الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE BLIDA 1
Faculté de Technologie
Département d'Électronique

MEMOIRE DE MASTER
En Télécommunication

Spécialité : Réseaux & Télécommunications

THÈME :

Exploration des Techniques de Chiffrement avec Python « Implémentation et Évaluation »

Réalisé par
BABA AMEUR Sabrina
EL MERRAOUI Safa

Encadré par
Mr. MEHDI Merouane

Juin 2024

وَلئن شكرتم لأزيدنكمْ

Je dédie ce mémoire avec toute ma gratitude.
À la source inépuisable de mon encouragement, celle qui m'a toujours soutenue avec un amour indéfectible. Maman, tu as été ma lumière dans les moments sombres, mon guide lorsque j'étais perdue, et le pilier sur lequel je me suis toujours appuyée. ton soutien, et ta confiance en moi ont été essentiels à ma réussite. À toi, **maman**,
À celui qui a fait de moi une femme, ma source de vie et d'affection, à mon support qui était toujours à mes côtés pour me soutenir et m'encourager, à mon prince, **Papa**.
À ma sœur Nihel et mon petit frère Mouad, qui ont partagé avec moi les rires et les larmes, et pour leur amour sincère et leur affection.
À la mémoire de mon grand-père, passionné des esprits cultivés et fervent défenseur de l'éducation, qui croyait en nous, ses petits-enfants, et qui nous encourageait toujours à nous dépasser, ses souvenirs m'ont toujours motivé, constituant ainsi une source d'inspiration constante tout au long de mon parcours académique.
À ma grand-mère, chez qui j'ai souvent trouvé refuge, et qui, dans ses prières, demandait la réussite de mes études. Sa tendresse et ses prières m'ont toujours insufflé une force.
À mes tantes, pour leur amour et leurs précieux conseils, ainsi qu'**à toutes mes cousines**, avec une mention spéciale, Nesrine, Samah et Sabrina, pour leur soutien inconditionnel et leur présence réconfortante.
À mes amis, et en particulier Yassmine et Ilhem, pour leur amitié indéfectible, leur soutien moral.
À tous ceux qui luttent pour la justice et la liberté des Palestiniens, symbole universel de résilience et de persévérance face à l'adversité.
Sans oublier ma binôme Sabrina, pour son soutien moral, sa patience et sa compréhension tout au long de ce projet, Merci d'avoir été à mes côtés dans les moments difficiles et de m'avoir encouragée à persévérer...

Safa

Je dédie ce mémoire à :

À ma mère, pour son amour inconditionnel, sa patience infinie et son soutien indéfectible. Ta confiance en moi et tes encouragements constants ont été les piliers de mon parcours et c'est grâce à toi que ce mémoire a vu le jour.

À mon père, pour sa sagesse, sa force et ses sacrifices. Ta persévérance et ton soutien m'ont toujours guidé, même dans les moments les plus difficiles. Tu as cru en moi avant que je ne croie en moi-même.

À mes frères, Ryad et Djabir, Vous êtes mes alliés, mes confidents, et mes plus grands supporters et **a ma grand-mère**, pour sa sagesse infinie et son amour inconditionnel. Ta tendresse et ton soutien ont été une source constante de réconfort et de motivation.

À ma seconde famille, pour son soutien réconfortant et ses précieux conseils. Votre générosité de cœur et votre bienveillance ont été des phares lumineux dans les moments de doute et **À Mounir** pour son soutien inconditionnel et sa présence réconfortante. Ta confiance en moi et ton encouragement ont été un moteur essentiel dans la réalisation de ce mémoire

À la mémoire de mes grands-parents et de ma tante décédés, dont les souvenirs et les valeurs continuent de m'inspirer chaque jour. Vous êtes présents dans chaque succès et chaque étape franchie.

À Selsabil, pour son amitié inestimable et ses encouragements constants. Ta gentillesse, ta compréhension et ton soutien m'ont apporté une grande force et**À Khalida,** pour son amitié précieuse et son soutien sans faille. votre présence dans ma vie est un cadeau inestimable..

À mon binôme Safa, pour son travail acharné, son dévouement et son soutien. Ton partenariat et ta collaboration ont été essentiels à la réalisation de ce mémoire.

À tout mon entourage, pour leur soutien moral et leur croyance en moi. . . .

Sabrina

Remerciement

Nous exprimons avant tout notre gratitude au Créateur, ALLAH, pour nous avoir dotés d'intelligence et pour nous avoir maintenus en bonne santé tout au long de ce travail.

Nous souhaitons remercier tout particulièrement notre directeur, M. Mehdi Merouane, dont le soutien indéfectible et les conseils avisés ont été d'une aide précieuse. Sa confiance en nos capacités et son accompagnement constant ont grandement contribué à la réussite de ce mémoire.

Nos remerciements vont également aux membres du jury qui ont accepté d'évaluer notre travail. Leur participation et leurs critiques constructives sont d'une grande importance pour nous.

Nous tenons à exprimer notre reconnaissance au Chef du Département d'Electronique, M. Hocine AIT SAADI, ainsi qu'à l'ensemble des enseignants et du personnel de notre département. Leur soutien et leurs encouragements ont été essentiels tout au long de notre parcours académique.

Nous adressons un merci chaleureux à nos familles, dont la présence réconfortante, l'écoute attentive, et le soutien constant nous ont permis de persévérer et de donner le meilleur de nous-mêmes.

Enfin, nous remercions toutes les personnes, proches ou moins proches, qui ont contribué de quelque manière que ce soit à l'élaboration de ce travail. Leur aide, même la plus modeste, a été d'une valeur inestimable et a enrichi notre cheminement.

ملخص

إن ضمان سرية البيانات الحساسة ضد الوصول غير المصرح به هو أولوية حاسمة يلعب فيها التشفير دوراً رئيسياًا في مواجهة نقاط الضعف لطرق التشفير القديمة ، يهدف هذا المشروع إلى تقييم وتجربة خوارزميات التشفير التقليدية والحديثة المختلفة من خلال تنفيذها على رسومية واجهة على بايثون ، واستخدام أدوات ومكتبات مختلفة. أبرزت نتائج الاختبارات التي أجريت على الواجهة كفاءتها التعليمية ، مما يسمح للمستخدمين بتمييز تقنيات التشفير المختلفة وتحقيق الهدف المستهدف.

الكلمات المفتاحية: الواجهة الرسومية ، السرية ، التشفير ، نقاط الضعف ، الخوارزميات الكلاسيكية ، الخوارزميات الحديثة ، بايثون.

Abstract

Guaranteeing the confidentiality of sensitive data against unauthorized access is a crucial priority in which cryptography plays a key role in the face of vulnerabilities of old encryption methods, this project aims to assess and experiment with different conventional and modern cryptography algorithms by implementing them on a graphical interface on Python, and using different tools and libraries. The results of the tests carried out on the interface highlighted its educational efficiency, allowing users to discern the different encryption techniques and to achieve the targeted objective.

Keywords: confidentiality, cryptography, vulnerabilities, classic algorithms, modern algorithms, graphical interface, python.

Résumé

Garantir la confidentialité des données sensibles contre les accès non autorisés est une priorité cruciale dans laquelle la cryptographie joue un rôle clé face aux vulnérabilités des anciennes méthodes de cryptage, ce projet vise a évaluer et expérimenter les différents algorithmes de cryptographie classiques et modernes en les implémentant sur une interface graphique sur Python, en utilisant différents outils et bibliothèques. Les résultats des tests effectués sur l'interface ont mis en évidence son efficacité pédagogique, permettant aux utilisateurs de discerner les différentes techniques de cryptage et d'atteindre l'objectif visé.

Mots Clée : cryptographie, sécurité des données, algorithmes classiques, algorithmes modernes, interface graphique, Python.

Table des matières

Table des figures

Introduction Générale

Imaginez-vous un monde où chaque mot tapé, chaque message envoyé, chaque transaction effectuée, c'est-à-dire chaque instant de votre vie numérique soit a la merci des regards indiscrets ? Un monde où la notion de vie privée s'évanouit, la confidentialité n'est qu'un mirage, où le vol d'identité et la fraude règnent en maîtres. Cette vision cauchemardesque pourrait devenir réalité sans la cryptographie, la forteresse virtuelle qui sécurise nos données à l'ère numérique. A l'heure où Internet tisse sa toile invisible a travers le monde, la cryptographie se révèle être une sentinelle incontournable. Elle chiffre nos messages, obscurcit nos transactions et enveloppe nos identités numériques d'un voile de sécurité, nous permettant traverser l'univers numérique en toute sérénité, sachant que nos données ne seront pas exposées aux enchères.

Néanmoins, la lutte pour la préservation des données est loin d'être gagnée. Dans les coulisses du cyberespace, une guerre acharnée se joue entre cryptographes et cryptanalystes. Les premiers élaborent des codes de plus en plus complexes, tandis que les seconds tentent les déchiffrer. Cette rivalité tenace stimule l'innovation et propulse cette discipline vers de nouveaux sommets à un rythme effréné.

