k_aP = (u1 + u2k_a)P = kP.$$

Algorithm ECDSA Signature

Input: private key d, an encoded integer $m \in [0, p-1]$ representing a message
Output: Signature (r, s)

1: $k \xleftarrow{\mathcal{R}} \{1, \ldots, t-1\}$
2: $Q \leftarrow [k]G$
3: $r \leftarrow x_Q \bmod t$
4: **if** $r = 0$ **then**
5: **go to** line 1
6: $k_{inv} \leftarrow k^{-1} \bmod t$
7: $s \leftarrow k_{inv}(dr + m) \bmod t$
8: **if** $s = 0$ **then**
9: **go to** line 1
10: **return** (r, s)

FIGURE 2.10 – Algorithme de ECDSA Signature. [30]

Algorithm ECDSA Verification

Input: public key P, an encoded integer $m \in [0, p-1]$ representing a message, signature (r, s)
Output: true or false

$s_{inv} \leftarrow s^{-1} \bmod t$
$u_1 \leftarrow s_{inv} \times m \bmod t$
$u_2 \leftarrow s_{inv} \times r \bmod t$
$Q \leftarrow [u_1]G + [u_2]P$
$v \leftarrow x_Q \bmod t$
if $v = r$ **then**
return true
else
return false

FIGURE 2.11 – Algorithme de ECDSA Vérification. [30]

2.6 Domaines d'utilisation de la cryptographie à base des courbes elliptiques

De nos jours, de nombreux protocoles et plates-formes utilisent un chiffrement elliptique, en raison de sa vitesse ? !, de sa sécurité et de son efficacité, parmi ces protocoles et plate-formes : [41]

— TLS Le protocole de sécurisation d'échange sur internet, utilise la ECC dans la phase "Handshake" pour l'agrément sur la clé à utiliser pour chaque session.
— Smart Cards a côté de RSA, les nouvelles smart-card support la ECC
— Passeport biométrique La ECC est utilisée pour construire la clé de session.
— Bitcoin : ECC est utilisée pour la signature Transport Layer Security électronique pendant les transactions des paiements avec Bitcoin, le courbe utilisé par Bitcoin c'est Secp256k1 (figure 2.12) avec l'équation : $y^2 = x^3 + 7$

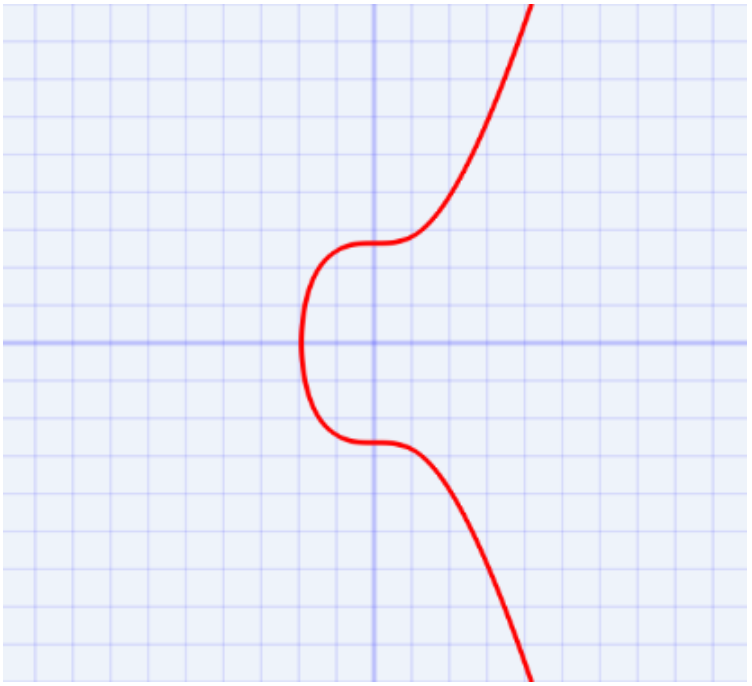

FIGURE 2.12 – Courbe Secp256k1 utilisé par Bitcoin. [42]

2.7 Conclusion

Dans ce chapitre, nous avons introduit la cryptographie elliptique, en plus d'un ensemble de termes mathématiques de base pour comprendre son travail, ses protocoles et son problème de base représenté par le problème du logarithme discret. nous avons également abordé certains domaines qui utilisent la cryptographie sur les courbes elliptiques. Dans le chapitre suivant, nous effectuons une étude analytique comparative sur ECC par rapport à d'autres systèmes de chiffrement connus.

CHAPITRE 3

POSITIONNEMENT D'ECC ENTRE LES SYSTÈMES DE CHIFFREMENT

3.1 Introduction

Dans ce chapitre, nous présentons l'application que nous avons implémentée, à partir de laquelle nous avons obtenu les résultats pour l'implémentation d'algorithmes de chiffrement symétrique (AES, RC4, 3DES) et asymétrique (ECC, RSA, El Gamal, Diffie Hellman), dans toutes les opérations (génération de clé, chiffrement et déchiffrement) et avec différentes tailles de clé, et une comparaison de temps d'exécution de la génération des clés, de chiffrement et de déchiffrement pour ECC et les autres systèmes précédents pour la taille de clé et pour le niveau de sécurité.
Noté que les expériences sont effectuées sur Netbeans 8.0.1 sur un processeur Intel (R)-Core (TM) i5-2520U et 2.50GHz avec 4GB de RAM sous-système de 32 bit et la plateforme Windows 7, L'expérimentation a été menée pour trouver le laps de temps lors du processus de chiffrement et le processus de déchiffrement par ECC.

3.2 L'environnement de développement

Avant de commencer la présentation de notre application, nous allons tout d'abord, citer les outils utilisés lors du développement.

Langage JAVA
Le Java est un langage de programmation orientée objet, en plus d'un langage de programmation, développé par Sun Microsystems en 1995, et racheté depuis par Oracle, le java est basé sur C++, mais avec une approche simplifiée et des fonctionnalités plus avancées, java fournit également une bibliothèque de classes qui regorge de tous les domaines des applications informatiques, telles que les bases de données, les cartes à puce et les téléphones mobiles, java possède des caractéristiques très intéressantes qui en font une plate-forme de développement qui est l'une des innovations les plus intéressantes apparues dans ces dernières années [43].

JDK (JAVA Développement Kit)

Le JDK est le kit de développement JAVA officiel publié par Oracle, intègre tous les outils et bibliothèques nécessaires pour créer des applications et est l'environnement dans lequel le code java est compilé en bytecode (code intermédiaire) afin que la machine virtuelle java puisse l'interpréter [44].

NetBeans IDE

NetBeans est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (Common Développement and Distribution License), en plus de java, netbeans permet également de supporter différents autres langages, comme Python, C, C++, XML et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages web).
Conçu en Java, netbeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Développement Kit JDK est requis pour les développements en java [45].

3.3 Applets de simulation

Définition d'une Applet

Une applet est un programme Java qui s'exécute dans un logiciel de navigation supportant java ou dans l'appletviewer du JDK (Java Développement Kit), le mécanisme d'initialisation d'une applet se fait en deux temps :
1. La machine virtuelle java instancie l'objet applet en utilisant le constructeur par défaut.
2. La machine virtuelle java envoie le message init à l'objet applet.

Les Applets utilisés

Afin de vulgariser le fonctionnement de certains algorithmes, des applets ont été réalisée pour simuler les étapes de fonctionnement et obtenir les résultats de traitements instantanément.
Les algorithmes dont les applets ont été programmées sont :

- Algorithme de l'AES
- Algorithme de RC4
- Algorithme de TripleDES

Les systèmes de chiffrement que nous avons implémenté en Java

- Algorithme de l'ECC
- Algorithme de El Gamal
- Algorithme de Diffie Hellman
- Algorithme de l'RSA

3.4 L'architecture de notre système

Dans la figure 3.1 on a représenté l'architecture de notre application, et les systèmes de chiffrement symétriques et asymétriques que nous avons implémenté.

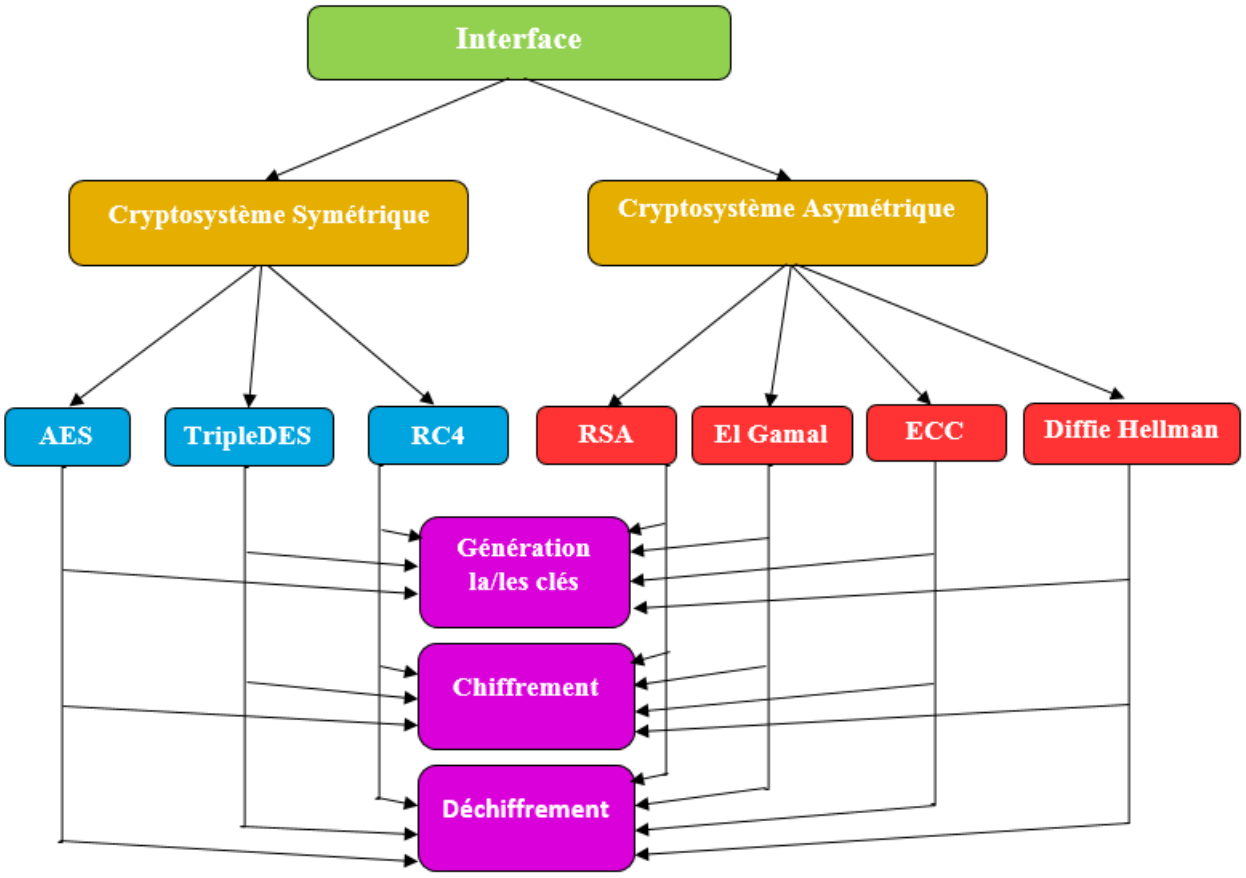

FIGURE 3.1 – L'architecteur de système.

3.5 Quelques interfaces de notre application

— Dans la figure 3.2 on a représenté l'interface d'accueil de notre application de comparaison des système de chiffrement puis on a présenté dans la figure 3.3 les système de chiffrement que nous avons implémenté.

— les figures (3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 3.10) représentent les capture des résultats de notre application que nous avons implémenté pour les différents systèmes de chiffrement.

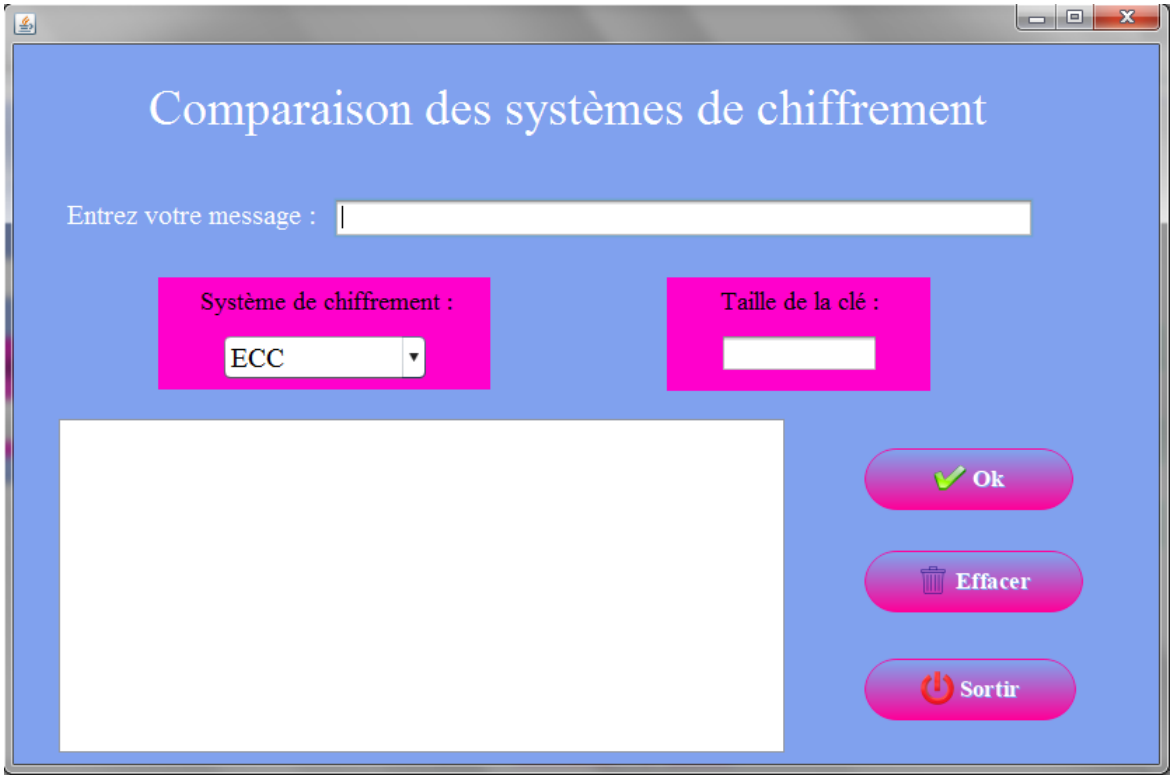

FIGURE 3.2 – Interface d'accueil l'application.

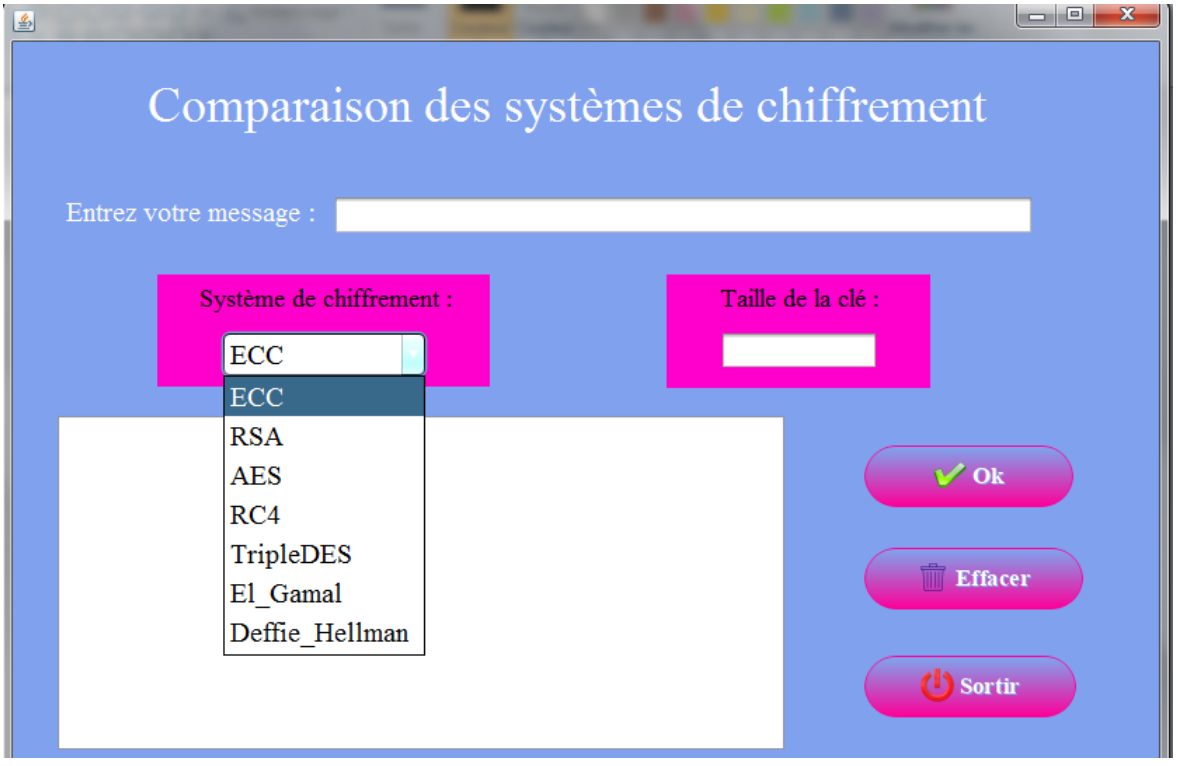

FIGURE 3.3 – Systèmes de chiffrement dans l'application.