Dans ce projet, nous vous invitons à plonger dans les méandres de la cryptographie qui est bien plus qu'une simple technique de codage car comprendre cette discipline, c'est comprendre les enjeux de la sécurité dans un monde où la protection des données est devenue une question de survie. C'est pour cela que nous avons développer une application graphique intuitive et convivial en utilisant le langage de programmation Python et diverses bibliothèques telles que Tkinter, Ttk, Messagebox, Crypto (PyCryptodome), Pillow (PIL), ainsi que les modules string, math et os intégrés à Python. Ces bibliothèques constituent le squelette du développement de nôtres interface.

L'objectif de notre application s'inscrit dans le cadres educatif permettant aux utilisateurs de découvrir et d'expérimenter de manière interactive les differents algorithmes de cryptage allant du classiques comme César et Vigenére en passant par la machine historique Enigma jusqu'à la cryptographie moderne tel que le DES, l'AES et le RSA et modernes, tout en offrant une expérience d'apprentissage immersive et sécurisée.

Le premier chapitre présente les généralités sur le cryptage, retraçant son histoire, ses principes fondamentaux, ses objectifs et ses types, ainsi que les méthodes de chiffrement classiques et modernes.

Le deuxième chapitre se concentre sur les mécanismes de chiffrement et de déchiffrement, analysant en détail le fonctionnement des algorithmes symétriques et asymétriques, leurs caractéristiques ainsi que leurs faiblesses et le processus de génération des clés pour certain.

Enfin, le troisième chapitre décrit la mise en pratique de ces concepts théoriques à travers le développement d'une interface graphique permettant aux utilisateurs d'explorer et de tester diverses méthodes de cryptage de manière interactive et conviviale

Chapitre 1

Généralités sur le cryptage

1.1 Introduction

Ce chapitre introductif explorera en profondeur les concepts fondamentaux du cryptage, discipline cruciale pour la protection des données confidentielles à l'ère numérique. Nous définirons le cryptage et retracerons son riche historique, des origines de la cryptographie classique à l'avènement de la cryptographie moderne et de ses algorithmes robustes. Les principes de base du cryptage seront abordés, notamment ses objectifs clés, les différents types de chiffrement existants ainsi que les principales techniques utilisées. Une attention particulière sera accordée aux faiblesses des anciennes méthodes ayant mené au développement des normes actuelles. Enfin, ce chapitre couvrira les protocoles de sécurité modernes essentiels, garants de la confidentialité des communications numériques.

1.2 Definition du cryptage

Le cryptage est une technique cruciale pour garantir la sécurité des données. Il implique l'utilisation d'algorithmes complexes pour convertir le texte brut en code illisible grâce a des clés de chiffrement, ce qui rend difficile l'accès des personnes non autorisées aux informations sensibles. [26].

1.2.1 L'importance du cryptage dans la Sécurité des Données

Le cryptage est vital pour la sécurité des donnée, car il ajoute une couche de protection supplémentaire aux informations sensibles. Sans cryptage, les données sont vulnérables à l'interception, au vol et à la modification par des personnes non autorisées, ce qui garan-

tit que seules les personnes autorisées peuvent accéder et lire les informations. De plus, il contribue également à protéger contre la falsification des données, qui peut compromettre leur intégrité. Le cryptage est donc un outil essentiel pour protéger les informations sensibles.

Un exemple illustrant cette importance est En 2015, Anthem Inc, l'un des plus grands assureurs maladie aux États-Unis, a été victime d'une violation de données au cours de laquelle les dossiers personnels de près de 78,8 millions de personnes ont été compromis. La violation aurait pu être évitée ou atténuée si les données avaient été cryptées, rendant ainsi beaucoup plus difficile pour les pirates informatiques d'accéder et d'exploiter les informations volées. [27]

1.2.2 Histoire du chiffrement

Le cryptage ne date pas d'aujourd'hui puisqu'il faut remonter à la civilisation babylonienne, environ 3 000 ans avant notre ère, pour en trouver les premières traces. Quant à son application, elle s'est peu à peu étendue des seuls champs militaire et politique pour investir la sphère civile, notamment sous l'impulsion d'Internet et de l'explosion des volumes de données qui révolutionnent notre quotidien sous bien des aspects.

Au Ier siècle avant J.C., on assista à l'émergence du chiffrement de César. Cette méthode de cryptage figure parmi les plus célèbres de l'histoire. Ensuite au XVe siècle, Leon Battista Alberti développa un prototype de chiffrement par substitution polyalphabétique. La plus marquante fut celle du Français Blaise de Vigenère, aussi connue sous le nom de « chiffre de Vigenère ».

Formulé manuellement jusqu'à la fin du XIXe siècle, le décryptage des chiffrements se compliqua davantage avec l'avènement des machines de chiffrement mécanique, au début du XXe siècle. La plus célèbre d'entre elles fut sans aucun doute Enigma, une machine portable et puissante mise au point par l'ingénieur allemand Arthur Scherbius en 1918. Le décryptage d'Enigma ne fut rendu public qu'en 1974, soit plus de 30 ans après les faits.

Depuis la Seconde Guerre mondiale, le cryptage et le décryptage sont passés du mécanique au numérique avec l'algorithme DES (Data Encryption Standard) approuvé par le NBS, qui devint la méthode de chiffrement standard dans le monde entier. Toutefois, ce dernier fut finalement cassé en 1994.

En 1977, la méthode de cryptographie RSA a été inventée par Ron Rivest, Adi Shamir et Len Adleman, à la suite de la découverte de la cryptographie à clé publique par Diffie et Hellman, qui est devenue un système universel.

En 2000, le NIST a accepté l'algorithme Rijndael et l'a baptisé AES (Advanced Encryption Standard). Aujourd'hui, l'AES est une norme largement acceptée et utilisée pour

le chiffrement symétrique.

Comme cette chronologie nous le rappelle, l'histoire de la cryptographie est un éternel recommencement d'inventions d'algorithmes, de cassage de ces codes, puis d'apparition de nouveaux algorithmes renforcés. [20]

1.3 Les principes fondamentaux du cryptage

1.3.1 L'Objectifs du cryptage

Le cryptage résout quatre problèmes majeurs :

— La confidentialité : garantir que le message chiffré ne peut être lu que par les destinataires légitimes empêchant ainsi tout accès non autorisé.

— L'authentification : certifier l'origine du message auprès du destinataire. Afin d'éviter toute usurpation d'identité.

— L'intégrité des données : permettre au destinataire de vérifier que le message n'a pas été modifier en cours de route cela évite à un intrus d'échanger un faux message en faisant croire qu'il est authentique.

— La non-répudiation : empêcher toute personnes ou entité de nier ses actes que ce soit écriture, envoie ou réception du message plus tard. [28]

1.3.2 Les types du cryptage

Les différents types de cryptographie incluent la cryptographie symétrique, asymétrique et les fonctions de hachage .

1.3.2.1 Le cryptage symétrique

Le cryptage symétrique utilise la même clé secrète pour le cryptage et le décryptage des données. On trouve le DES et l'AES Parmi les exemples de ce type.

Ce dernier est souvent utilisé pour des opérations de cryptage simples, telles que la sécurisation des communications par courrier électronique entre deux personnes. [16, 27]

FIGURE 1.1 – Cryptographie symétrique [5]

La figure ci-dessus illustre le principe du cryptage symétrique fonctionnant avec une clé secrète pour le chiffrement et le déchiffrement du texte clair.

1.3.2.2 Le cryptage asymetrique

Le cryptage asymétrique utilise deux clés distinctes à la fois une combinaison d'une clé publique et d'une clé privée. La clé publique, comme son nom l'indique, est mise à la disposition de tous afin de convertir, l'information originale en un format codé. Tandis que la clé privée est gardée secrète par le propriétaire pour décoder l'information. Un exemple courant de cette méthode de cryptage est la sécurisation des transactions en ligne. [21,27]. La figure 1.2 représente le principe du cryptage asymétrique avec une clé publique pour le chiffrement et une clé privé pour le déchiffrement.

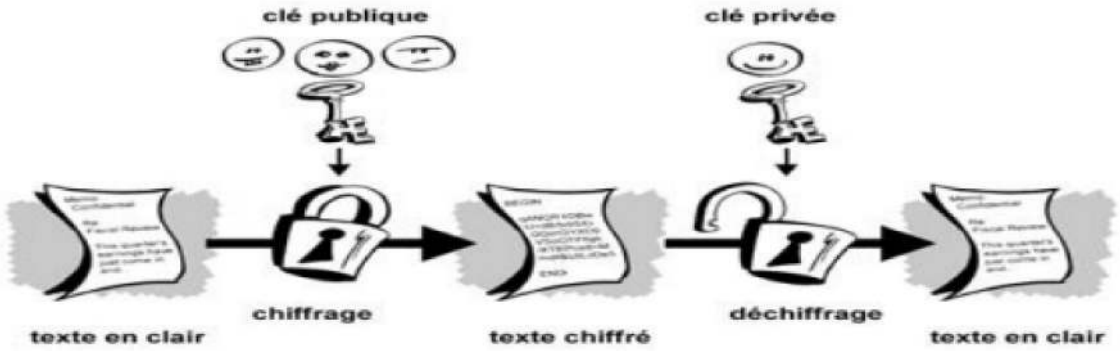

FIGURE 1.2 – Chiffrement asymétrique [4]

1.3.2.3 Les fonctions de hachage

Les fonctions de hachage transforment les données en une chaîne alphanumérique de taille fixe, dites valeur de hachage. Contrairement à la cryptographie symétrique et asymétrique, c'est des fonctions irréversibles qui ne peuvent pas être déchiffrées vu qu'elles n'utilisent pas de clé cryptographique.

Ces dernières sont principalement utilisés pour vérifier l'intégrité des données. [15, 27, 28]

la figure ci-dessous représente le principe du fonctionnement de hachage avec un texte clair en entré et une empreinte numérique en sortie.

FIGURE 1.3 – Fonction de hachage [29]

1.3.3 Les méthodes de chiffrement

1.3.3.1 Le Chiffrement par substitution

C'est une méthode de chiffrement dans laquelle chaque lettre du message clair est substitué par une autre lettre différente dans le texte chiffré comme le montre la figure 1.4 qui représente un exemple du chiffrement par substitution ou la lettre [4]

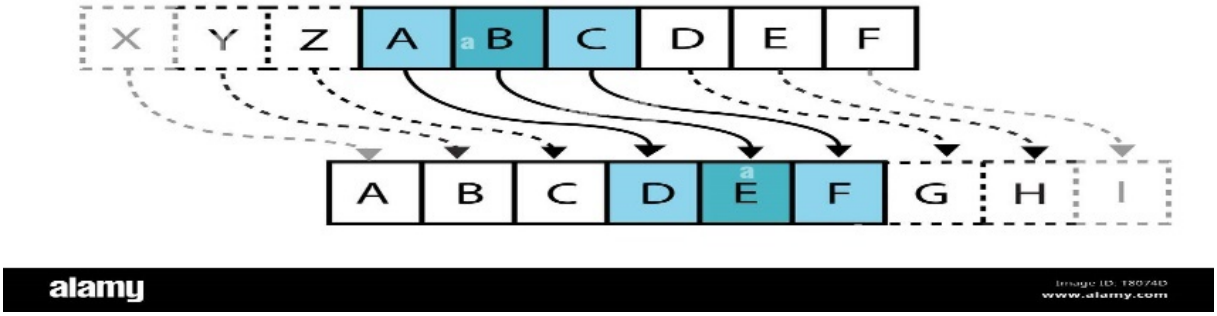

FIGURE 1.4 – Exemple de chiffrement par substitution [2]

1.3.3.2 Le chiffrement par transposition

Ce chiffrement consiste à réorganiser les lettres du message clair suivant des règles déterminées par la clé de chiffrement. On peut prendre pour exemple le réarrangement géométrique des données représenter dans la figure1.5. [18, 21]

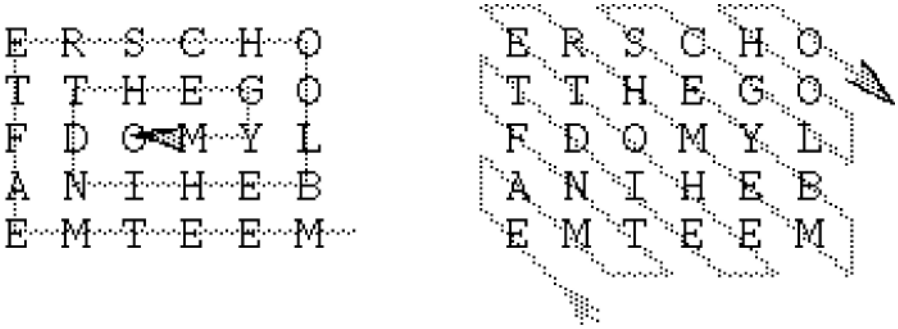

FIGURE 1.5 – Exemple de chiffrement par transposition [3]

1.3.3.3 Le chiffrement par blocs

Dans un système par blocs, chaque texte clair est découpé en blocs de données de taille fixe, généralement de 64 ou 128 bits, ensuite il est chiffré bloc par bloc comme le montre la figure 1.6 qui représente un exemple de chiffrement par bloc. [18, 25]

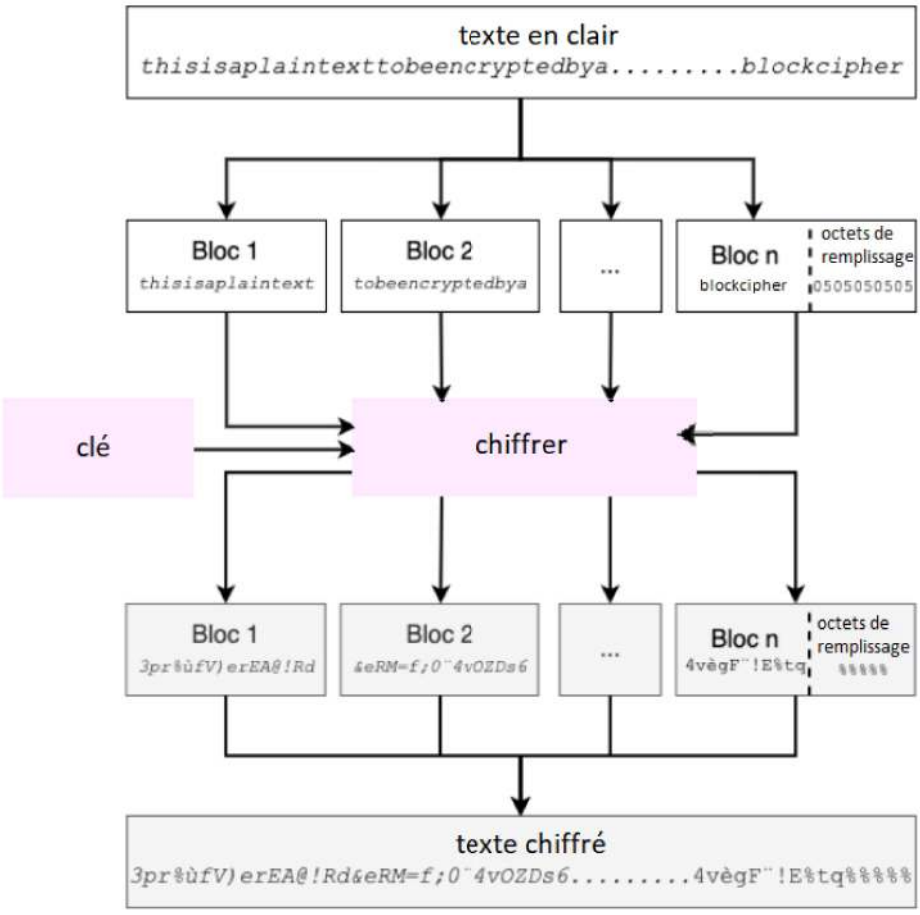

FIGURE 1.6 – Exemple de chiffrement par blocs [1]

1.3.3.4 Le chiffrement par flux

Autrement dit chiffrement par flot, c'est une méthode de cryptage qui chiffre les données du texte clair bit par bit au fur et à mesure qu'il se présente, en utilisant un flux de clés, qui est un jeu de caractères aléatoires pour remplacer ceux du texte ordinaire, et peut inclure des caractères alphabétiques, numériques, et des symboles ou caractères spéciaux. Le processus de chiffrement et de déchiffrement se fait en XOR le texte brut et le flux de clés. [18, 26]

La figure ci-dessous illustre le processus du chiffrement par flot

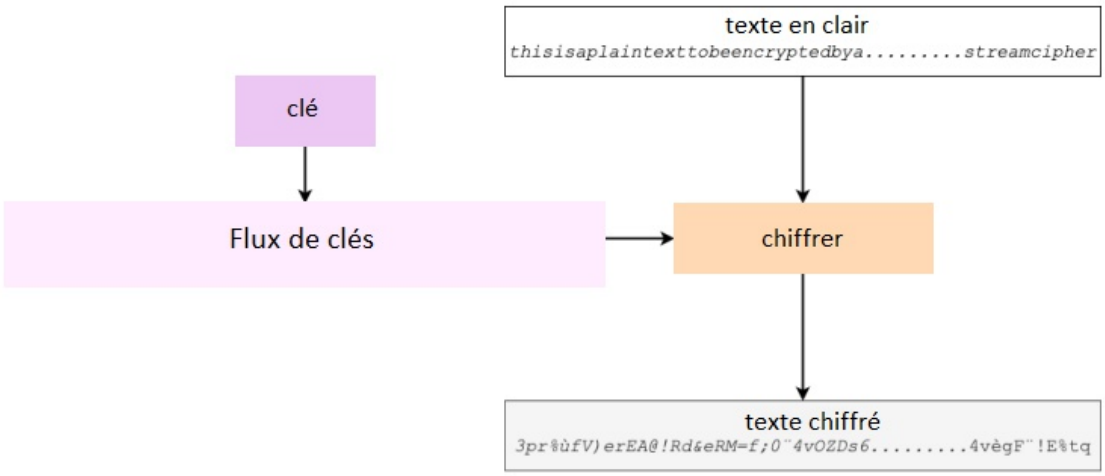

Figure 1.7 – Exemple de chiffrement par flux [1]

1.4 Cryptographie classique

1.4.1 Origine de la cryptographie classique

La cryptographie classique, trouve ses origines en Rome antique, avec le tout premier code de chiffrement connu, le code de César. Elle décrit la période avant les ordinateurs et a persisté jusqu'à la fin des années 1970, durant cette période un grand nombre de systèmes de chiffrement ont été inventés tel que le chiffre de Vigenère et la machine à rotors Enigma. Toutefois la cryptographie classique céda sa place avec l'avènement de la cryptographie moderne, qui utilise des algorithmes mathématiques complexes pour assurer la sécurité des communications. [18, 20]