3.5.1 Captures d'écran du l'exécution des systèmes

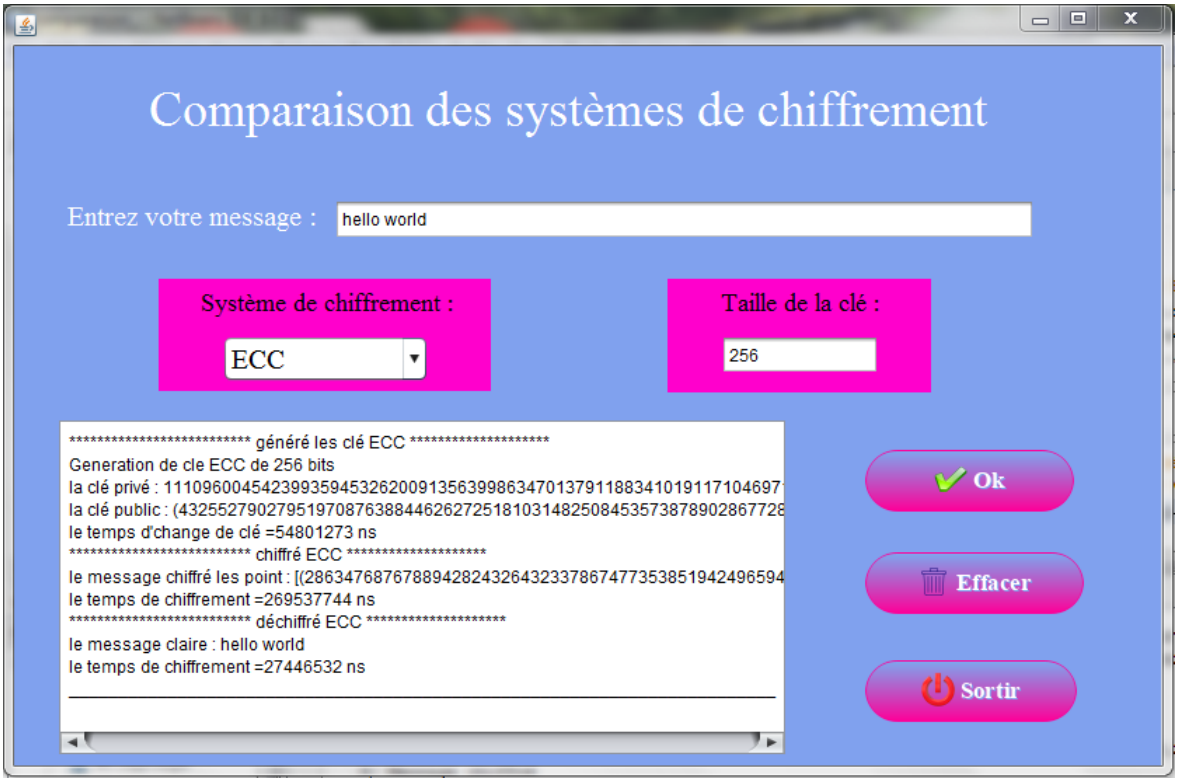

FIGURE 3.4 – Interface de l'application au résultat de l'exécution du système ECC.

FIGURE 3.5 – Interface de l'application au résultat de l'exécution du système RSA.

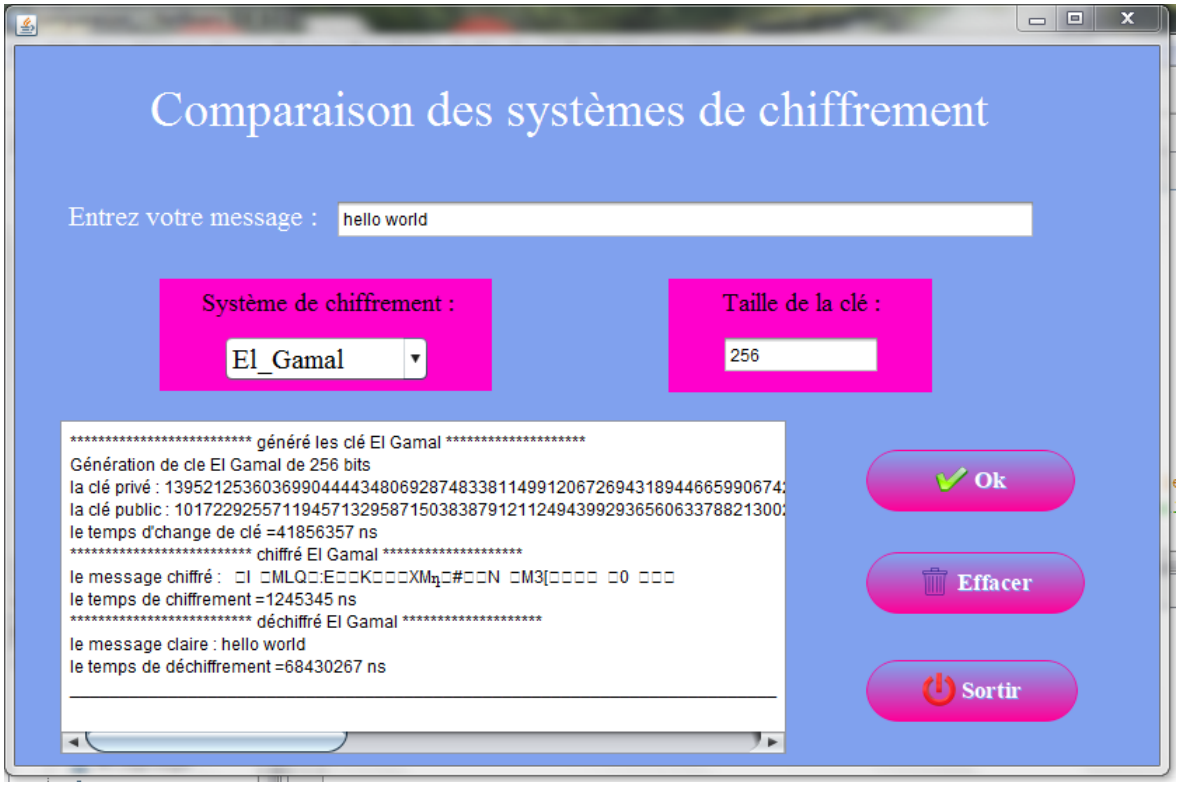

FIGURE 3.6 – Interface de l'application au résultat de l'exécution du système El Gamal.

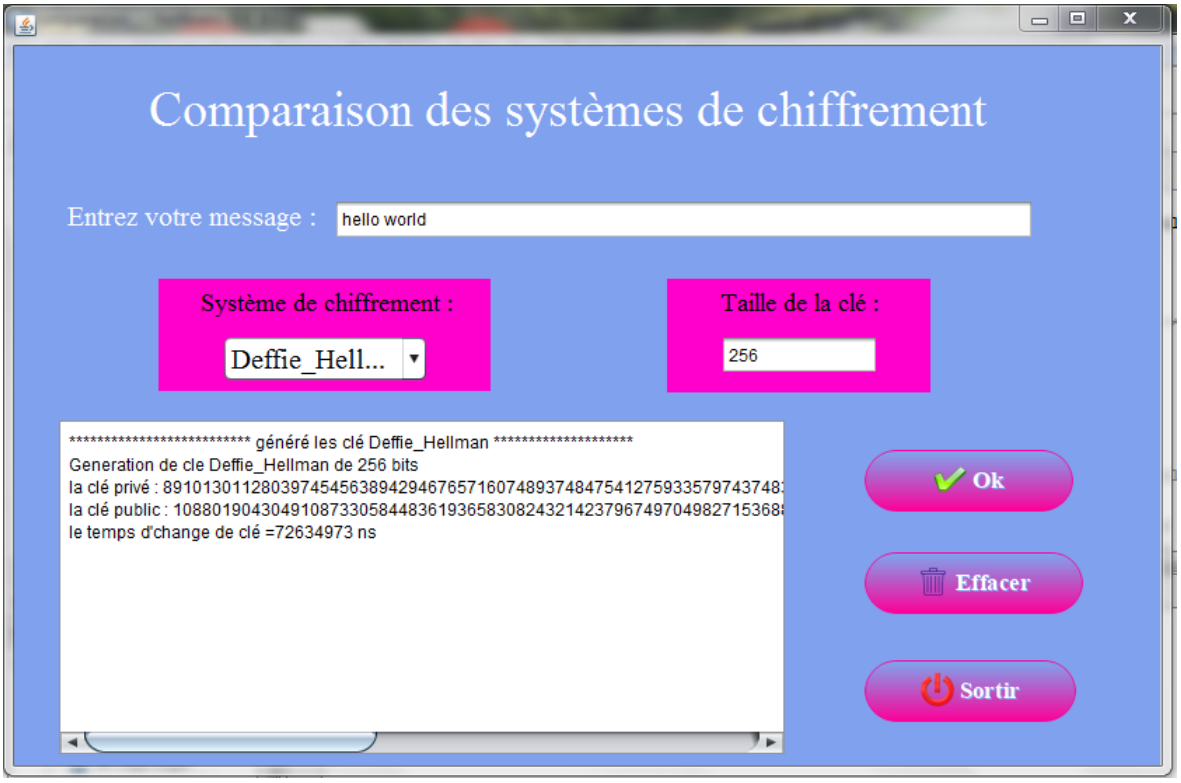

FIGURE 3.7 – Interface de l'application au résultat de l'exécution du système Diffie Hellman.

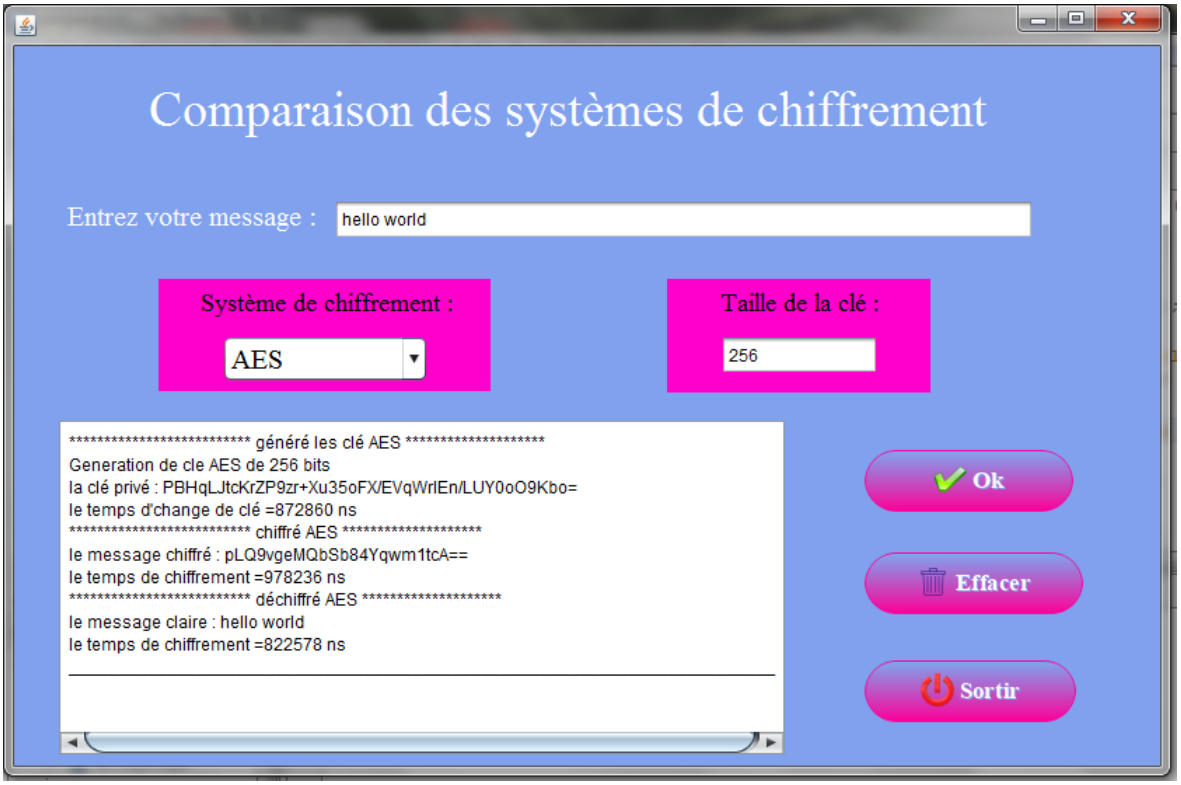

FIGURE 3.8 – Interface de l'application au résultat de l'exécution du système AES.

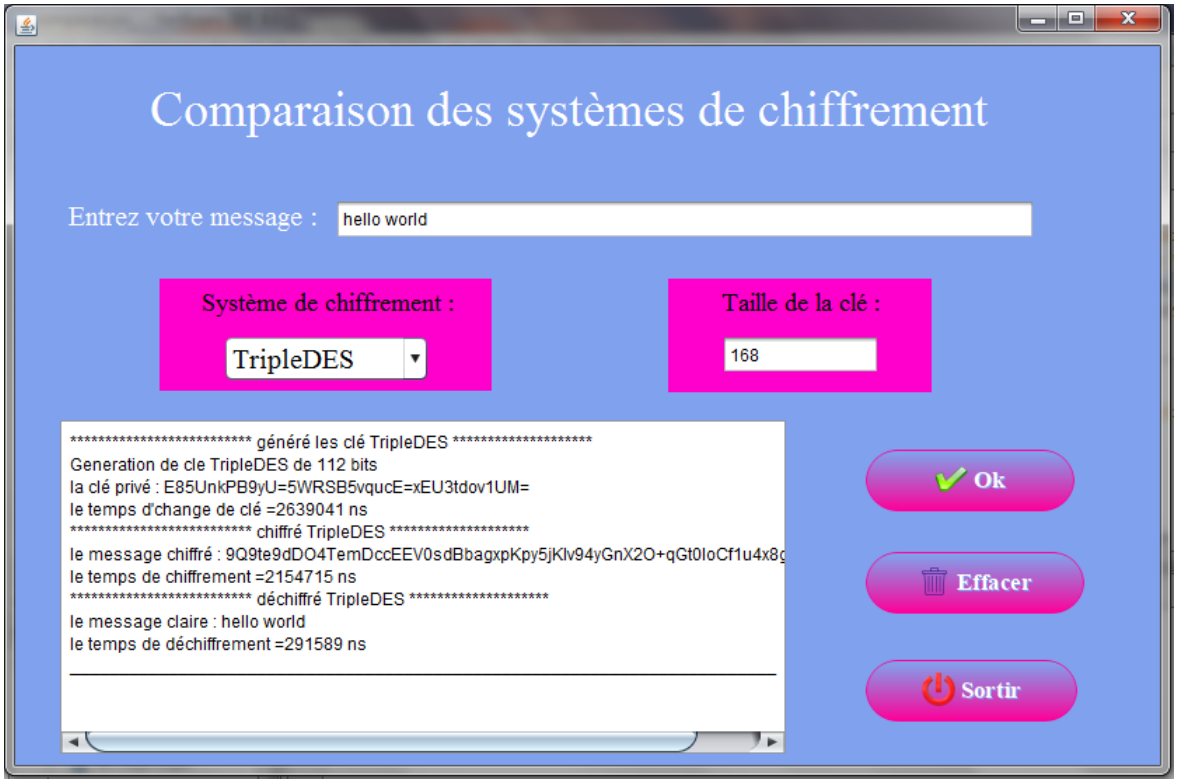

FIGURE 3.9 – Interface de l'application au résultat de l'exécution du système TripleDES.

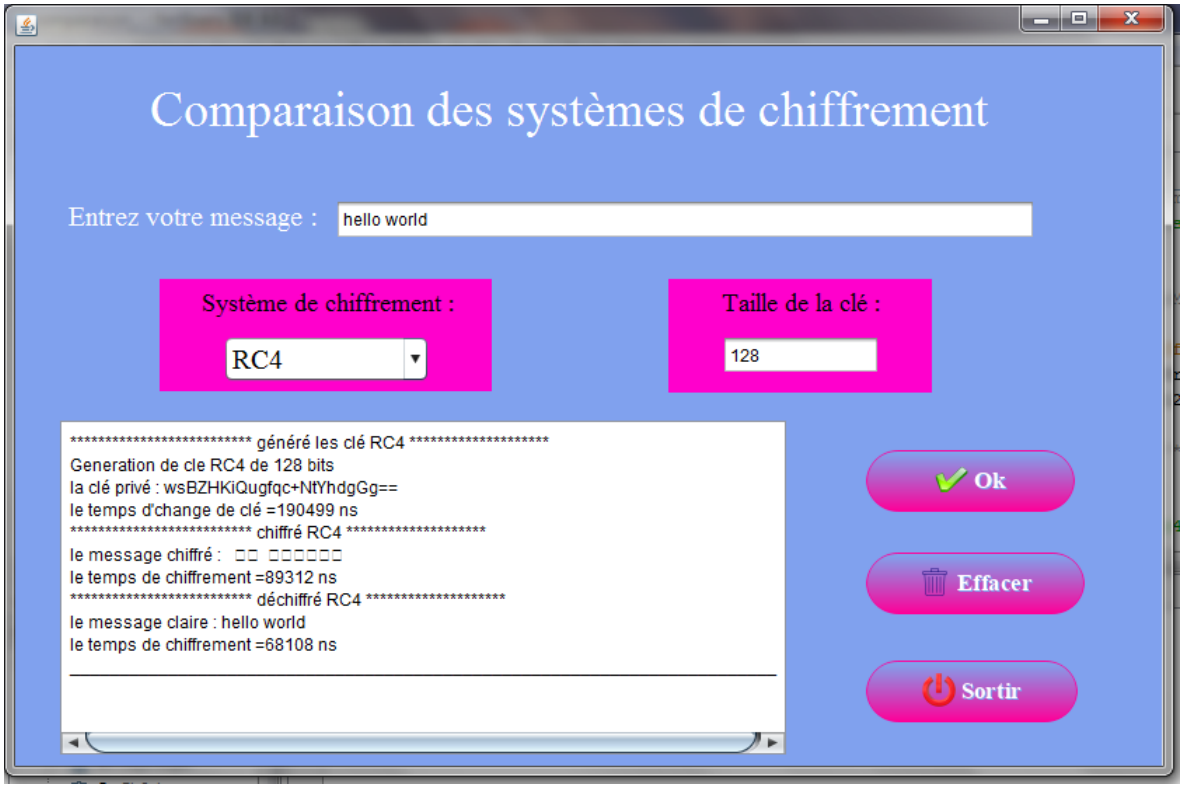

FIGURE 3.10 – Interface de l'application au résultat de l'exécution du système RC4.

3.6 Détails de l'implémentation d'ECC

La comparaison dans ce cas-là se faite en fonction du temps d'exécution sur la taille de clé ainsi que le même niveau de sécurité, nous avons choisir 4 types de courbes parmi les courbes proposées par la recommandation du NIST sur $GF(p) : \{P-160, P-256, P-384, P-521\}$ (les tables(3.1, 3.2, 3.3, 3.4)), ces courbes sont illustrés dans les tableaux suivants ,tel que : [46]

— a et b les coefficients de : $x^3 + ax + b \bmod p$.

— r : Ordre de point de base.

— G(x,y) : le point de base.

Selon la taille de la clé a, b, p, r seront :

Courbe P-160	
a	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 7FFFFFFC
p	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 7FFFFFFC
r	01 00000000 00000000 0001F4C8 F927AED3 CA752257
b	1C97BEFC 54BD7A8B 65ACF89F 81D4D4AD C565FA45
Gx	4A96B568 8EF57328 46646989 68C38BB9 13CBFC82
Gy	23A62855 3168947D 59DCC912 04235137 7AC5FB32

Table 3.1 – Courbe P-160.

Courbe P-256	
a	FFFFFFFF 00000001 00000000 00000000 000000000 FFFFFFFF FFFFFFFF FFFFFFFC
p	FFFFFFFF 00000001 00000000 00000000 000000000 FFFFFFFF FFFFFFFF FFFFFFFF
r	FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6PAAD A7179E84 F3B9CAC2 FC632551
b	5ac635d8 aa3a93e7 b3ebbd55 76988bc 651d06b0 cc53b0f6 3bce3c3e 27d2604b
Gx	6b17d1f2 e12c4247 f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945 d898c296
Gy	4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5

Table 3.2 – Courbe P-256.