1.4.2 Principes de la cryptographie classique

Les cryptographies classiques consistaient à faire subir à un texte clair une transformation plus ou moins complexe pour en déduire un texte inintelligible, dit chiffré. En traitant des systèmes basés sur les lettres ou caractère d'une langue naturelle (allemand, anglais, français, etc.). Les différents algorithmes cryptographiques substituaient,les caractères par d'autres ou les transposaient dans des ordres différents. La figure 1.8 représente une classification des méthodes de cryptographie classique en deux types substitution et transposition. [25]

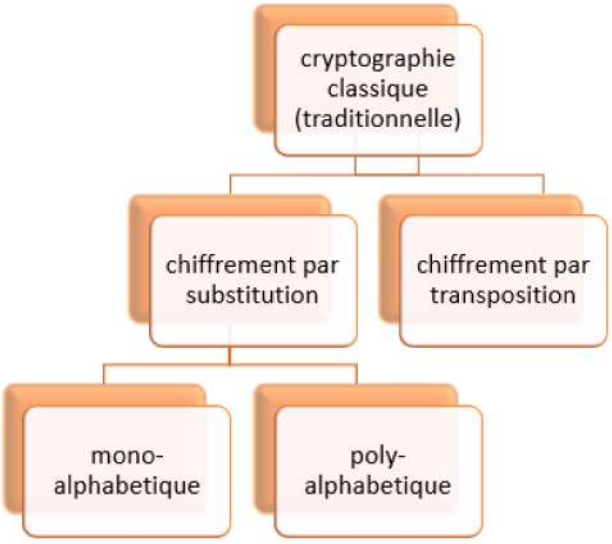

FIGURE 1.8 – Cryptographie classique

Le chiffrement de César et le chiffrement de Vigenère illustrent des exemples de chiffrement par substitution largement employés dans le passé .

1.4.2.1 Le chiffrement de César

Le chiffrement attribué à Jules César est une méthode de chiffrement par substitution où les lettres sont décalées d'un certain nombre de positions dans l'alphabet afin de chiffrer un message, tandis que le déchiffrement revient à décaler dans le sens inverse. En prend pour exemple le ROT13 un exemple particulier du chiffrement de César, en utilisant un décalage de 13 lettres, comme le montre la figure1.9 qui représente le chiffrement du mot "HELLO" avec un décalage de 13 position. [11, 14, 18]

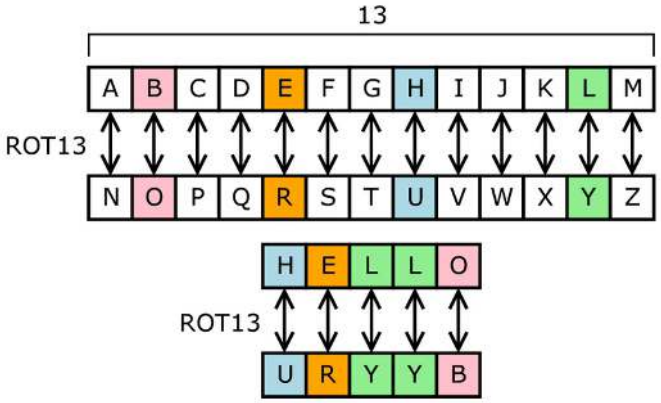

FIGURE 1.9 – Exemple du chiffrement de César [7]

1.4.2.2 Le chiffrement de Vigenère

Le chiffrement de Vigenère est un chiffrement symétrique utilisant une substitution poly-alphabétique, c'est-à-dire que la lettre d'origine peut être remplacée par plusieurs lettres (cela dépendra de sa place dans le message, mais aussi de la clé utilisée).

La clé de chiffrement est une chaîne de caractères, que l'on répété autant de fois que nécessaire afin d'avoir la même longueur que le message à crypté tel que l'illustre la figure 1.10 qui représente un exemple du chiffrement en utilisant la clé "code" .

Ce chiffrement est une amélioration décisive du chiffre de César, sa force réside dans l'utilisation non pas d'un, mais de 26 alphabets décalés pour chiffrer un message. [**?**,11,15]

M	E	S	S	A	G	E	S	E	C	R	E	T
+	+	+	+	+	+	+	+	+	+	+	+	+
C	O	D	E	C	O	D	E	C	O	D	E	C
=	=	=	=	=	=	=	=	=	=	=	=	=
O	S	V	W	C	U	H	W	G	Q	U	I	V

FIGURE 1.10 – Exemple du chiffrement de vigenère [12]

1.4.2.3 La machine Enigma

La machine Enigma est une machine électromécanique, reposant sur un système de rotors et de contact électrique qui comme le montre la figure 1.11 se présente comme une machine à écrire n'ont pas de message ordinaire mais de message codé. [20]

FIGURE 1.11 – La machine Enigma [8]

Le fonctionnement d'Enigma repose sur les éléments suivants :

— Des rotors : le rotor tourne après chaque frappe donc Une lettre se répète après 26 tours effectué par ce dernier c'est la raison pour laquelle la machine Enigma utilise 3 rotors à chaque fois, qu'un rotor complète son tour le suivant bouge d'un cran ce qui rend l'attaque par analyse de fréquence difficile.

— Un tableau de connexion qui agit comme brouilleur du cryptage en utilisant des fiches de connexion de deux lettres.

— Un réflecteur qui permet de renvoyer le courant à nouveau dans les rotors ,ce qui permet de faciliter le déchiffrement.

— Un tableau lumineux pour illuminé la lettre chiffre.

Lorsque l'opérateur tape une lettre le courant passe par ses éléments pour afficher la lettre chiffrée. La figure ci-dessous expose le fonctionnement de la machine historique Enigma. [5, 20, 25]

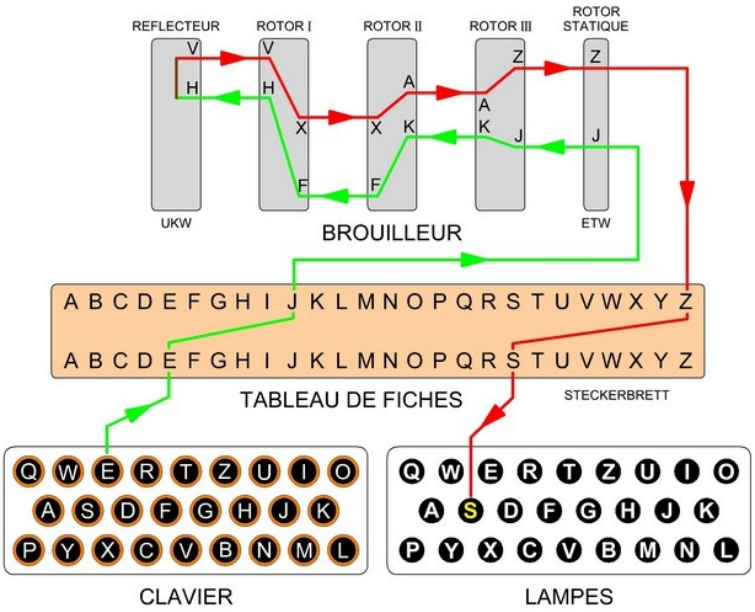

Figure 1.12 – Fonctionnement de la machine Enigma

1.4.3 Les faiblesses de la cryptographie classique

La cryptographie traditionnelle a été confrontée à plusieurs limitations, parmi ces faiblesses, on peut citer :

- Insuffisante sécurité : les algorithmes étaient relativement simples et peu sûrs, exposant ainsi les messages à des attaques de cryptanalyse tel que l'analyse fréquentielle.
- Gestion des clés complexe : l'échange de clé sécurisé entre les parties, était complexe à réaliser.
- Limitation des longueurs de clé : les chiffrements classiques étaient limités en termes de longueur de clé, les rendant ainsi moins robustes face aux attaques modernes.
- Absence de confidentialité : la sécurité des messages chiffrés reposait sur la confidentialité des clés utilisées, Une seule clé compromise peut nuire à tous les messages chiffrés par celle-ci. [9]

1.5 Cryptographie moderne

1.5.1 Développement de la cryptographie moderne

Les faiblesses de la cryptographie classique combiné à l'invention de l'ordinateur ont propulsé l'essor de l'avènement de la cryptographie modernes. Cette transition a souligné le passage des simples algorithmes de substitution aux algorithmes de cryptographie

modernes tels que le DES, l'AES et le RSA qui reposent sur des principes et des problèmes mathématiques sophistiqués difficiles à résoudre sans les clés cryptographiques appropriées. [9]

1.5.2 Les algorithmes de cryptographie moderne

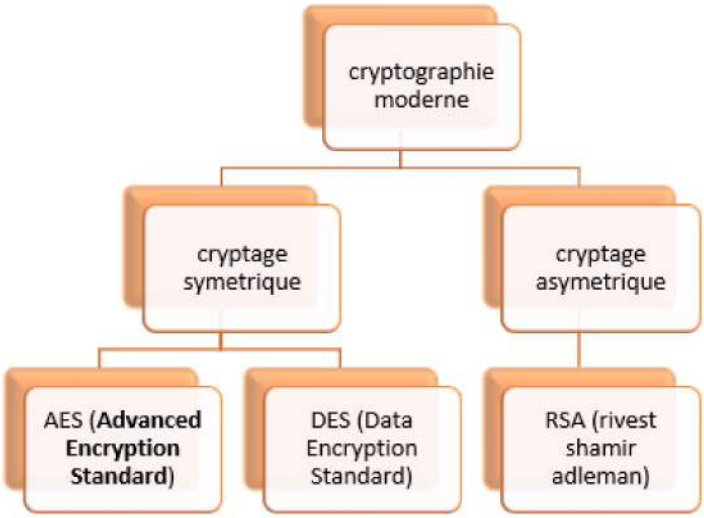

FIGURE 1.13 – La cryptographie moderne

La figure ci-dessus expose les deux principaux types de cryptographie moderne soit symétriques et asymétriques ainsi que leur algorithmes.