Courbe P-384	
a	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFC
p	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFF
r	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF C7634D81 F4372DDF 581AODB2 48BOA77A ECEC196A CCC52973
b	b3312fa7 e23ee7e4 988e056b e3f82d19 181d9c6e fe814112 0314088f 5013875a c656398d 8a2ed19d 2a85c8ed d3ec2aef
Gx	aa87ca22 be8b0537 8eb1c71e f320ad74 6e1d3b62 8ba79b98 59f741e0 82542a38 5502f25d bf55296c 3a545e38 72760ab7
Gy	3617de4a 96262c6f 5d9e98bf 9292dc29 f8f41dbd 289a147c e9da3113 b5f0b8c0 0a60b1ce 1d7e819d 7a43ld7c 90ea0e5f

Table 3.3 – Courbe P-384.

Courbe P-521	
a	01FF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFC
p	01FF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
r	01FF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFA 51868783 BF2F966B 7FCCO148 F7O9A5D0 3BB5C9B8 899C47AE BB6FB71E 91386409
b	051 953eb961 8e1c9a1f 929a21a0 b68540ee a2da725 99b315f3 b8b48991 8ef109e1 56193951 ec7e937b 1652c0bd 3bb1bf07 3573df88 3d2c34f1 ef451fd4 6b503f00
Gx	c6 858e06b7 0404e9cd 9e3ecb66 2395b442 9c648139 053fb521 f828af60 6b4d3dba a14b5e77 efe75928 fe1dc127 a2ffa8de 3348b3c1 856a429b f97e7e31 c2e5bd66
Gy	118 39296a78 9a3bc004 5c8a5fb4 2c7d1bd9 98f54449 579b4468 17afbd17 273e662c 97ee7299 5ef42640 c550b901 3fad0761 353c7086 a272c240 88be9476 9fd16650

Table 3.4 – Courbe P-521.

3.6.1 Temps d'exécution du l'ECC

Nous commençons d'abord par ECC et le temps d'exécution (Génération des clés, le Chiffrement et le Déchiffrement) par rapport aux tailles des clés comme elle est représentée dans le tableau 3.5 et les figures (3.11, 3.12, 3.13) :

Génération clé ECC		Chiffrement ECC		Déchiffrement ECC	
Taille clé	Temps (ms)	Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	160	105	160	10
256	54	256	269	256	27
384	89	384	706	384	65
521	179	521	1564	521	148

TABLE 3.5 – Comparaison des tailles de clés de ECC et du temps d'exécution pour la génération des clés, le chiffrement et le déchiffrement de ECC.

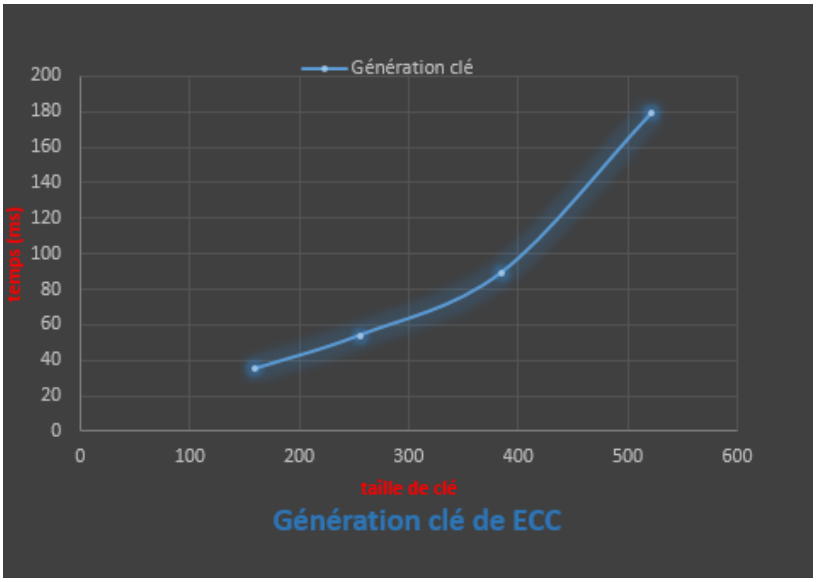

FIGURE 3.11 – Courbe de changement du temps d'exécution (ms) de génération les clés ECC en fonction des tailles des clés.

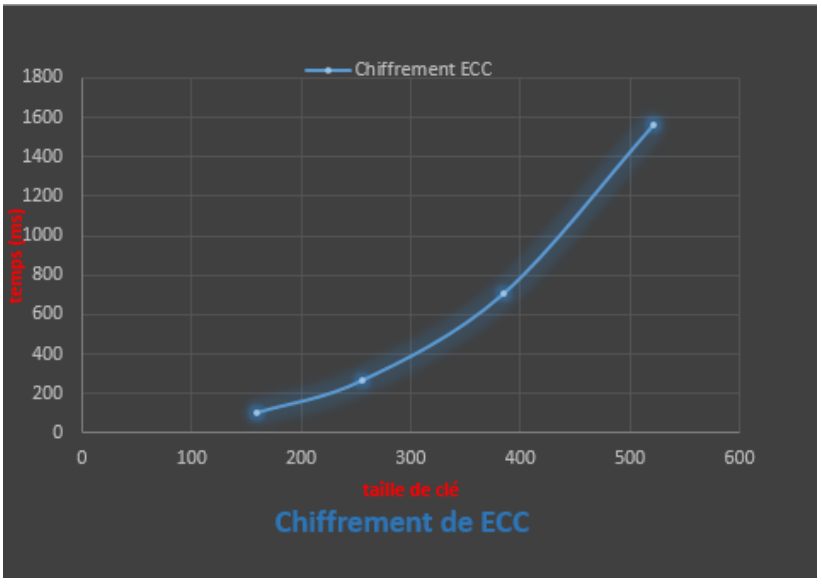

FIGURE 3.12 – Courbe de changement du temps d'exécution (ms) de chiffrement ECC en fonction des tailles des clés.

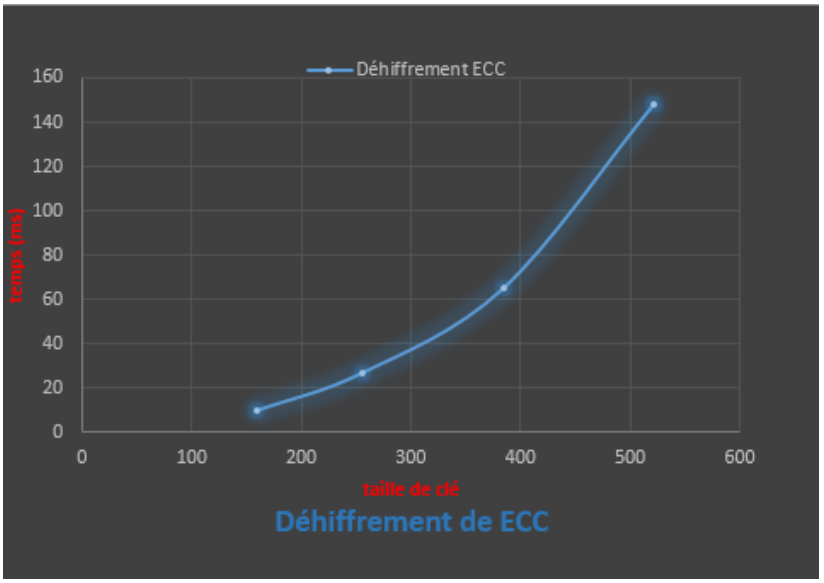

FIGURE 3.13 – Courbe de changement du temps d'exécution (ms) de déchiffrement ECC en fonction des tailles des clés.

Dans le tableau 3.5 et les graphiques représentées sur les figures(3.11, 3.12, 3.13) de ECC, nous remarquons quand la taille de la clé augment, le temps d'exécution augmente aussi de manière significative.

3.7 Expérimentation et analyse des résultats

3.7.1 ECC vs RSA

3.7.1.1 Comparaison de changement du temps d'exécution ECC et RSA sur les tailles de clé

1 Génération des clés ECC vs RSA

Génération clé ECC		Génération clé RSA	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	128	46
256	54	256	130
384	89	512	362
521	179	1024	4344
		2048	57837
		3072	194805

TABLE 3.6 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au RSA.

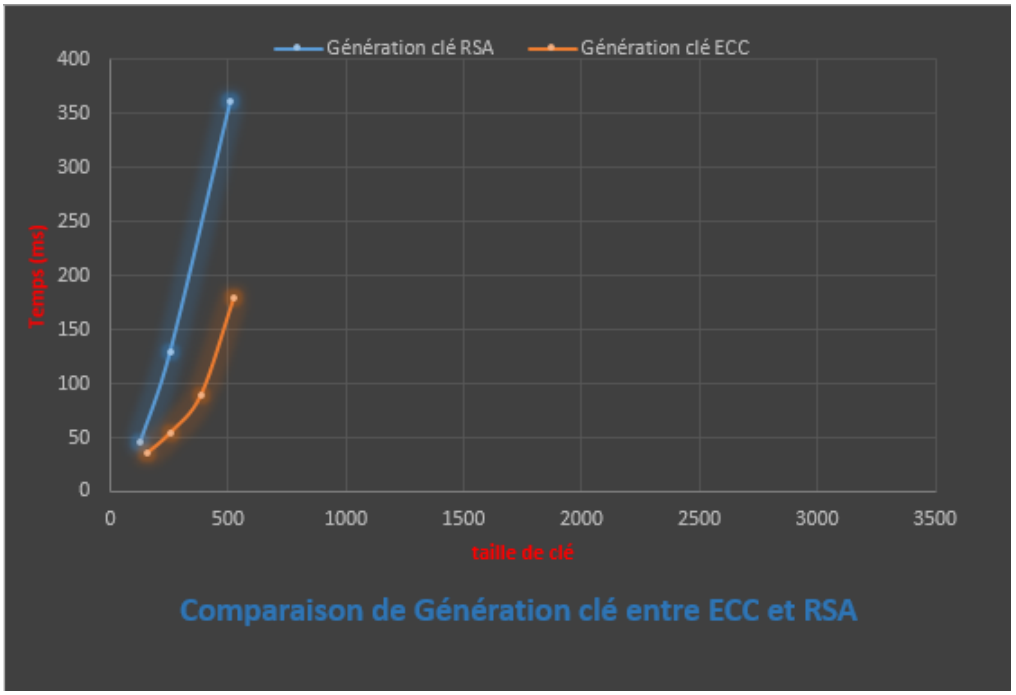

FIGURE 3.14 – Courbe comparative du temps d'exécution de génération de clé entre ECC et RSA.

Dans le tableau 3.6 et le graphique représentée à la figure 3.14 qui montrent le temps d'exécution de ECC et RSA de génération des clés, nous constatons une augmentation du temps d'exécution pour les deux systèmes de chiffrement similaire de la fonction carrée, mais ECC est meilleur en termes des tailles de clé pour RSA.

2 Chiffrement ECC vs RSA

Chiffrement ECC		Chiffrement RSA	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	105	128	0,39
256	269	256	1,2
384	706	512	8
521	1564	1024	63
		2048	489
		3072	1642

TABLE 3.7 – Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au RSA.

Dans le tableau 3.7 et le graphique représentée à la figure 3.15 qui montrent le temps d'exécution de ECC et RSA de chiffrement, nous constatons une augmentation du temps d'exécution pour les deux systèmes, ECC similaire à la fonction carrée, mais RSA est meilleur qu'ECC en terme des tailles de clé .

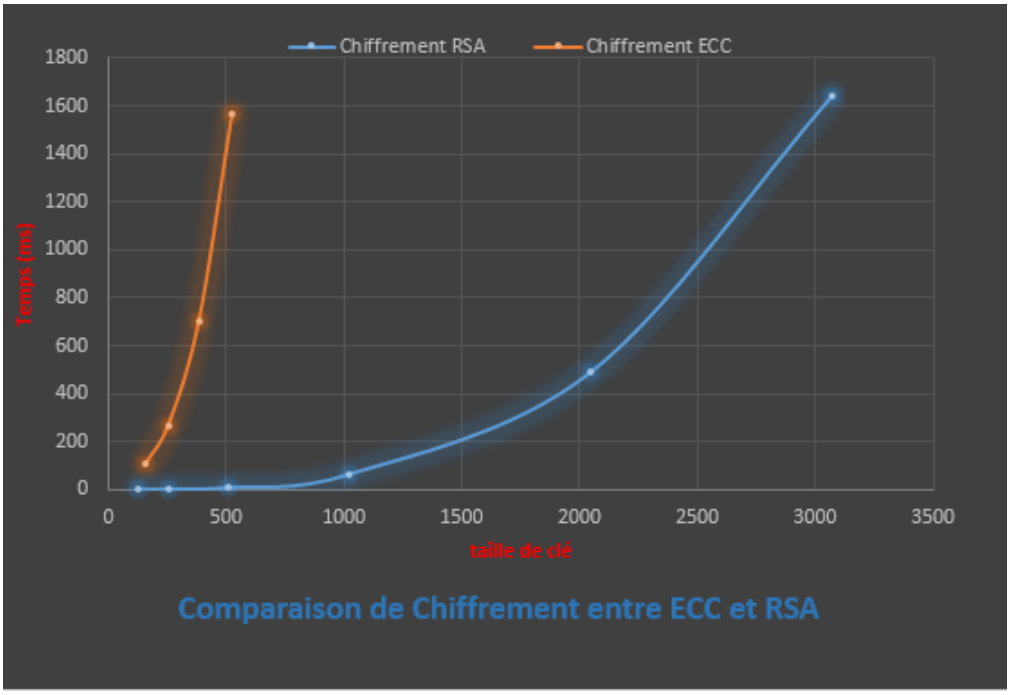

FIGURE 3.15 – Courbe comparative du temps d'exécution de chiffrement entre ECC et RSA.

Déchiffrement ECC		Déchiffrement RSA	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	10	128	0,6
256	27	256	2
384	65	512	16
521	148	1024	124
		2048	952
		3076	3155

TABLE 3.8 – Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au RSA.

3 Déchiffrement ECC vs RSA

Dans le tableau 3.8 et le graphique représentée à la figure 3.16 qui montrent le temps d'exécution de ECC et RSA de déchiffrement, la fonction entre le temps d'exécution et la taille de clé des deux systèmes RSA et ECC , s'augmentent et elle est similaire à la fonction carrée, la fonction RSA reste meilleure que la fonction ECC en termes des taille de clé.

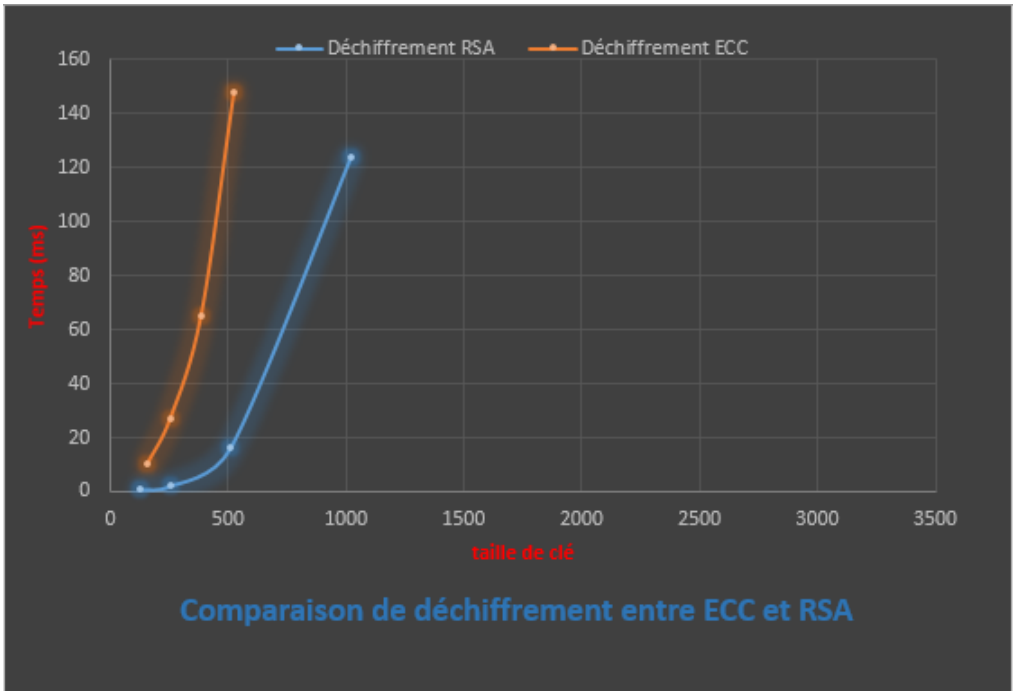

FIGURE 3.16 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et RSA.

3.7.1.2 Comparaison de changement du temps d'exécution ECC et RSA sur le même niveau de sécurité (équivalente des clé) :

Le niveau de sécurité entre l'ECC et RSA est la taille de la clé comparée à la force de chiffrement, l'ECC peut fournir la même force de chiffrement qu'un système basé sur l'algorithme RSA, mais avec des clés beaucoup plus courtes, par exemple, une clé ECC de 256 bits équivaut à des clés RSA de 3072 bits, qui sont alors 50% plus longues que les clés de 2048 bits couramment utilisées à l'heure actuelle, selon l'institut national des normes et de la technologie(NIST). [47]
Le tableau 3.9 montre une comparaison des tailles de clés entre ECC et RSA et le niveau de sécurité entre eux : [44]

Bits de sécurité	RSA	ECC
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

TABLE 3.9 – Comparaison les taille de clé entre ECC et RSA pour la force de chiffrement.

Le tableau 3.10 montre une comparaison de temps d'exécution entre ECC et RSA sur le niveau de sécurité entre eux :

Taille clé (Bit)		RSA			ECC		
	équivalent	Généclé(ms)	Chiff(ms)	Déchif(ms)	Généclé(ms)	Chiff(ms)	Déchif(ms)
1	1024/160	4344	63	124	35	105	10
2	3072/256	194804	1642	3155	54	269	27
3	7680/384	>1h	>1h	>1h	89	706	65

TABLE 3.10 – Comparaison du temps d'exécution entre ECC et RSA en mode équivalent sur les tailles des clés.