1.5.2.1 Data Encryptions Standard (DES)

Le DES est l'ancien standard de chiffrement des données sensibles non secrètes, il s'agit d'un algorithme de chiffrement par blocs qui convertit le texte clair en blocs de 64 bits et les transforment en texte chiffré en utilisant une clé secrète. Celle-ci est généralement représentée sous la forme d'un nombre de 64 bits, mais seuls 56 bits servent réellement à la définir car un bit par octet est utilisé pour le contrôle de parité. Comme le montre la figure 1.14 qui illustre le principe de fonctionnement du DES. [18, 21, 24]

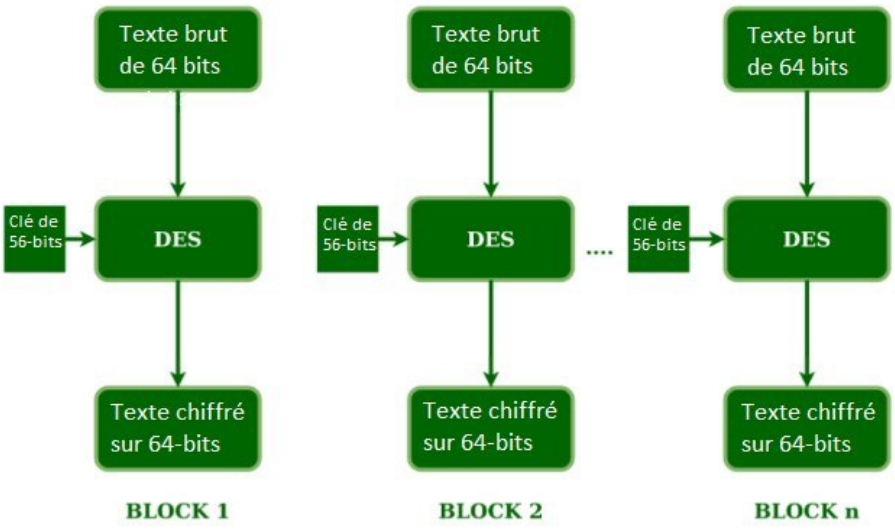

FIGURE 1.14 – Fonctionnement du DES [6]

1.5.2.2 Advanced Encryptions Standard (AES)

L'AES est la norme standard recommandé pour le chiffrement à clé secrète qui repose sur le chiffrement par bloc utilisant des clés de 128, 192 et 256 bits, offrant ainsi différents niveaux de sécurité selon la longueur de la clé ce qui le révèle plus performant par rapport au DES. Les clés de 128 bits conviennent à la plupart des applications. Tandis que celles de 192 et 256 bits sont utilisées pour le chiffrement à haut rendement. [3, 18, 23]

La figure ci-dessous reflète un exemple de chiffrement en utilisant l'AES.

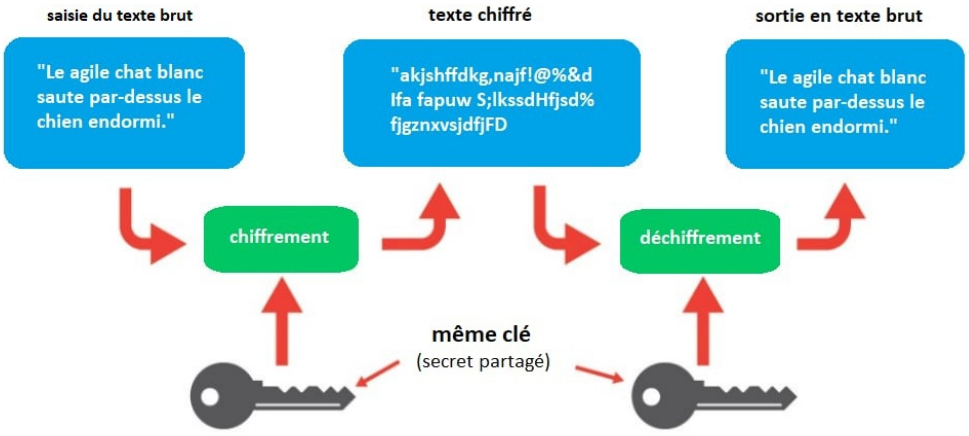

FIGURE 1.15 – Principe de chiffrement de l'AES [11]

1.5.2.3 Rivest Shamir Adleman (RSA)

Le RSA est un algorithme largement utilisé pour le chiffrement des données avec une clé publique et leur déchiffrement avec une clé privée dans le but de sécuriser la transmission des données comme le montre la figure 1.16 qui explore le principe de fonctionnement du RSA où seule la clé privée du destinataire peut décrypter le message crypté. [18, 29]

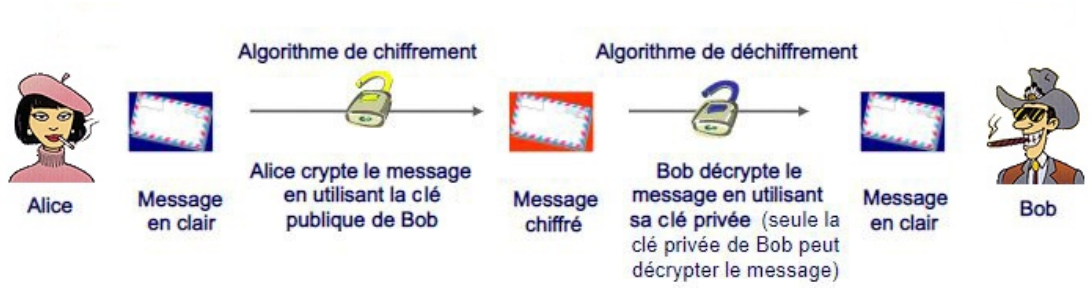

Figure 1.16 – Principe du chiffrement RSA [10]

Il est également utilisé pour la signature des messages contrairement au chiffrement, le message est signé avec la clé privé et cette signature est vérifié avec la clé publique comme l'indique la figure ci dessous qui reflète le principe de signature du RSA [22, 24]

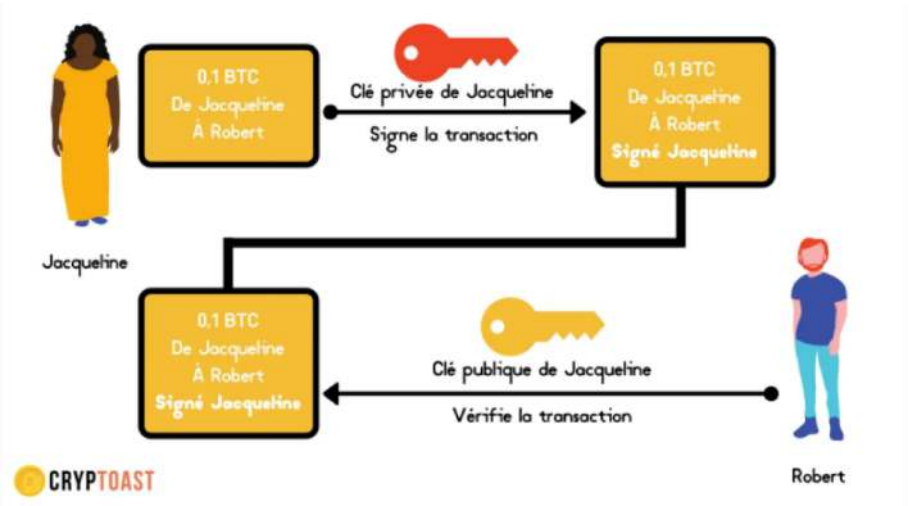

Figure 1.17 – Signature du RSA

La sécurité du RSA repose sur la difficulté de factoriser des grands nombres premiers. [24]

1.5.3 Les protocoles de securité

Un protocole de sécurité est un ensemble de règles mise en place pour assurer certains objectifs de sécurité. Principalement la confidentialité, l'intégrité et l'authenticité des données transmises sur Internet. Les protocoles les plus utilisés sont SSL/TLS, IPSec et SSH. [13]

1.5.3.1 SSL/TLS

Secure Socket Layer (SSL) et Transport Layer Security (TLS) sont des protocoles de sécurité les plus utilisés sur internet pour une connexion réseau sécurisé, ils permettent d'établir des liens cryptographiques entre un serveur web et un navigateur. Pour assurer l'intégrité et la confidentialité des données échangées. Ce protocole est essentiel pour sécuriser les transactions en ligne et les connexions bancaires. [13]

1.5.3.2 IPsec

IPSEC Dans le terme "IPsec," "IP" signifie "Internet Protocol" et "sec" pour "sécure". C'est un protocole de sécurité qui assure la sécurisation de la communication à la couche réseau du modèle OSI. Il exploite diffèrent types d'algorithmes de chiffrement et de service d'authentification. IPSec est souvent utilisé pour sécuriser les réseaux privés virtuels (VPN) et les connexions d'accès à distance. [13]

1.5.3.3 SSH

Le protocole SSH (Secure Shell) est un protocole qui permet d'établir une connexion sécurisée entre un client et un serveur,et qui fournit un accès à distance sécurisé aux périphériques réseau. Il repose sur l'utilisation de mot de passe de clés publiques et d'authentification à deux facteurs pour authentifier les utilisateurs et chiffrer les données. [13]

Ces protocoles de sécurité jouent un rôle crucial dans la cryptographie en fournissant des mécanismes de sécurité pour garantir la confidentialité, l'intégrité et l'authenticité des informations échangées sur les réseaux informatiques. [13]

1.6 Description du projet

Notre projet vise à concrétiser les concepts théoriques de la cryptographie en implémentant une interface graphique sur Python. permettant a l'utilisateur d'explorer et de tester les diffèrent algorithmes de chiffrement, allant des méthodes classiques comme César, Vigenére et la substitution jusqu'aux approches modernes tel que le DES, l'AES et le

RSA et en passant aussi par la machine historique Enigma. cette interface comprendra des champs de saisie pour le texte en clair, la clé de chiffrement et pour d'autre paramètres spécifiés a certains algorithmes. De plus elle seras munis de boutons dédiés au opérations de chiffrement et de déchiffrement mais aussi de boites de dialogues concernant les messages d'erreur des clés relatives au algorithmes exigeant.les résultat des opérations de cryptage seront affiché dans des champs dédiés. Grâce à cette application, nous souhaitons rendre la cryptographie plus accessible et compréhensible pour tous en expérimentant les divers algorithmes présent et en observant leur fonctionnement en temps réel.

1.7 Conclusion

Ce chapitre offre une vue d'ensemble complète du domaine de la cryptographie, nous avons exploré les fondements du cryptage en soulignant son importance dans la sécurité des données et en détaillant son histoire et ses objectifs. Nous avons examinés en détail, les types de cryptage symétrique et asymétrique ainsi que les méthodes de chiffrement classiques et modernes que nous allons voir dans le chapitre 2 consacré au mécanismes techniques et mathématiques des différents algorithmes de chiffrements.

Chapitre 2

Mécanismes de chiffrement et de déchiffrement

2.1 Introduction

Le chapitre précédent été consacré aux généralités et aux principes de base du cryptage et des types de ce dernier, tandis que ce chapitre explore le fonctionnement technique et mathématique des processus de chiffrement et de déchiffrement des algorithmes cryptographiques, ainsi que la génération de clés pour certains allant des systèmes symétriques, comme le chiffrement de César et Vigenère aux avancées historiques comme la machine Enigma, en passant par des algorithmes modernes tels que le DES et l'AES, jusqu'à la cryptographie asymétrique avec l'algorithme RSA.

Nous détaillerons aussi les caractéristiques et les faiblesses de ces derniers, qui ont mené a l'évolution et au développement des méthodes cryptographique.

2.2 Les algorithmes de cryptographie symétrique

2.2.1 Chiffrement par substitution

2.2.1.1 Concept de substitution mono-alphabétique

La substitution mono-alphabétique, aussi connu sous le nom d'alphabets désordonnés, est l'une des méthodes les plus simples à réaliser, il s'agit de remplacer chaque lettre du message clair par une autre, Il existe ainsi un seul alphabet appelé alphabet clé. Ce qui fait que toutes les occurrences d'une même lettre sont codées de la même manière. [17]

2.2.1.2 Technique de chiffrement

La figure 2.1 illustre le processus de chiffrement par substitution mono-alphabétique. On commence par lire le texte clair que l'on souhaite chiffrer ensuite une clé de substitution est généré aléatoirement pour effectuer le chiffrement. Cette clé est utilisée pour substituer chaque lettre du texte clair par son équivalent dans celle-ci. Par exemple, si la clé de substitution est "BDFHJLNPRTVXZACEGIKMOQSUWY", alors la lettre "A" sera remplacée par "B", "B" par "D", et ainsi de suite pour afficher a la fin le message chiffré composé des lettres substituées.

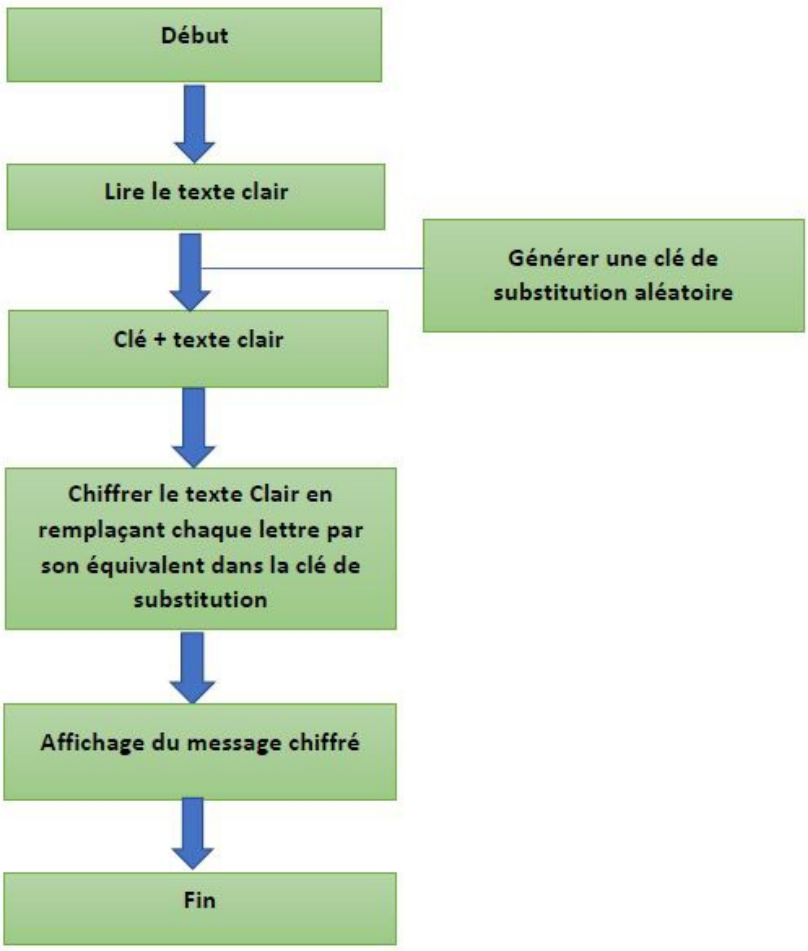

FIGURE 2.1 – Organigramme du chiffrement par substitution mono-alphabetique

2.2.1.3 Technique de déchiffrement

La figure 2.2 représente le processus de déchiffrement par substitution mono-alphabétique il s'agit du processus inverse du chiffrement. On commence par lire le texte chiffré au quel nous allons soustraire la clé de substitution utilisée lors du chiffrement initial pour rem-

placé chaque lettre du texte chiffré par son équivalent dans celle-ci pour afficher ensuite Le message clair.

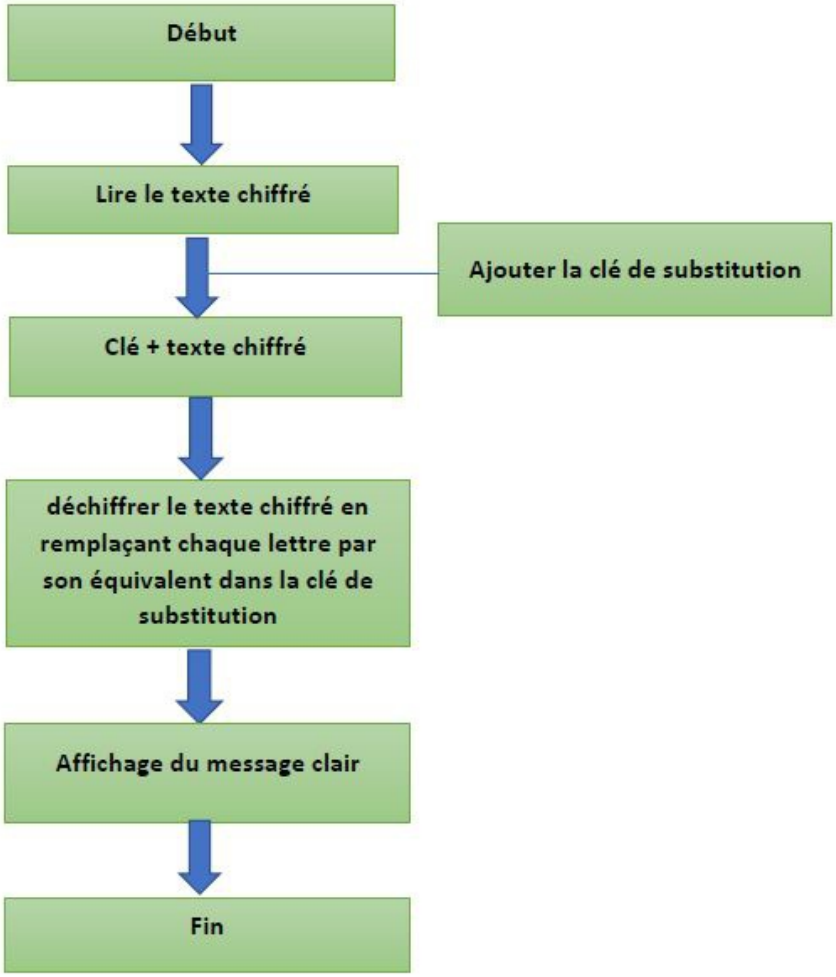

FIGURE 2.2 – Organigramme du déchiffrement par substitution mono-alphabetique

2.2.1.4 Faiblesses et contre-mesures

Le chiffrement par substitution mono-alphabétique est sujet a plusieurs attaques de cryptanalyse telles que :