1 Génération de clé ECC vs RSA

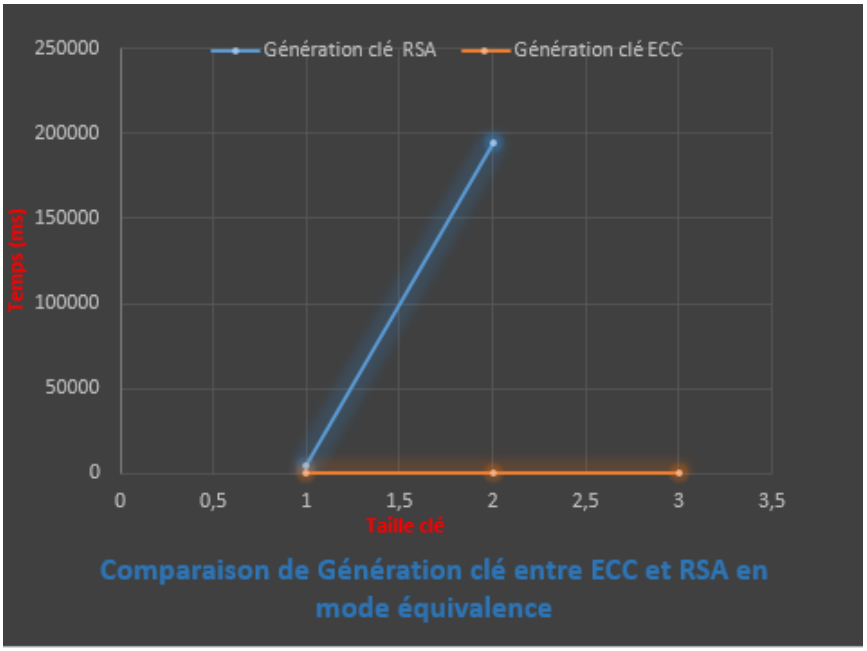

FIGURE 3.17 – Courbe comparative du temps d'exécution de génération de clé entre ECC et RSA dans le cas de clés équivalentes.

Dans le tableau 3.10 et le graphique représentée à la figure 3.17 qui montrent le temps d'exécution de ECC et RSA de génération des clés (mode équivalent), nous constatons qu'il existe une grande différence entre ECC et RSA dans le cas de clés équivalentes, de sorte que ECC est similaire à la fonction Linéarité et proche de zéro par rapport à RSA qui s'apparente à une fonction affine croissante.

2 Chiffrement ECC vs RSA

Dans le tableau 3.10 et le graphique représentée à la figure 3.18 qui montrent le temps d'exécution de ECC et RSA de chiffrement (mode équivalent), on remarque dans la clé 1024/160 RSA est meilleur que ECC au moment de l'exécution, puis nous notons que ECC devient parfait par rapport à RSA.

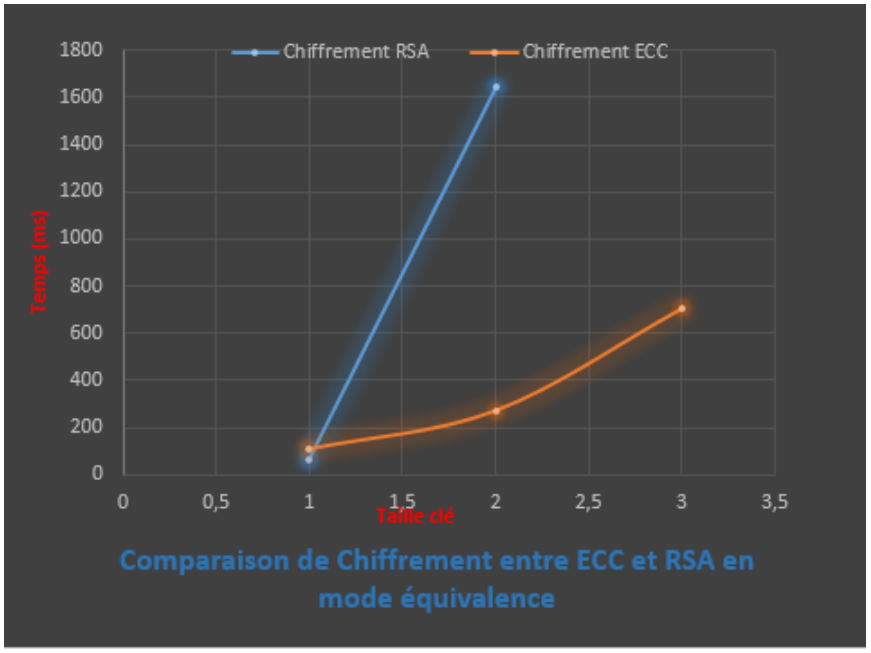

FIGURE 3.18 – Courbe comparative du temps d'exécution du chiffrement entre ECC et RSA dans le cas de clés équivalentes.

3 Déchiffrement ECC vs RSA

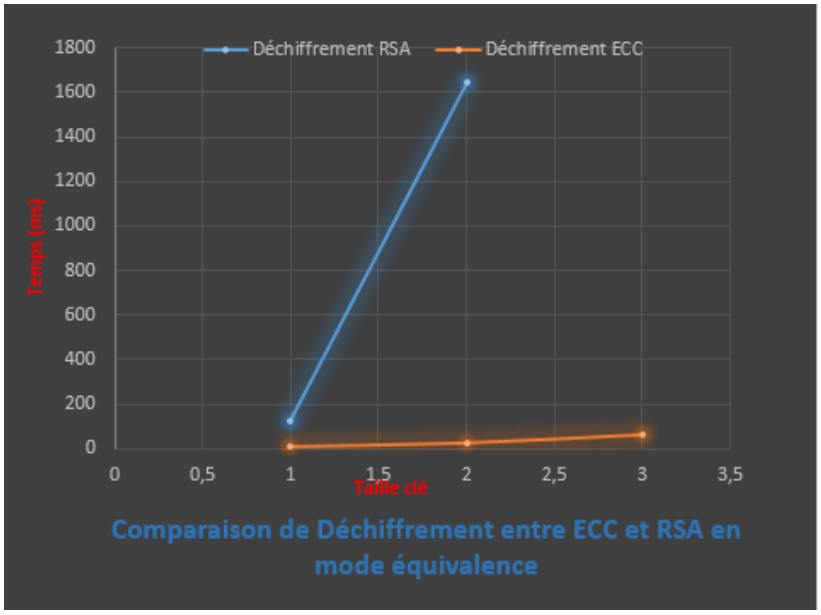

FIGURE 3.19 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et RSA dans le cas de clés équivalentes.

Dans le tableau 3.10 et le graphique représentée à la figure 3.19 qui montrent le temps d'exécution de ECC et RSA de déchiffrement (mode équivalent), on a la même note la génération des clés ECC vs RSA dans le cas d'équivalence.

3.7.2 ECC vs El Gamal

3.7.2.1 Comparaison de changement du temps d'exécution ECC et El Gamal sur les taille de clé

1 Génération de clé ECC vs El Gamal

Génération clé ECC		Génération clé El Gamal	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	160	24
256	54	256	41
384	89	512	107
521	179	1024	808
		2048	7020

TABLE 3.11 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au El Gamal.

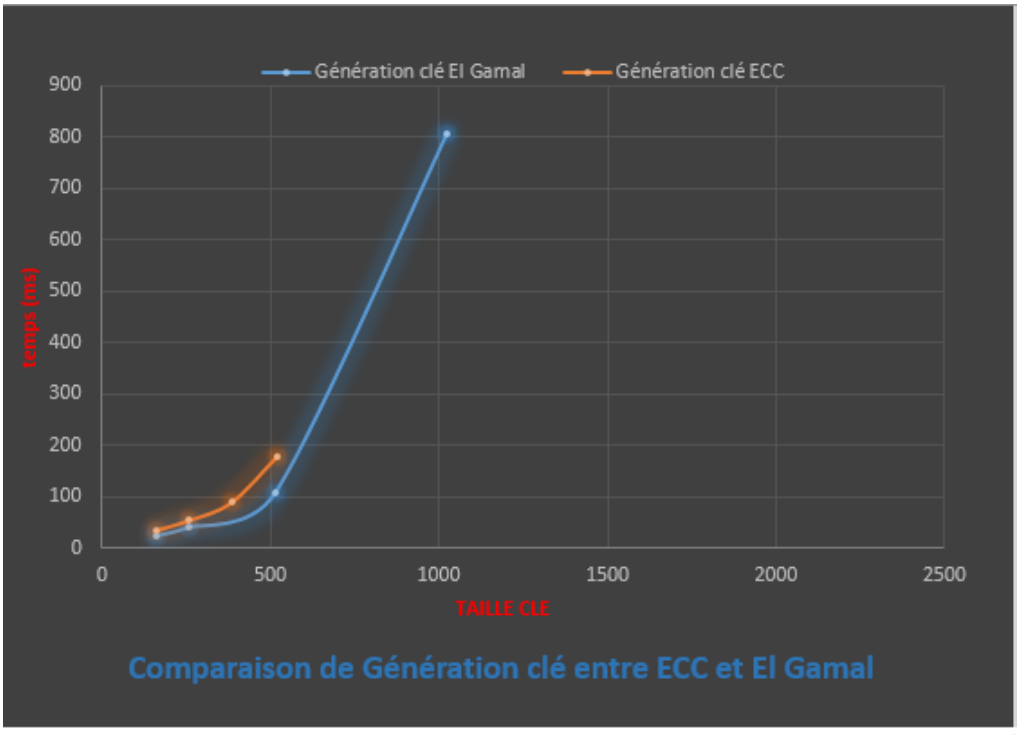

FIGURE 3.20 – Courbe comparative du temps d'exécution de génération de clé entre ECC et El Gamal.

Dans le tableau 3.11 et le graphique représentée à la figure 3.20 qui montrent le temps d'exécution de ECC et ElGamal de génération des clés, nous remarquons une augmentation significative du temps d'exécution pour les deux systèmes ECC et El Gamal, de sorte qu'El Gamal semble meilleur que ECC, mais il y a une convergence entre eux.

2 Chiffrement ECC vs El Gamal

Chiffrement ECC		Chiffrement El Gamal	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	105	160	0,43
256	269	256	1,24
384	706	512	5
521	1564	1024	32
		2048	240

TABLE 3.12 – Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au El Gamal.

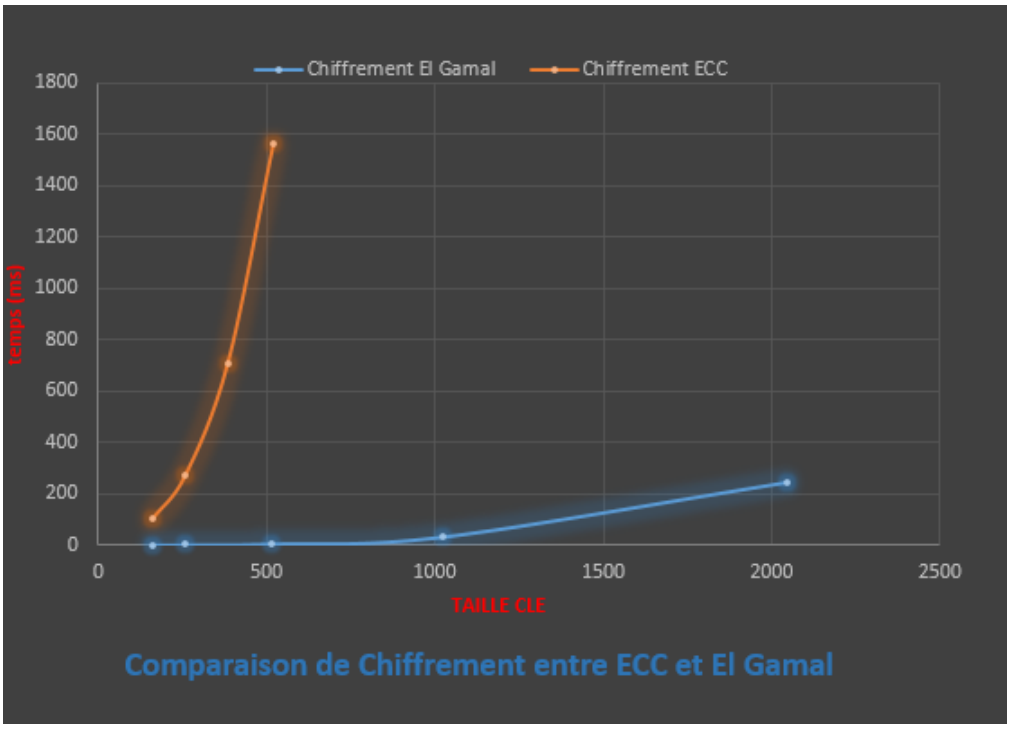

FIGURE 3.21 – Courbe comparative du temps d'exécution de chiffrement entre ECC et El Gamal.

Dans le tableau 3.12 et le graphique représentée à la figure 3.21 qui montrent le temps d'exécution de ECC et ElGamal de chiffrement, on voit une augmentation du temps d'exécution pour les deux systèmes, ECC est comme une fonction carrée et ElGamal est comme une fonction linéaire et augmente ensuite de façon affine mais El Gamal reste parfait par rapport à ECC.

3 Déchiffrement ECC vs El Gamal

Déchiffrement ECC		Déchiffrement El Gamal	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	10	160	0,6
256	27	256	0,68
384	65	512	3,5
521	148	1024	19
		2048	121

TABLE 3.13 – Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au El Gamal.

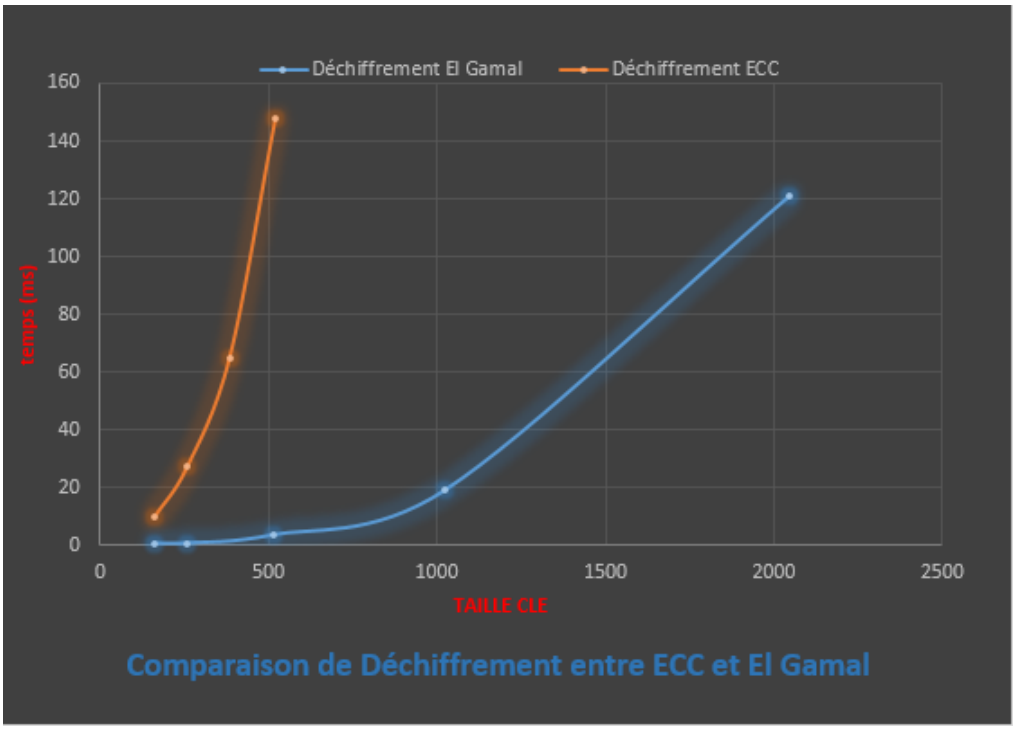

FIGURE 3.22 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et El Gamal.

Dans le tableau 3.13 et le graphique représentée à la figure 3.22 qui montrent le temps d'exécution de ECC et El Gamal de déchiffrement, on remarque la même commentaire pour le chiffrement ECC vs El Gamal seulement que la courbe de El Gamal augmente plus vite.

3.7.2.2 Comparaison de changement du temps d'exécution ECC et El Gamal sur le même niveau de sécurité (équivalente des clé) :

L'ECC peut fournir la même force de chiffrement qu'un système basé sur l'algorithme El Gamal, mais avec des clés plus courtes, par exemple, une clé ECC de 256 bits équivaut à des clés El Gamal de 2048 bits, selon l'institut national des normes et de la technologie

(NIST). [47]
Le tableau 3.14 montre une comparaison des tailles de clés entre ECC et El Gamal : [44]

Bits de sécurité	El Gamal	ECC
80	512	160
112	2048	256
256	15360	512

TABLE 3.14 – Comparaison les taille de clé entre ECC et El Gamal pour la force de chiffrement.

Le tableau 3.15 montre une comparaison de temps d'exécution entre ECC et ElGamal sur le niveau de sécurité entre eux :

Taille clé (Bit)		El Gamal			ECC		
	équivalent	Généclé(ms)	Chiff(ms)	Déchif(ms)	Généclé(ms)	Chiff(ms)	Déchif(ms)
1	512/160	107	1,24	3,5	35	105	10
2	2048/256	7020	240	121	54	269	27
3	15360/521	<1h	<1h	<1h	89	706	65

TABLE 3.15 – Comparaison du temps d'exécution entre ECC et El Gamal en mode équivalent sur les tailles des clés.

1 Génération clé ECC vs El Gamal

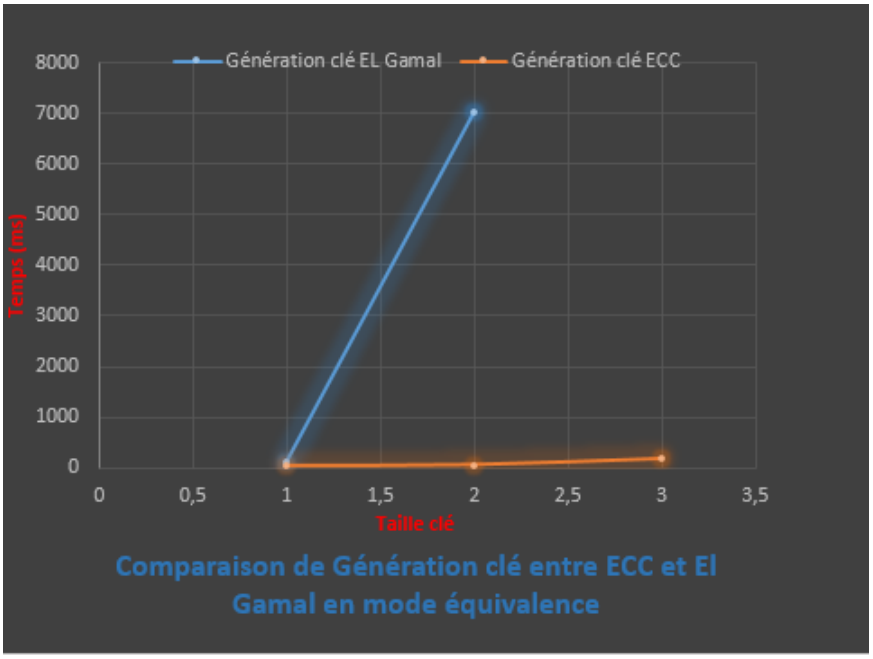

FIGURE 3.23 – Courbe comparative du temps d'exécution de génération de clé entre ECC et El Gamal dans le cas de clés équivalentes.

Dans le tableau 3.15 et le graphique représentée à la figure 3.23 qui montrent le temps d'exécution de ECC et ElGamal de génération des clés (mode équivalent), nous constatons qu'il existe une grande différence entre ECC et El Gamal dans le cas où les clés sont

équivalentes, de sorte que ECC est parfaitement similaire à la fonction linéarité proche de zéro par rapport à El Gamal qui s'apparente à une fonction affine croissance.