- L'analyse de fréquence : les langues naturelle tel que le français ont une fréquence d'apparition des lettres non uniforme pouvant ainsi identifier les lettres les plus fréquente ainsi que leur distribution dans le texte chiffré.
- Motifs répétitifs : la récurrence de mot dans le texte clair se traduit par des motifs répétitifs dans le texte chiffré facilitant ainsi le cassage du code.
- L'ttaques par force brute : une même lettre est toujours chiffré de la même façon vu qu'on utilise un alphabet de substitution unique et fixe, donc il n'y a que 26

combinaison de clés possible.

— L'attaque par dictionnaire : utilisation des mots courant contenu dans un dictionnaire pour essayer de deviner le texte originale.

Pour contrer les faiblesses et les attaques du chiffrement mono-alphabétique il est recommandé d'utiliser des méthodes de chiffrement plus modernes et plus robustes tels que le chiffrement par blocs comme l'AES, par flux ou a clé publique RSA [19, 25].

2.2.2 Chiffrement de César

2.2.2.1 Mécanisme de chiffrement

La figure 2.3 représente l'organigramme du chiffrement de César. On commence par lire le texte clair pour le convertir en nombres (ex : A = 1, B = 2) puis on choisis le décalage qui représente la clé pour l'appliquer par la suite au texte clair convertit en nombres selon cette équation : soit (n) le décalage choisi, alors pour chaque lettre (L) dans le texte original " lettres chiffrés $= (L + n) mod 26$ ".

Alors chaque lettres est décalé d'un nombre de positions fixe produisant ainsi un texte chiffré en nombre que l'on reconverti en lettres pour obtenir le texte chiffré final.

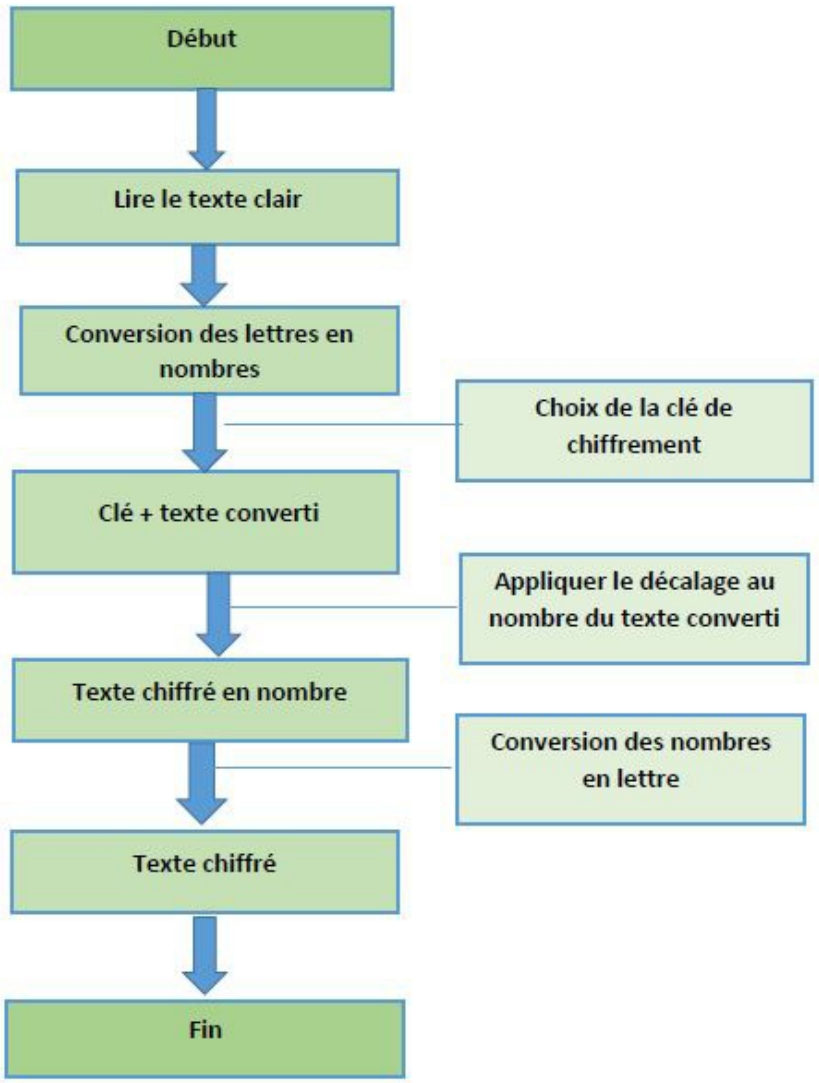

FIGURE 2.3 – Organigramme du chiffrement de César

2.2.2.2 Mécanisme de déchiffrement

La figure 2.4 expose l'organigramme de récupération du message original. On lit le texte chiffré que l'on convertit en nombres selon la valeur numérique de chaque lettres (ex : D = 4) ensuite on applique le décalage inverse en utilisant la même clé que celle du chiffrement selon l'équation suivante : "lettres déchiffrés $= (L' - n) mod 26$ ".

Où L' représente chaque lettre chiffrée et n est le décalage utilisé pour chiffrer. On soustrait ainsi la clé du nombre représentant chaque lettre chiffrée. Ensuite Les nouveaux nombres sont convertis en lettres pour former le texte clair déchiffré.

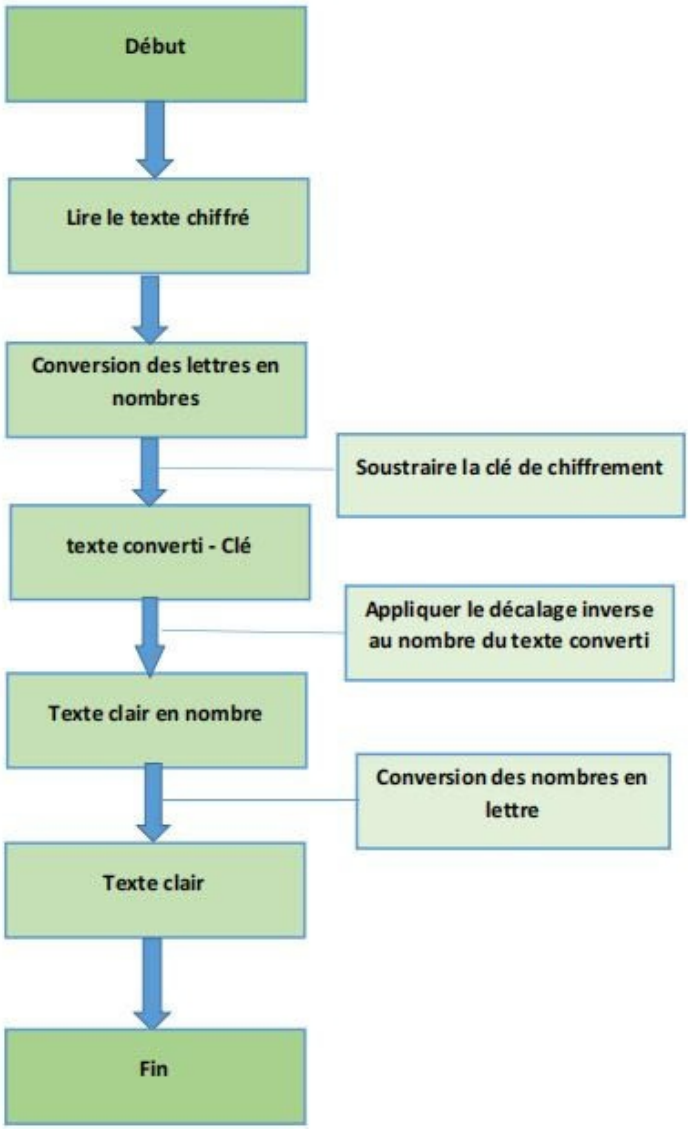

FIGURE 2.4 – Organigramme du déchiffrement de César

2.2.2.3 Limitation et vulnérabilités technique

Malgré son utilité historique le chiffre de césar manque de fiabilité en terme de sécurité pour deux raison d'une part, le nombre limités de clés, pour 26 lettres d'alphabet il existe 25 décalage possible et d'autre part la lettre est toujours chiffré de la même façon. Ce qui le rend vulnérable aux attaques par force brute et a l'analyse de fréquence. [11, 14, 26]