2 Chiffrement ECC vs El Gamal

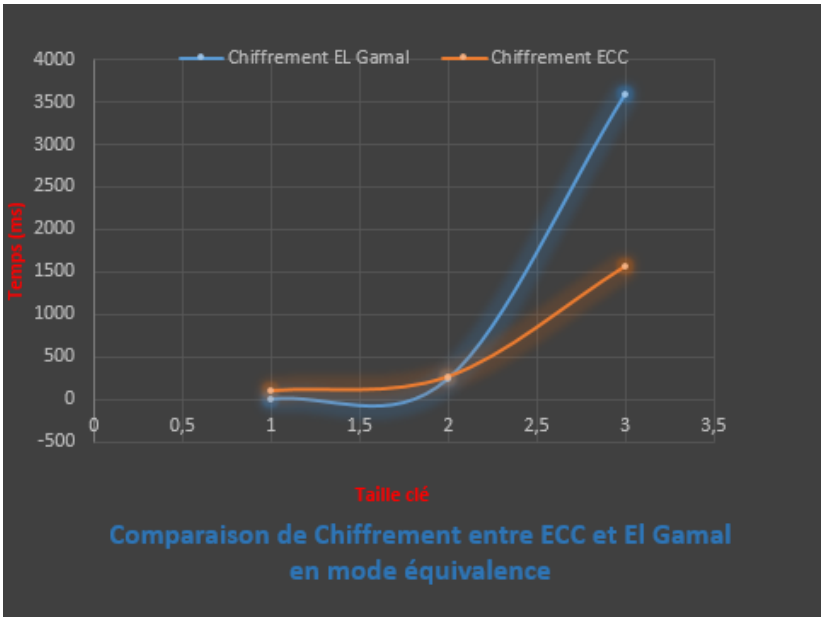

FIGURE 3.24 – Courbe comparative du temps d'exécution de chiffrement entre ECC et El Gamal dans le cas de clés équivalentes.

Dans le tableau 3.15 et le graphique représentée à la figure 3.24 qui montrent le temps d'exécution de ECC et ElGamal de chiffrement (mode équivalent), on remarque que dans la clé 512/160, El Gamal est meilleur que ECC à l'exécution, puis on remarque que dans la clé 2048/256, ECC devient parfait par rapport à El Gamal.

Déchiffrement ECC vs El Gamal

Dans le tableau 3.15 et le graphique représentée à la figure 3.25 qui montrent le temps d'exécution de ECC et El Gamal de déchiffrement (mode équivalent), nous constatons que dans la clé 512/160 environ le même temps d'exécution puis ils augmentent, la fonction ECC devient bien meilleure que l'El Gamal.

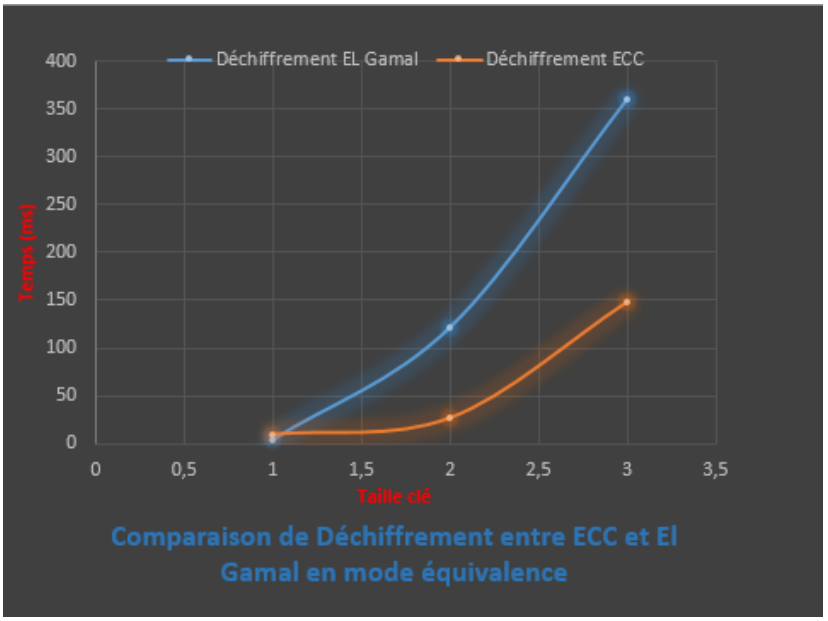

FIGURE 3.25 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et El Gamal dans le cas de clés équivalentes.

3.7.3 ECC vs Diffie Hellman

3.7.3.1 Comparaison de changement du temps d'exécution ECC et Diffie Hellman sur les tailles de clé

1 Génération de clé ECC vs Diffie Hellman

Génération clé ECC		Génération clé Diffie Hellman	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	128	33
256	54	256	72
384	89	512	281
521	179	1024	2181
		2048	26164

TABLE 3.16 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au Diffie Hellman.

Dans le tableau 3.16 et le graphique représentée à la figure 3.26 qui montrent le temps d'exécution de ECC et Diffie Hellman de génération des clés, nous notons que les deux systèmes augmentent rapidement en fonction du temps d'exécution comme la fonction carrée, mais le système ECC est meilleur que le système d'échange de clés Diffie Hellman.

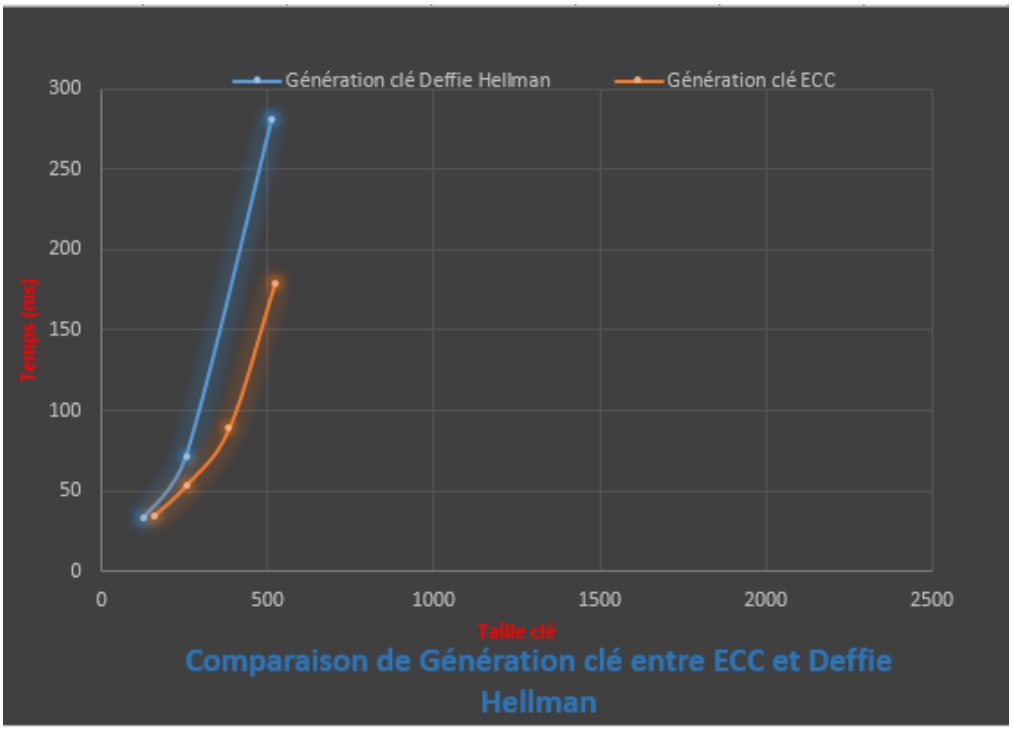

FIGURE 3.26 – Courbe comparative du temps d'exécution de génération de clé entre ECC et Diffie Hellman.

3.7.4 ECC vs AES

3.7.4.1 Comparaison de changement du temps d'exécution ECC et AES sur les tailles de clé

1 Génération de clé ECC vs AES

Génération clé ECC		Génération clé AES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	128	0,91
256	54	192	0,75
384	89	256	0,87
521	179		

TABLE 3.17 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au AES.

Dans le tableau 3.17 et le graphique représentée à la figure 3.27 qui montrent le temps d'exécution de ECC et AES de génération des clés.

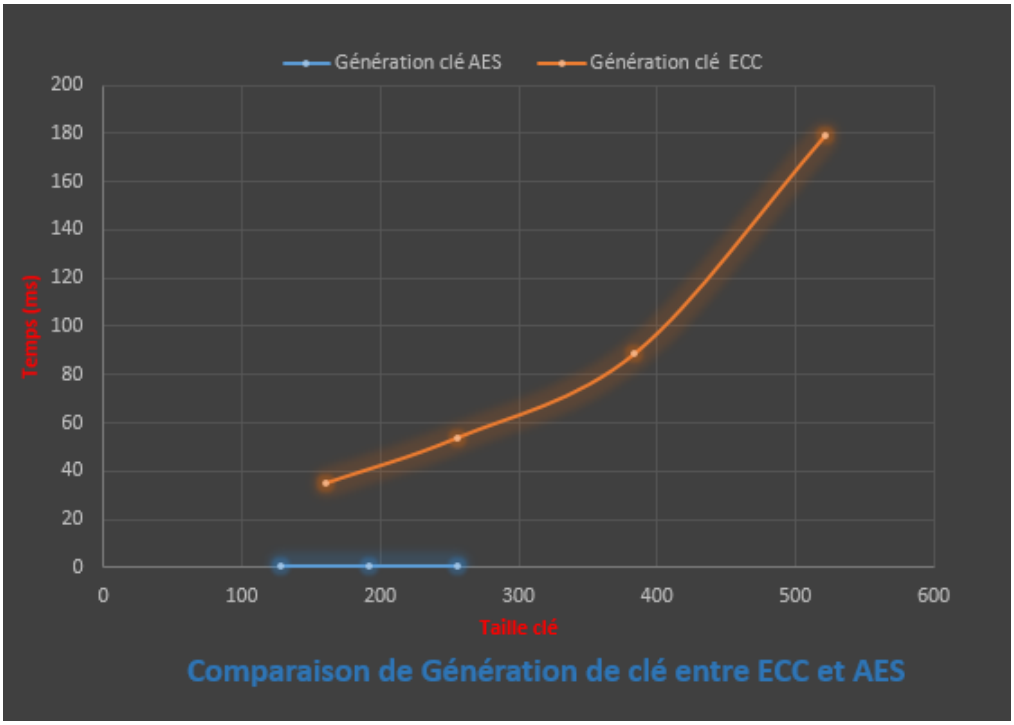

FIGURE 3.27 – Courbe comparative du temps d'exécution de génération de clé entre ECC et AES.

2 Chiffrement ECC vs AES

Chiffrement ECC		Chiffrement AES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	105	128	1,02
256	269	192	0,85
384	706	256	0,97
521	1564		

TABLE 3.18 – Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au AES.

Dans le tableau 3.18 et le graphique représentée à la figure 3.28 qui montrent le temps d'exécution de ECC et AES de chiffrement .

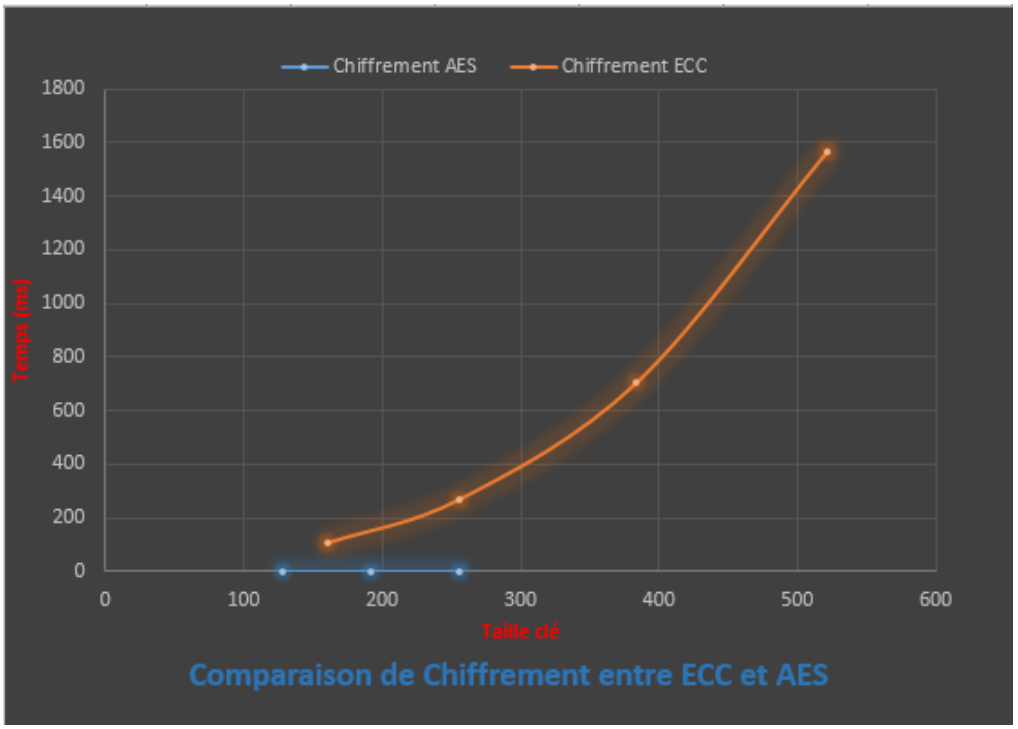

FIGURE 3.28 – Courbe comparative du temps d'exécution de chiffrement entre ECC et AES.

3 Déchiffrement ECC vs AES

Déchiffrement ECC		Déchiffrement AES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	10	128	0,86
256	27	192	0,9
384	65	256	0,82
521	148		

TABLE 3.19 – Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au AES.

Dans le tableau 3.19 et le graphique représentée à la figure 3.29 qui montrent le temps d'exécution de ECC et AES de déchiffrement .

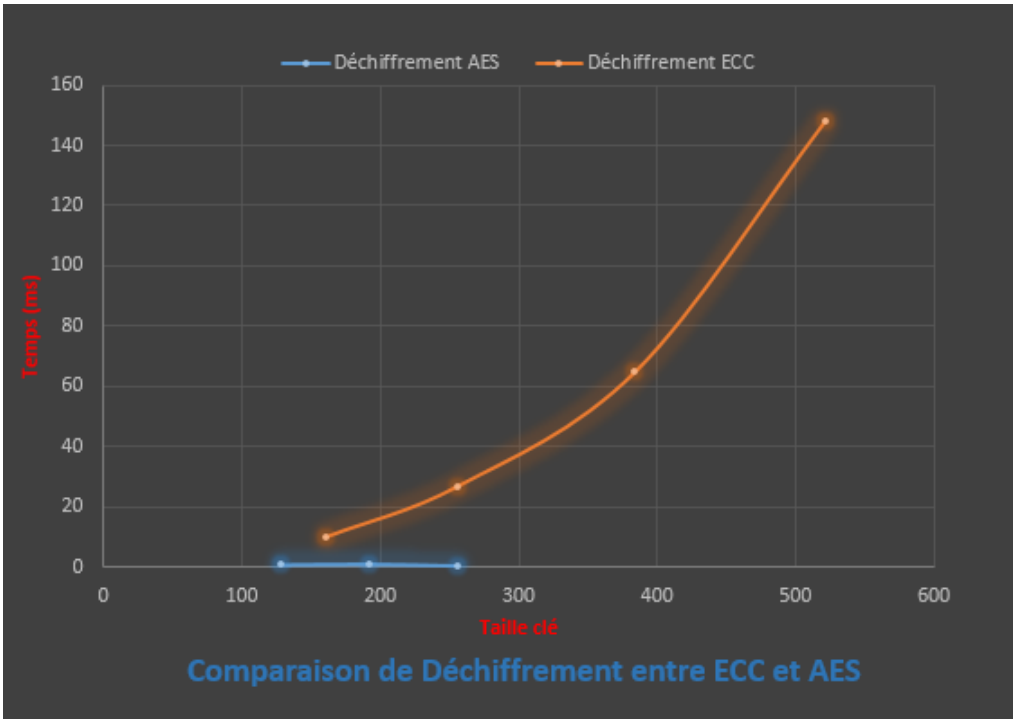

FIGURE 3.29 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et AES.

3.7.4.2 Comparaison de changement du temps d'exécution ECC et AES sur le même niveau de sécurité (équivalente des clé)

L'ECC peut fournir la même force de chiffrement qu'un système basé sur l'algorithme AES, mais avec des clés longue, par exemple, une clé ECC de 256 bits équivaut à des clés AES de 128 bits, Selon l'institut national des normes et de la technologie(NIST). [47]
Le tableau 3.20 montre une comparaison de temps d'exécution entre ECC et AES sur le niveau de sécurité entre eux :

Taille clé (Bit)		AES			ECC		
	équivalent	Généclé(ms)	Chiff(ms)	Déchif(ms)	Généclé(ms)	Chiff(ms)	Déchif(ms)
1	128/256	0,915	1,027	0,86	54	269	27
2	192/384	0,757	0,85	0,9	89	706	65
3	256/521	0,87	0,97	0,82	179	1564	148

TABLE 3.20 – Comparaison du temps d'exécution entre ECC et AES en mode équivalent sur les tailles des clés.

1 Génération de clé ECC vs AES

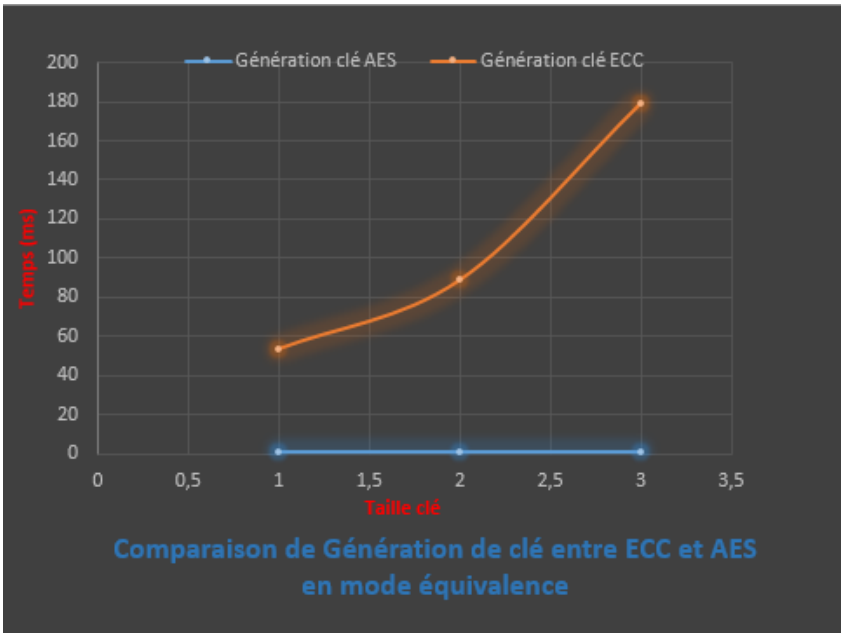

FIGURE 3.30 – Courbe comparative du temps d'exécution de génération de clé entre ECC et AES dans le cas de clés équivalentes.

Dans le tableau 3.20 et le graphique représentée à la figure 3.30 qui montrent le temps d'exécution de ECC et AES de génération des clés (mode équivalent).

2 Chiffrement ECC vs AES

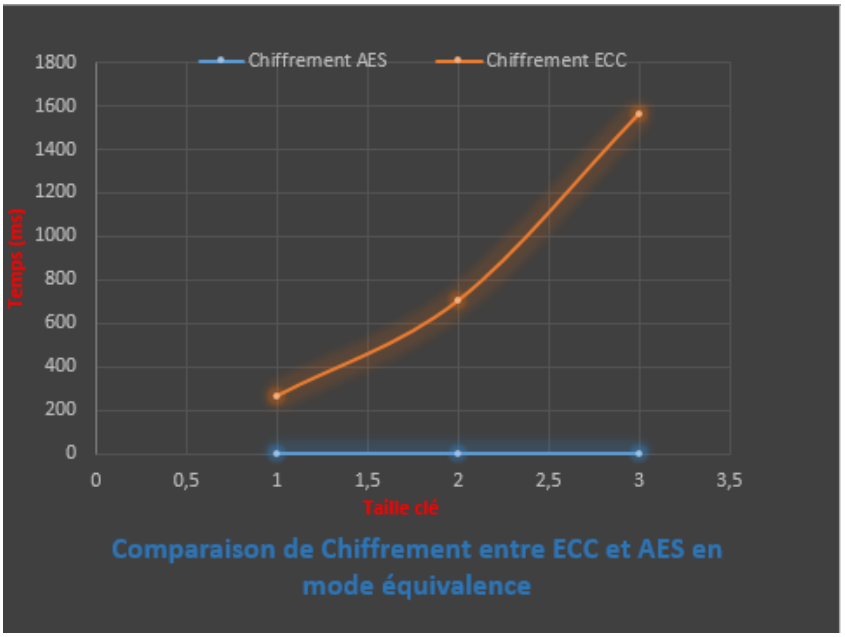

FIGURE 3.31 – Courbe comparative du temps d'exécution de chiffrement entre ECC et AES dans le cas de clés équivalentes.

Dans le tableau 3.20 et le graphique représentée à la figure 3.31 qui montrent le temps d'exécution de ECC et AES de chiffrement (mode équivalent)..

3 Déchiffrement ECC vs AES

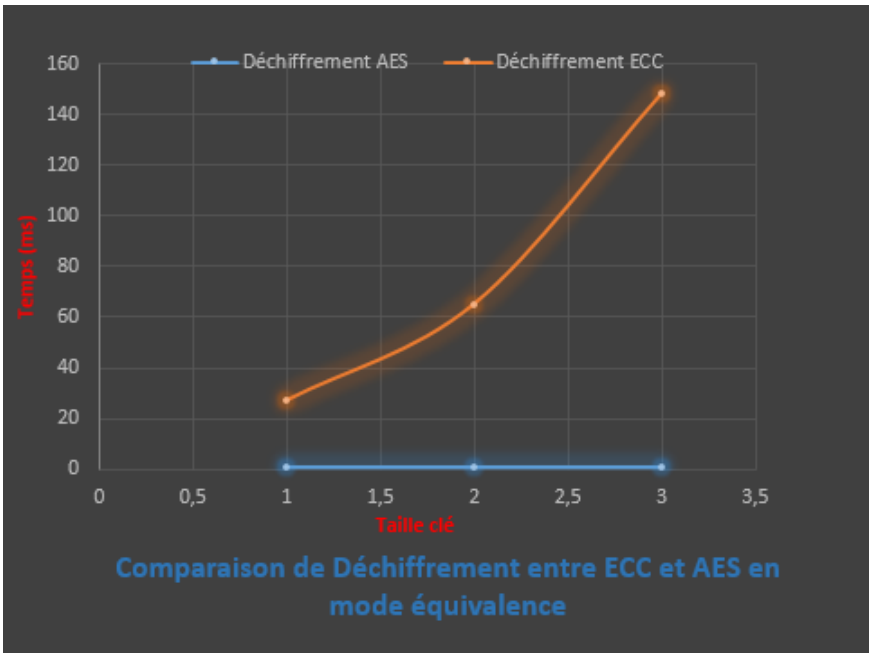

FIGURE 3.32 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et AES dans le cas de clés équivalentes.

Dans le tableau 3.20 et le graphique représentée à la figure 3.32 qui montrent le temps d'exécution de ECC et AES de déchiffrement (mode équivalent).

3.7.5 ECC Vs TripleDES

3.7.5.1 Comparaison de changement du temps d'exécution ECC et TripleDES sur les taille de clé

1 Génération de clé ECC vs TripleDES

Génération clé ECC		Génération clé TripleDES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	112	1,93
256	54	168	2,93
384	89		
521	179		

TABLE 3.21 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au TripleDES.

Dans le tableau 3.21 et le graphique représentée à la figure 3.33 qui montrent le temps d'exécution de ECC et TripleDES de génération des clés.

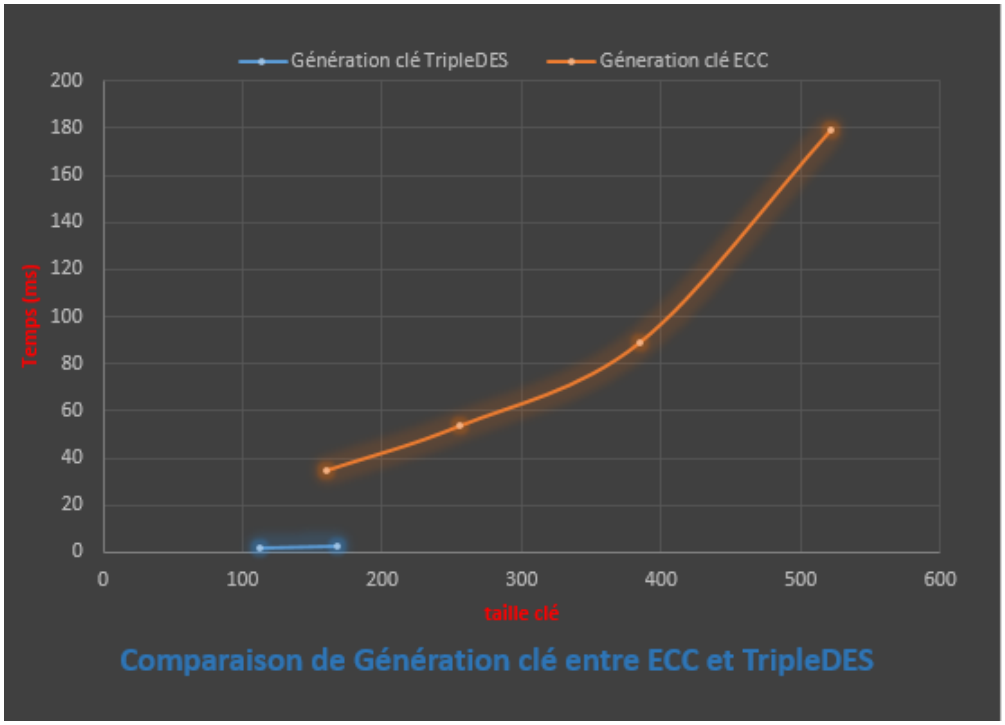

FIGURE 3.33 – Courbe comparative du temps d'exécution de génération de clé entre ECC et TripleDES.

2 Chiffrement ECC vs TripleDES

Chiffrement ECC		Chiffrement TripleDES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	105	112	1,64
256	269	168	2,15
384	706		
521	1564		

TABLE 3.22 – Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au TripleDES.

Dans le tableau 3.22 et le graphique représentée à la figure 3.34 qui montrent le temps d'exécution de ECC et TripleDES de chiffrement.

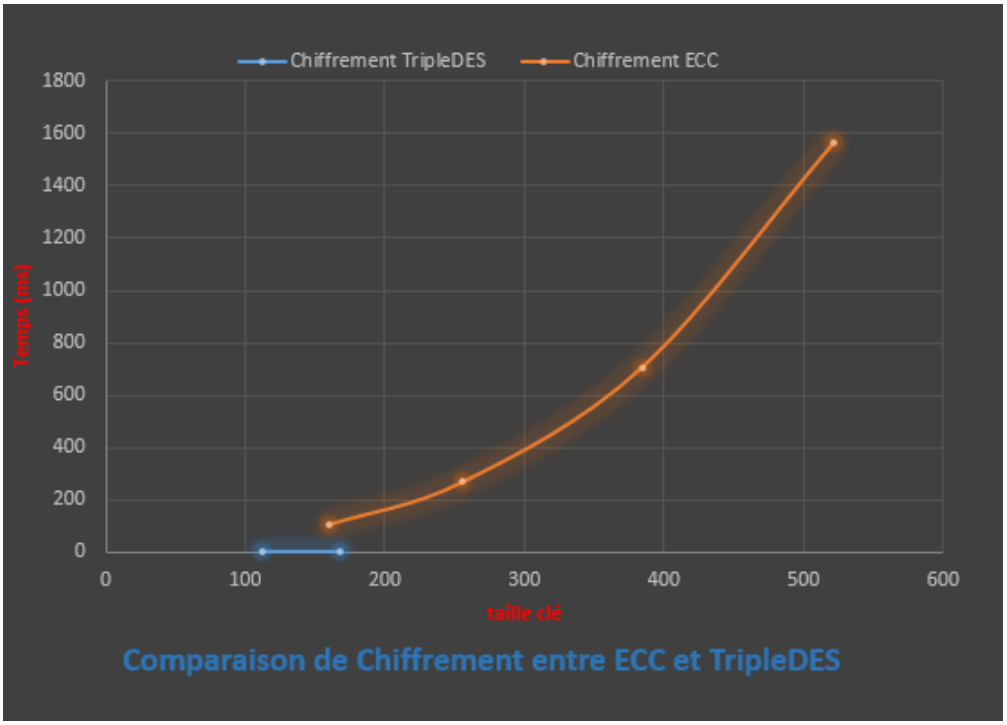

FIGURE 3.34 – Courbe comparative du temps d'exécution de chiffrement entre ECC et TripleDES

3 Déchiffrement ECC vs TripleDES

Déchiffrement ECC		Déchiffrement TripleDES	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	10	112	0,27
256	27	168	0,29
384	65		
521	148		

TABLE 3.23 – Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au TripleDES

Dans le tableau 3.23 et le graphique représentée à la figure 3.35 qui montrent le temps d'exécution de ECC et TripleDES de déchiffrement.

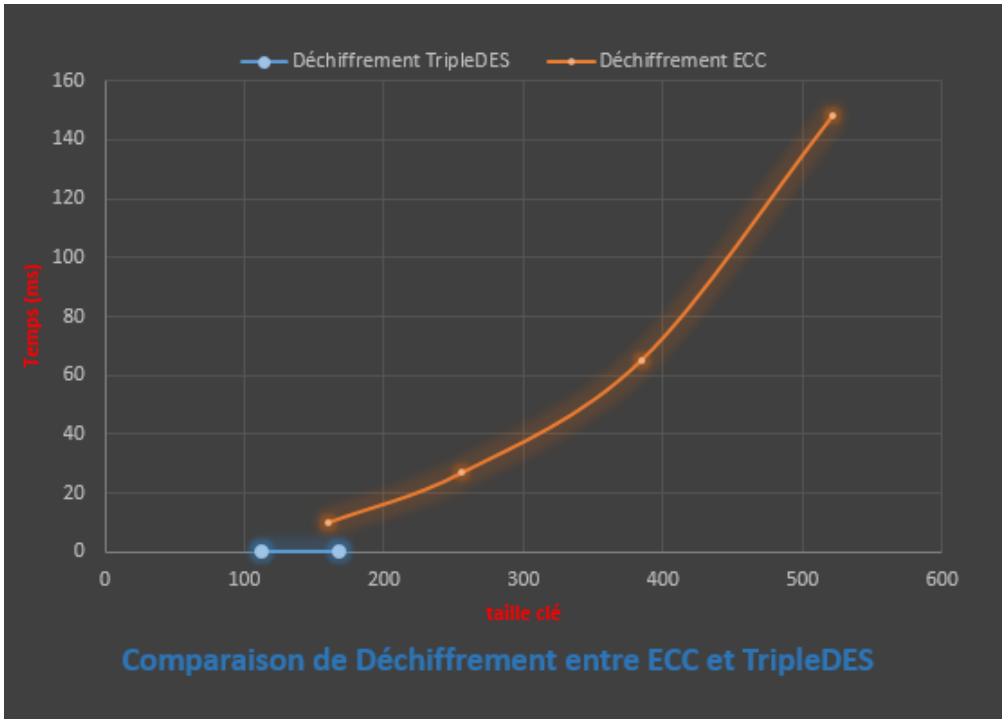

FIGURE 3.35 – Courbe comparative du temps d'exécution duedéchiffrement entre ECC et TripleDES.

3.7.6 ECC vs RC4

3.7.6.1 Comparaison de changement du temps d'exécution ECC et RC4 sur les tailles de clé

1 Génération de clé ECC vs RC4

Génération clé ECC		Génération clé RC4	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	35	64	0,57
256	54	80	0,58
384	89	104	0,55
521	179	128	0,19

TABLE 3.24 – Comparaison du temps d'exécution en fonction des tailles des clés pour la génération des clés ECC par rapport au RC4.

Dans le tableau 3.24 et le graphique représentée à la figure 3.36 qui montrent le temps d'exécution de ECC et RC4 de génération des clés.

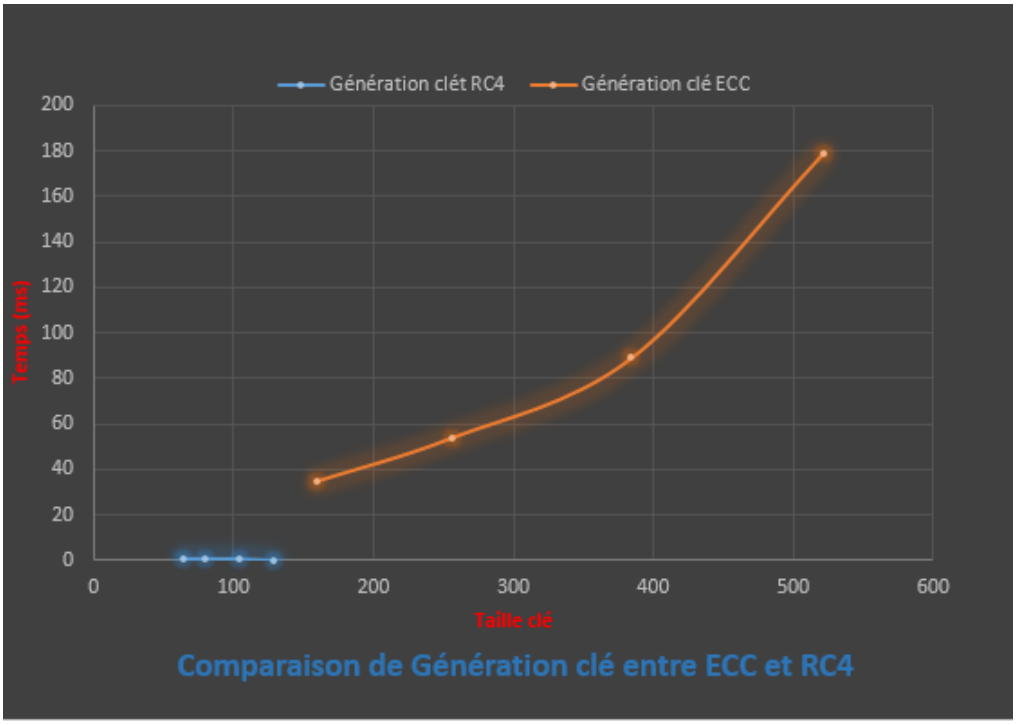

FIGURE 3.36 – Courbe comparative du temps d'exécution de génération de clé entre ECC et RC4.

2 Chiffrement ECC vs RC4

Chiffrement ECC		Chiffrement RC4	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	105	64	0,28
256	269	80	0,26
384	706	104	0,34
521	1564	128	0,08

TABLE 3.25 – Comparaison du temps d'exécution en fonction des tailles des clés pour le chiffrement ECC par rapport au RC4.

Dans le tableau 3.25 et le graphique représentée à la figure 3.37 qui montrent le temps d'exécution de ECC et RC4 de chiffrement.

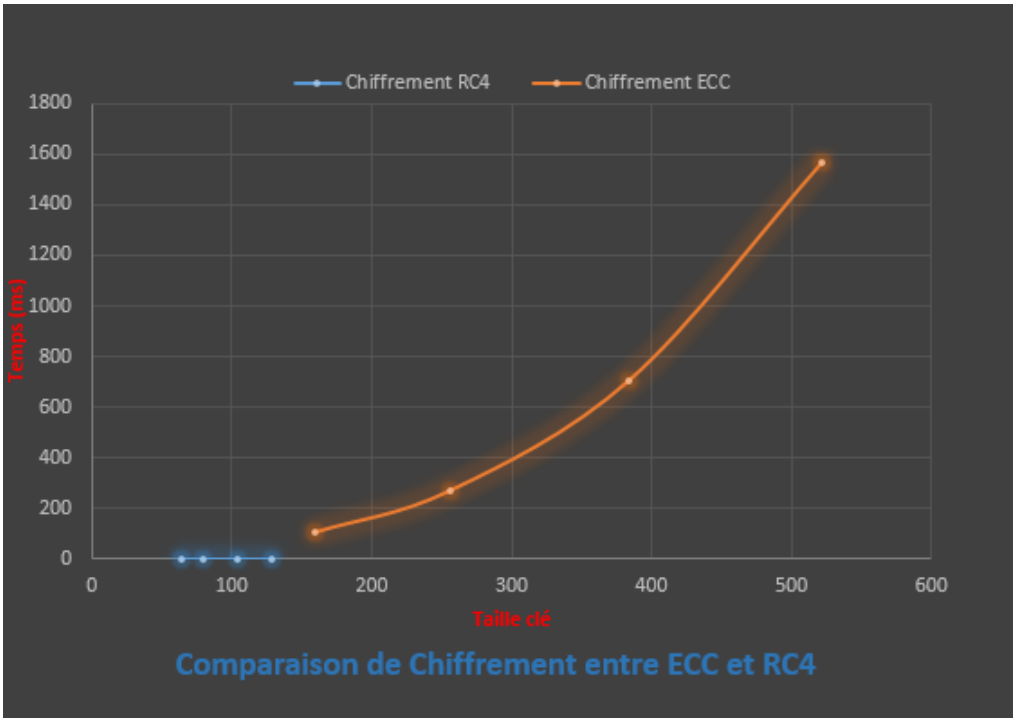

FIGURE 3.37 – Courbe comparative du temps d'exécution de chiffrement entre ECC et RC4.

3 Déchiffrement ECC vs RC4

Déchiffrement ECC		Déchiffrement RC4	
Taille clé	Temps (ms)	Taille clé	Temps (ms)
160	10	64	0,053
256	27	80	0,054
384	65	104	0,056
521	148	128	0,068

TABLE 3.26 – Comparaison du temps d'exécution en fonction des tailles des clés pour le déchiffrement ECC par rapport au RC4 .

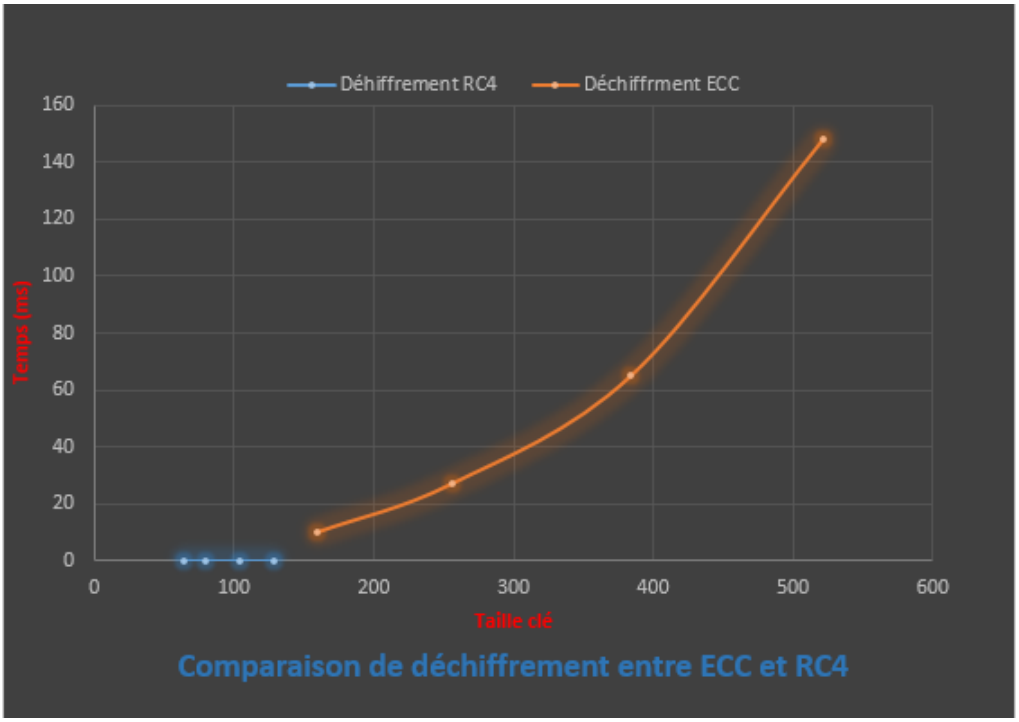

FIGURE 3.38 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et RC4.

Dans le tableau 3.26 et le graphique représentée à la figure 3.38 qui montrent le temps d'exécution de ECC et RC4 de déchiffrement.

D'après les tableaux et graphiques précédents (pages 61...72) des systèmes de chiffrement symétrique AES, TripleDES et RC4, il s'avère que ces systèmes sont bien meilleurs que le système ECC tant par rapport à les tailles de clé qu'au même niveau de sécurité, Ces systèmes est similaire à une fonction linéaire proche de zéro.

3.7.7 ECC vs tous les Systèmes étudiés

3.7.7.1 Comparaison de changement du temps d'exécution ECC et tous les systèmes étudiés sur les tailles de clé

1 Génération de clé ECC vs tous les Systèmes étudiés

la figure 3.39 présenté un courbe comparative du temps d'exécution de génération de clé entre ECC vs tous les systèmes étudiés.

Les systèmes de chiffrement symétriques ressemblent tous d'avantage à une fonction linéaire et le temps de génération de clé est proche de zéro, tandis que les systèmes asymétriques le temps de génération des clés augmente rapidement.

Le système de chiffrement El Gamal est le meilleur parmi les systèmes asymétriques en termes de temps d'exécution pour la génération des clés, suivi par ECC, Diffie Hellman et RSA.

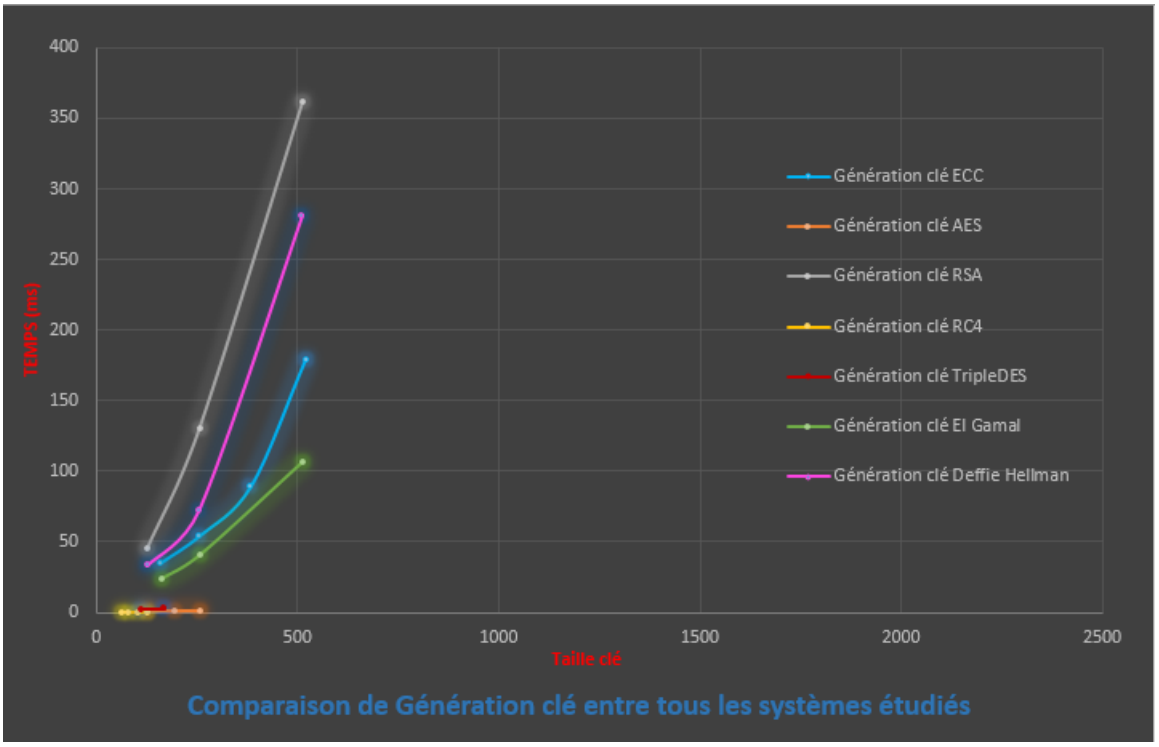

FIGURE 3.39 – Courbe comparative du temps d'exécution de génération de clé entre ECC et tous les Systèmes étudiés.

2 Chiffrement ECC vs tout les Systèmes étudiés

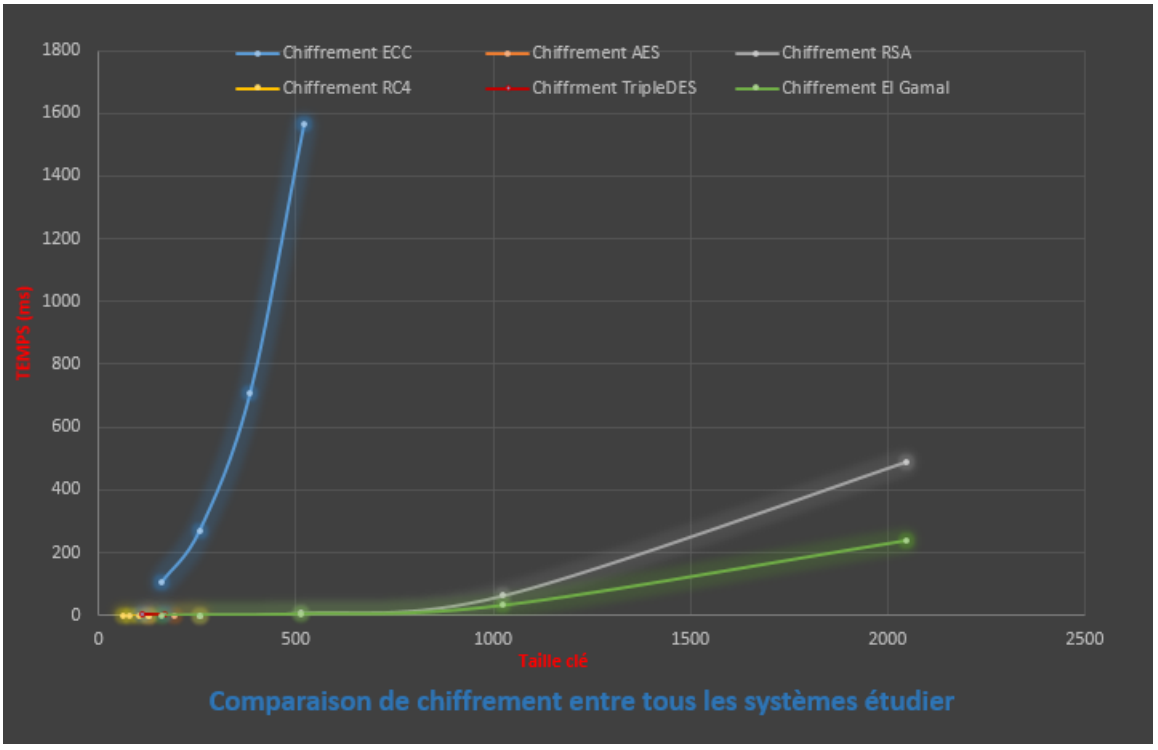

FIGURE 3.40 – Courbe comparative du temps d'exécution de chiffrement entre ECC et tous les systèmes étudiés.

la figure 3.40 présenté un courbe comparative du temps d'exécution de chiffrement

entre ECC vs tous les systèmes étudiés.
Les systèmes de chiffrement symétriques sont parfaits par rapport aux autres, El Gamal est meilleur que les systèmes asymétriques, puis RSA, ECC est similaire à la fonction carrée.

3 Déchiffrement ECC vs tout les systèmes étudiés

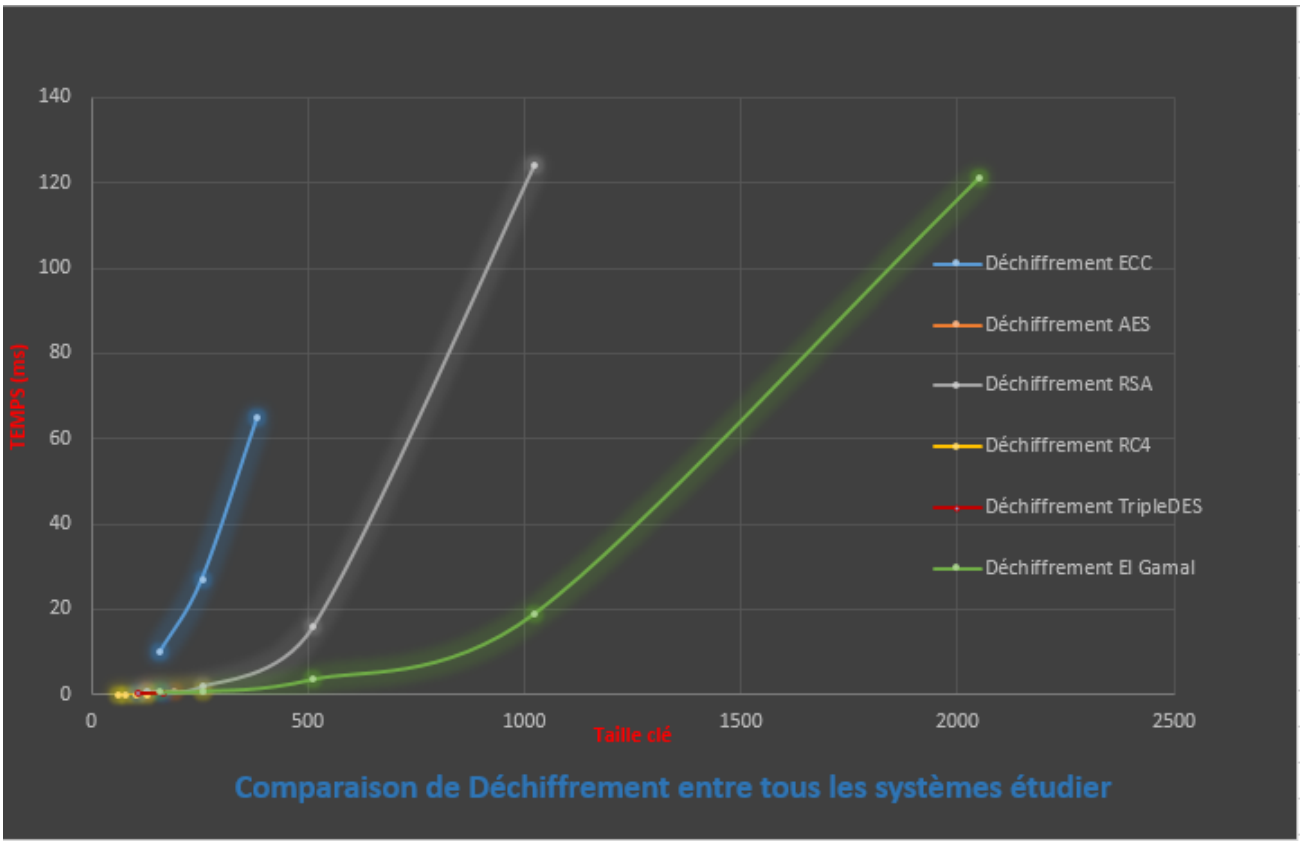

FIGURE 3.41 – Courbe comparative du temps d'exécution de déchiffrement entre ECC et tous les systèmes étudiés.

la figure 3.41 présenté un courbe comparative du temps d'exécution de déchiffrement entre ECC vs tous les systèmes étudiés, on a le même chose pour le chiffrement seuls RSA et El Gamal augmentent plus rapidement.

3.7.7.2 Comparaison de changement du temps d'exécution ECC et tous les Systèmes de chiffrement étudiés sur le même niveau de sécurité (équivalente des clé)

1 Génération de clé ECC vs tous les systèmes étudiés

Le tableau 3.27 et le graphique représentée à la figure 3.42, présenté un comparative de temps d'exécution entre ECC et tous les systèmes étudiés pour la génération des clés en même niveau de securité.

Les systèmes de chiffrement ECC, AES, El Gamal se caractérisent par un temps d'exécution très faible par rapport le système RSA.
AES est le parfait en termes de temps, puis ECC, puis El Gamal, et le dernier est RSA.

Génération des clé (équivalente des clé)				Temps d'exécution (ms)			
ECC	RSA	EL Gamal	AES	ECC	RSA	EL Gamal	AES
160	1024	512		35	4344	107	
256	3072	2048	128	54	194805	7020	0,91
384	7680		192	89	>1h		0,75
521		15360	256	179		>1h	0,87

TABLE 3.27 – Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes étudiés pour la génération des clés en mode équivalent les taille de clé .

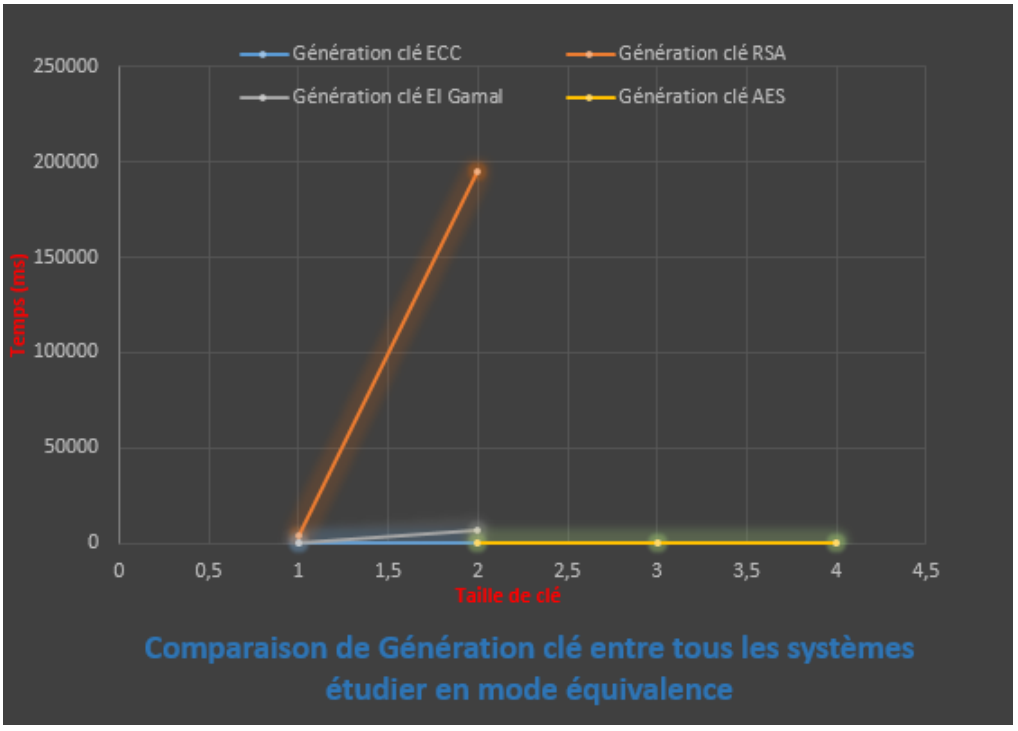

FIGURE 3.42 – Courbe comparative du temps d'exécution de génération de clé entre ECC et tous les systèmes de chiffrement dans le cas de clés équivalentes.

2 Chiffrement ECC vs tous les systèmes

La table 3.28 et le graphique représentée à la figure 3.43, présenté un comparative de temps d'exécution entre ECC et tous les systèmes étudiés pour le chiffrement en même niveau de securité.

Chiffrement (équivalente des clé)				Temps d'exécution (ms)			
ECC	RSA	EL Gamal	AES	ECC	RSA	EL Gamal	AES
160	1024	512		105	63	1,24	
256	3072	2048	128	269	489	240	1,02
384	7680		192	706	>1h		0,85
521		15360	256	1564		>1h	0,97

TABLE 3.28 – Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes pour le chiffrement en mode équivalent.

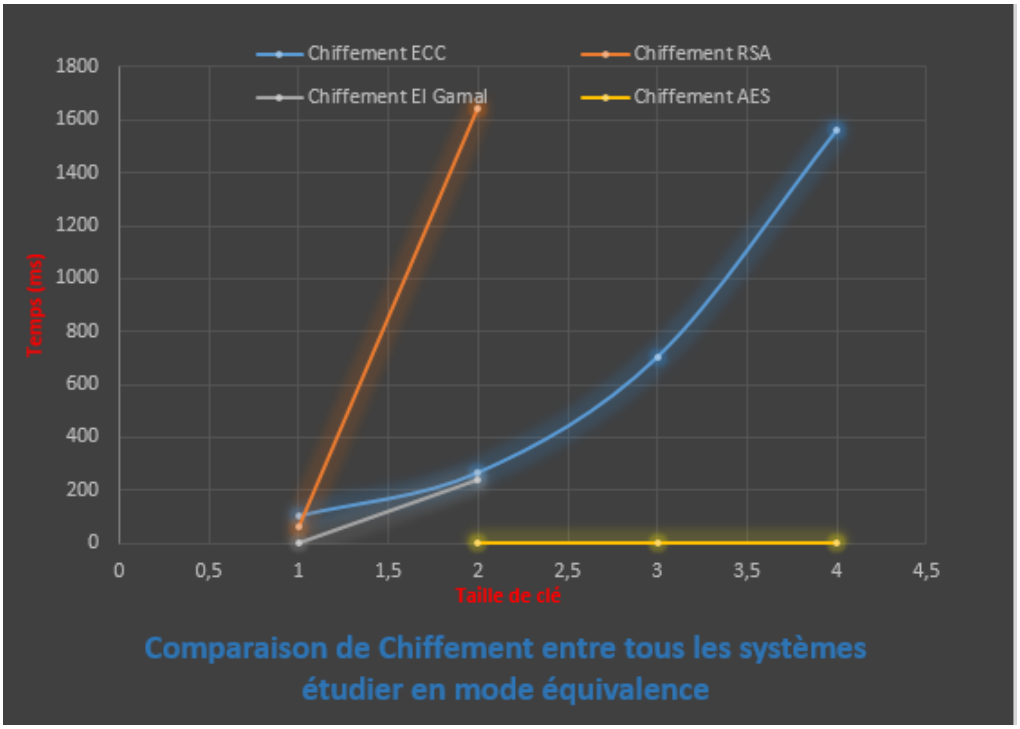

FIGURE 3.43 – Courbe comparative du temps d'exécution de chiffrement entre ECC et tous les Systèmes de chiffrement dans le cas de clés équivalentes.

Le système AES est idéal pour d'autres systèmes.
A la première clé, ECC est le pire en terme de temps, suivi de RSA puis El Gamal, mais il change et devient RSA le pire.

Déchiffrement ECC vs tous les système

La table 3.29 et le graphique représentée à la figure 3.44, présenté un comparative de temps d'exécution entre ECC et tous les systèmes étudiés pour le déchiffrement en même niveau de securité.

Déchiffrement (équivalente des clé)				Temps d'exécution (ms)			
ECC	RSA	EL Gamal	AES	ECC	RSA	EL Gamal	AES
160	1024	512		10	124	3,5	
256	3072	2048	128	27	3155	121	0,86
384	7680		192	65	>1h		0,9
521		15360	256	148		>1h	0,82

TABLE 3.29 – Comparaison le temps d'exécution (ms) entre ECC et tous les systèmes pour le déchiffrement en mode équivalent.

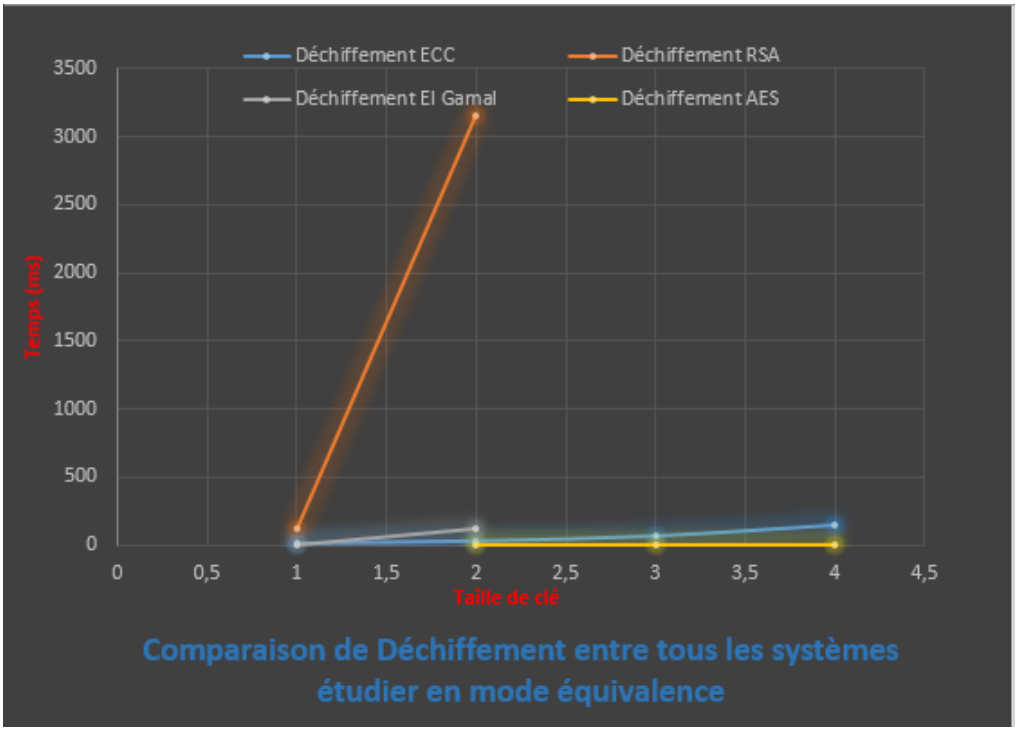

FIGURE 3.44 – Courbe comparative du temps d'exécution du chiffrement entre ECC et tous les Systèmes de déchiffrement dans le cas de clés équivalentes.

Nous constatons que l'AES est parfait pour d'autres systèmes, le temps de déchiffrement de ECC et El Gamal est à peu près le même alors que RSA toujours le pire avec une grande différence.

3.8 Discussion des résultats

Lorsque nous prenons les résultats du temps d'exécution en termes des tailles de clé et de même niveau de sécurité (cas de clés équivalentes), nous pouvons classer les systèmes comme suit :

3.8.1 ECC Vs Les systèmes symétriques

Le tableau 3.30 montre les résultats de ECC vs les système de chiffrements symétriques étudies.

ECC Vs Symétrique	
Génération clé	Symétrique
Chiffrement	Symétrique
Déchiffrement	Symétrique

TABLE 3.30 – Résultat de ECC Vs Symétrique.

Les systèmes symétriques surpassent ECC, dans les deux facteurs (facteur de taille des

clés et le facteur de même niveau de sécurité), pour toutes les opérations (génération des clés, chiffrement et déchiffrement), parce que les systèmes symétriques utilisent une seule clé privée et leurs algorithmes dépendent de la permutation et de la commutation, tandis que ECC dépend des nombres premiers de grande taille et des opérations mathématiques effectuées sur eux telles que l'addition et le doublement qui prennent beaucoup de temps.

3.8.2 ECC Vs Les systèmes Asymétriques

Nous pouvons présenter les résultats en deux parties :

1. **Sur le facteur de taille de clé**
   Le tableau 3.31 montre les résultats de ECC vs les système de chiffrements asymétriques étudies en terme de taille de clé.

ECC Vs Asymétrique (même taille des clés)			
	ECC Vs RSA	ECC Vs El Gamal	ECC Vs Diffie Hellman
Génération clé	ECC	El Gamal	ECC
Chiffrement	RSA	El Gamal	
Déchiffrement	RSA	El Gamal	

TABLE 3.31 – Résultat de ECC Vs Asymétrique en terme des tailles de clé.

Lorsque nous comparons des systèmes asymétriques selon la taille de clé, nous constatons que ECC est meilleur que RSA et Diffie Hellman dans la génération des clés, tandis que le chiffrement et le déchiffrement RSA et El Gamal sont meilleurs que ECC, en termes du temps d'exécution, cette différence est due aux différents algorithmes et à leur mode de fonctionnement.

2. **Sur le facteur de même niveau de sécurité**
   Le tableau 3.32 montre les résultats de ECC vs les système de chiffrements asymétriques étudies en terme de même niveau de sécurité.

ECC Vs Asymétrique (même niveau de sécurité)	
Génération clé	ECC
Chiffrement	ECC
Déchiffrement	ECC

TABLE 3.32 – Résultat de ECC Vs Asymétrique en même niveau de sécurité.

Lorsque nous avons comparé ECC avec des systèmes asymétriques en termes de même niveau de sécurité, nous avons constaté que ECC est le meilleur en raison de ses petits clés par rapport aux autres systèmes.

3.9 Conclusion

D'après notre étude, nous pouvons dire que l'ECC peut atteindre un certain niveau de sécurité que les systèmes asymétriques atteignent avec des clés beaucoup plus grandes, tandis que les systèmes symétriques avec des clés plus petites, Bien que le temps d'exécution ECC soit moyen pour les systèmes asymétriques. en effet, ce système devient aujourd'hui le plus attirant en raison de sa haute sécurité et est utilisé dans de nombreux domaines nécessitant un niveau de sécurité très élevé tels que le bitcoin, les passeports biométriques et les signatures numériques.

CONCLUSION GÉNÉRALE

Les courbes elliptiques fait partie d'un domaine très important de la cryptographie qui présente de nombreux avantages en termes d'efficacité et de sécurité, et jusqu'à présent, personne n'a prouvé qu'elles ne pouvaient pas les craquer ou les décrypter en un temps relativement court, en fait de nombreux scientifiques et mathématiciens travaillent dans ce domaine.

Ce travail a été réalisé dans le cadre de projet de fin d'étude au niveau de master au sein d'université de jijel. dans lequel nous avons appliqué beaucoup de compétences et connaissances acquises durant nos années d'étude, nous avons fait une étude comparative du temps d'exécution selon la taille de clé entre ECC et les systèmes de chiffrements les plus populaires, symétriques tels que AES, 3DES et RC4 et asymétriques tels que RSA, El Gamal et Diffie Hellman.

Nous avons constaté que le temps d'exécution d'ECC par la même taille de clé est moyen par rapport aux autres systèmes asymétriques et varie selon le processus (génération des clés, chiffrement ou déchiffrement), et par le même niveau de sécurité étant le meilleur d'entre eux, pour les systèmes symétriques, ils surpassent ECC en terme du temps d'exécution pour les deux facteurs, parce qu'ils dépendent qu'une seule clé secrète et que ses algorithmes ne dépendent pas de problèmes mathématiques difficiles.

On peut conclure que l'ECC est adéquate d'être utilisé dans des domaines et des applications qui nécessitent une très haute sécurité comme le bitcoin et les signatures numériques en s'attendant à ce que la cryptographie à base des courbes elliptiques devienne l'un des systèmes les plus populaires pour ses nombreux avantages et soit utilisé dans la plupart des applications pour protéger leurs informations sensibles et confidentielles.

Alors d'après notre étude, nous conseillions d'utiliser l'ECC, bien que cela prenne un temps moyen entre les systèmes de chiffrement asymétriques, mais il est hautement sécurisé avec des clés courtes relatives aux autres systèmes, et est recommandé pour les applications qui nécessitent un niveau de sécurité élevé avec un temps d'exécution moyen.

Nous envisageons de faire, comme continuité de notre travail, des démonstrations mathématiques sur l'efficacité d'ECC en terme du niveau de sécurité, d'espace mémoire et d'énergie.

Nous envisageons aussi une étude comparative selon la complexité de l'algorithme de chaque système de chiffrement et le chiffrement des images avec ECC.

BIBLIOGRAPHIE

[1] J. Fattahi. Analyse des protocoles cryptographiques par les fonctions témoins. *Thèse Doctorat en informatique, Université Laval*, 2016.

[2] MA. Cheragui. Cryptographie. *Cour, Plateforme d'ensignement à distance Université d'Adrar*.

[3] l'encyclopédie informatique CommentÇaMarche. Cryptographie-chiffrement par substitution. *document, Communauté informatique*, https ://web.maths.unsw.edu.au/ lafaye/CCM/crypto/crypto.htm, consulter en janvier 2022.

[4] SH.Nawal. Conception et réalisation d'un système collaboratif pour les experts métier à base d'agent et des algorithmes de cryptage. *Thèse de doctorat, Université Ahmed ben Bella d'Oran*, 2017.

[5] ichi.pro.fr. 8 bits uniquement :qu'est-ce que la cryptographie ? *Article*, 2020, consulter en février 2022.

[6] I. Makhlouf. les systèmes informatiques complexes. *Mémoire de master, Université Larbi Ben M'hidi (Oum El Bouaghi)*, 2019.

[7] K. Stefan P. Fabien. Information hiding techniques for steganography and digital watermarking. *liver*, 203.162.10.101, 2000.

[8] I. Souici. La cryptographie moderne , principes fondateurs de la cryptographie. *cour master 1 ilm , Université Mohammed Seddik Ben Yahia (jijel)*, 2021.

[9] N. Merabet. Développement et mise en œuvre de méthodes de cryptographie et de tatouage pour la protection de données numériques. *Thèse de Doctorat ,Université de Batna 2*, 2018.

[10] Y.Challal H.Bettahar. Introduction à la sécurité informatique. *Articl*, 2008.

[11] R. Yende. cours de sécurité informatique. *Cour informatique, , Facultés Africaine Bakhita*, 2018.

[12] S. Jajodia Hca Van Tilborg. Encyclopedia of cryptography and security. *Livre, Springer Boston MA*, 978-1-4419-5907-2, 2011.

[13] M. Videau. Critère de sécurité des algorithmes de chiffrement à clé secrète. *Thèse de Doctorat, Université de Paris 6, (France)*, 2005.

[14] C. Blondeau. La cryptanalyse différentielle et ses généralisations. *Thèse de doctorat, Université Pierre et Marie Curie*, 2011.

[15] G. Dartois B. Daniel. Cryptographie paris 13. *Cours informatique*, 2010.

[16] B.Rabab. Sécurité des images numériques compressées jpeg. *Thèse de doctorat, Université Djillali Liabès de Sidi Bel Abbes*, 2019.

[17] DR.Stinson. Cryptography : theory and practice. *livre informatique, Third Edition*, 2005.

[18] A.Ben Ammar K. Haddouche. Amélioration de la génération des sous clés de l'algorithme cryptographique des. *Mémoire de Master, Université Akli Mohand Oulhadj Bouira (Algerie)*, 2017.

[19] K.Allouane S.allo. Cryptographie et sécurité des réseaux implémentation de l'aes sous matlab. *Mémoire de fin d'études,Universite Mouloud Mammeri Tizi-Ouzou*, 2008.

[20] W.Stallings. Cryptography and network security : Principles and practice. *livre informatique*, 2014.

[21] SecuriteInfo.com. L'aes : Advanced encryption standard. *Article*, 2001, consulter en février 2022.

[22] M. Ramdhani. Problème de sécurité dans les réseaux capteurs avec prise en charge de l'énergie. *Mémoire de magistère, Université Saad Dahlab De Blida (Algérie)*, 2013.

[23] Academic.com. Cryptosysteme de elgamal. *fr-academic.com*, 2010, consulter en mars 2022.

[24] N.Atta H.Cherfa. Etude sur l'applicabilité de la cryptographie asymétrique aux réseaux de capteurs sans fil. *Mémoire de Mastère ,Université Abderrahmane Mira (Bejaïa)*, 2012.

[25] I. Souici. Cryptosystèmes hybrides. *cour master 1, Universite Mohammed Seddik Ben Yahia (jijel)*, 2021.

[26] L. Ghislaine. Introduction à la cryptologie. *document, http ://www. labouret.net/crypto/*, 1998, consulter en mars 2022.

[27] A. Zoé. Cryptographie quantique et applications spatiales. *Thèse de Doctorat, Université de Limoges*, 2016.

[28] R.Assi. Qu'est-ce que le chiffrement quantique et est-il vrai qu'il n'est pas piratable ? *Article*, 2017.

[29] Y.Shou. Cryptographie sur les courbe elliptique et tolérance aux pannes dans les réseaux de capteurs. *Thèse de Doctorat, Université de Franche-Comté*, 2014.

[30] C.Murdica. Sécurité physique de la cryptographie sur courbes elliptiques. *These de Doctorat, Ecole de l'institut mines*, 2014.

[31] F.Thierry. Cryptographie et courbes elliptiques. *Projet de semestre, Ecole polytechnique fédérale de lausanne*, 2011.

[32] A.Loiseau. Implémentation légère et sécurisée pour la cryptographie sur courbes elliptiques pour l'internet des objets. *Thèse de Doctorat, Université de Lyon opérée au sein de l'Ecole des Mines de Saint-Etienne*, 2019.

[33] JoMendes. Factorisation par courbe elliptique. *Article*, https ://cryptobourrin.wordpress.com/, 2018, consulter en mars 2022.

[34] A. Gluher. Problème du logarithme discret appliqué à la cryptanalyse sur courbe elliptique : algorithme mov. *Article*, 2015.

[35] V. Verneuil. Courbes elliptiques et attaques par cannaux auxiliaires. *Article*, 2009.

[36] S.Pontie. Sécurisation matérielle pour la cryptographie à base de courbes elliptiques. *These de Doctorat, Université Grenoble Alpes*, 2016.

[37] J.Razakarivony T.Rakotondraina, F.Randimbindrainible. Performances des cryptosystème basés sur les courbes elliptiques. *Article informatique, Ecole Supérieure Polytechnique Antananarivo Université d'Antananarivo*, 2010.

[38] H. Bouchakour. la sécurité de l'information par le biais des courbes elliptiques. *Thèse de Doctorat, Université Djillali Liabes, Sidi Bel Abbés*, 2018.

[39] C.Gonçalves. Cryptographie avancé courbes elliptiques. *Livre*, 2015.

[40] Wikipédia. Elliptic curve digital signature algorithm. *Article, consulter en mai 2022.*

[41] I. Lotfi. Cryptographié à base de courbes elliptiques. *Thése de doctorat PhD thesis, Ecole Nationale Supérieure d'Informatique*, 2017.

[42] G. Chanut. La programmation avec bitcoin : Les courbes elliptiques. *Article*, 2019.

[43] Wikipédia. Java$_{(}$*langage*$)$. *Article.*

[44] Les Numériques. Java se development kit (jdk). *Article*, 2022, consulter en mai 2022.

[45] Wikipédia. Netbeans. *Article.*

[46] Certicom Research. Sec 2 :recommended elliptic curve domain parameters. *document, Version 1.0*, 2000.

[47] D. Giry. Blue krypt|cryptographic key length recommendation. *https ://www.keylength.com/ v32.3*, 2020, consulter en avril 2022.

[48] H. Boumerzoug. Gestion de sécurité dans les réseaux de capteurs sans fil. *Mémoire de magistère, Université de Québec a trois rivières*, 2011.

[49] R. Tripathi S. Agrawal. Comparative study of symmetric and asymmetric cryptography techniques. *International Journal of Advance Foundation and Research in Computer (IJAFRC)*, 2014.

[50] L. Adleman RL. Rivest, A. Shamir. A method for obtaining digital signatures and public-key cryptosystems. *Article*, 21(2) :120–126, 1978.