2.2.3 Chiffrement de vigenere

2.2.3.1 Fonctionnement technique du chiffrement poly-alphabetique

Le principe du chiffrement poly-alphabétique implique l'utilisation d'une série de substitutions mono-alphabétiques réutilisées périodiquement. Chaque lettre du texte clair est substituée par une autre pouvant correspondre a plusieurs lettres différentes choisi parmi un ensemble d'alphabets aléatoires associés en fonction de sa position dans le message. [26]

2.2.3.2 Processus de chiffrement

La figure 2.5 illustre l'organigramme du chiffrement de Vigenère, une méthode plus complexe que César. Le processus de chiffrement de Vigenére se fait selon les étapes suivantes : On commence par lire le message en clair et la clé secrète, que l'on convertit en nombres ensuite on ajuste la clé en la répétant pour couvrir tout le message pour chiffré chaque lettre du message en fonction de la lettre correspondante de la clé répétée. A la fin on convertit le texte obtenue en nombres pour afficher le texte chiffré

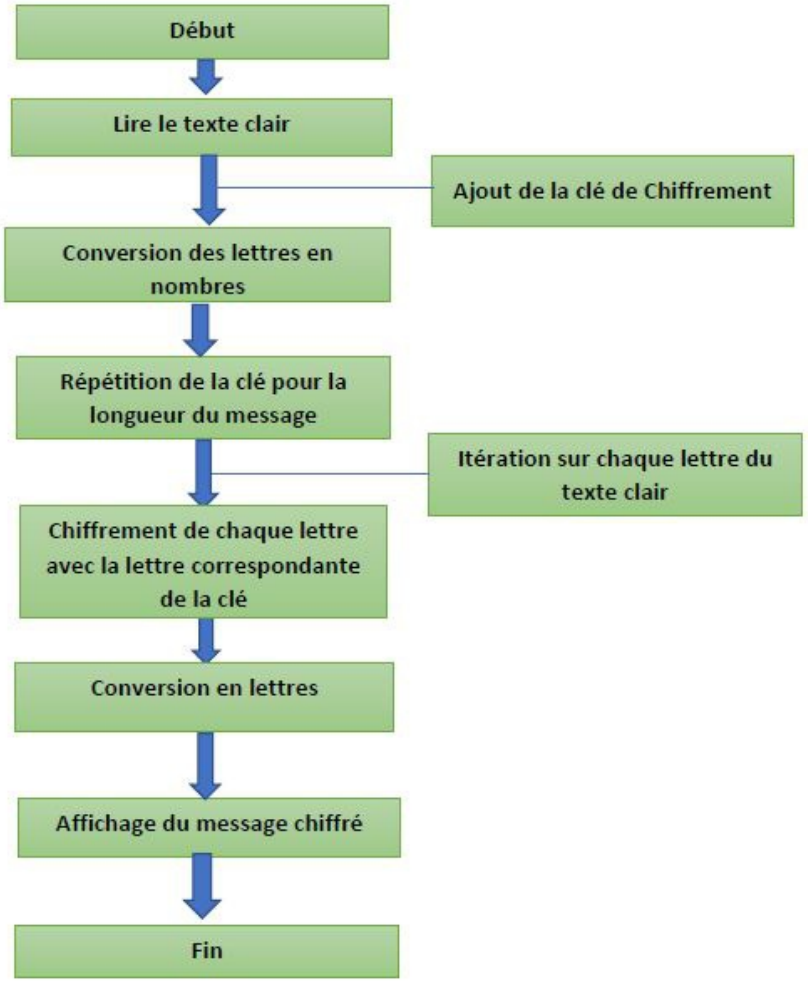

FIGURE 2.5 – Organigramme du chiffrement de vigenére

2.2.3.3 Processus de déchiffrement

Concernant le déchiffrement il se fait en appliquant le processus inverse tel que le montre la figure 2.6 qui représente l'organigramme de déchiffrement de Vigenére. On utilise le processus inverse soit le même processus mais avec un message chiffré en entré et une substitution poly-alphabétique inverse, chaque lettre chiffrée est remplacée par sa lettre correspondante du bloc de clé correspondant, pour retrouver la lettre du message en clair.

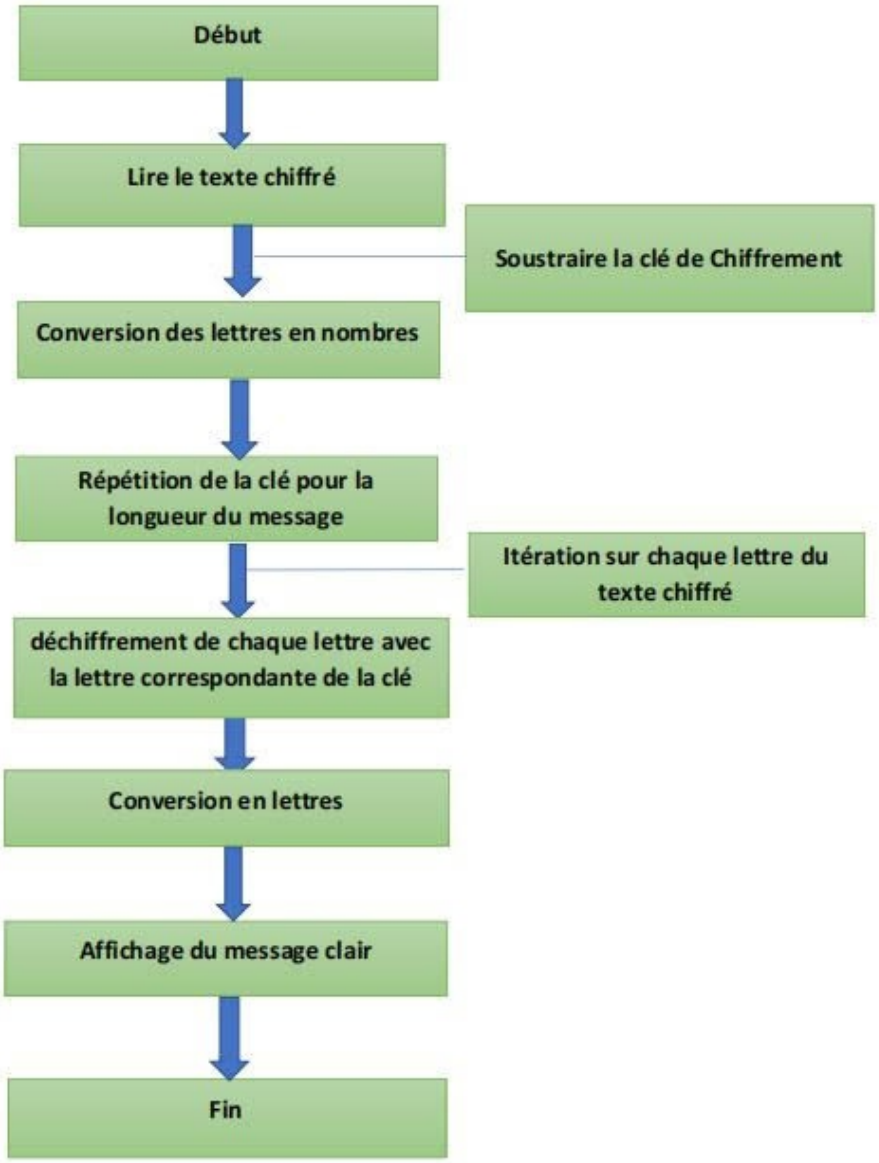

FIGURE 2.6 – Organigramme de déchiffrement de Vigenére

2.2.3.4 Attaques et méthodes de cryptanalyse

Bien que le chiffre de Vigenère fut considéré comme inviolable pendant trois siècle, il est désormais facile d'en venir à bout dès lors que le texte chiffré est assez long. Deux grandes approches se démarquent dans cette discipline la méthode historique et la méthode moderne.

L'approche historique est fondée sur des techniques ancestrale tel que les méthodes de Kasiski et Babbage qui utilisent la répétition d'une séquence de 3 lettres au sein du texte chiffré pour calculer la longueur de la clé de chiffrement puis la déterminent avec l'analyse de fréquence.

Tandis que la méthode moderne repose sur des techniques plus récentes telles que L'indice de coïncidence dont le concept est assez simple, il s'agit de calculer la probabilité d'obtenir des lettres identiques en tirant aléatoirement deux lettres d'un texte ce qui permet d'identifier à la fois la méthode de substitution utilisé mono ou poly-alphabétique pour chiffré le message ainsi que la longueur probable de la clé utilisée. [11, 15, 26]

2.2.4 Machine de chiffrement historique Enigma

2.2.4.1 Fonctionnement et configuration

La figure 2.7 représente l'organigramme de chiffrement avec la machine Enigma illustrant ainsi le fonctionnement et la configuration de cette machine.

On commence par configuré les rotors qui déterminent la substitution des lettres ainsi que le tableau de connexion qui permute les paires de lettres, pour ensuite saisir la lettre du message qui passe en premier lieu par le tableau de connexion pour vérifier les permutations possibles.

La sortie de cette lettre passe à travers les rotors pour une substitution poly-alphabétique et par le réflecteur pour un brouillage supplémentaire.

La lettre sortie repasse à travers les rotors dans l'ordre inverse pour passer ensuite par le tableau de connexion pour afficher par la suite la lettre résultante sur le tableau lumineux.

Le processus est répété pour chaque lettre du message, aboutissant au message chiffré

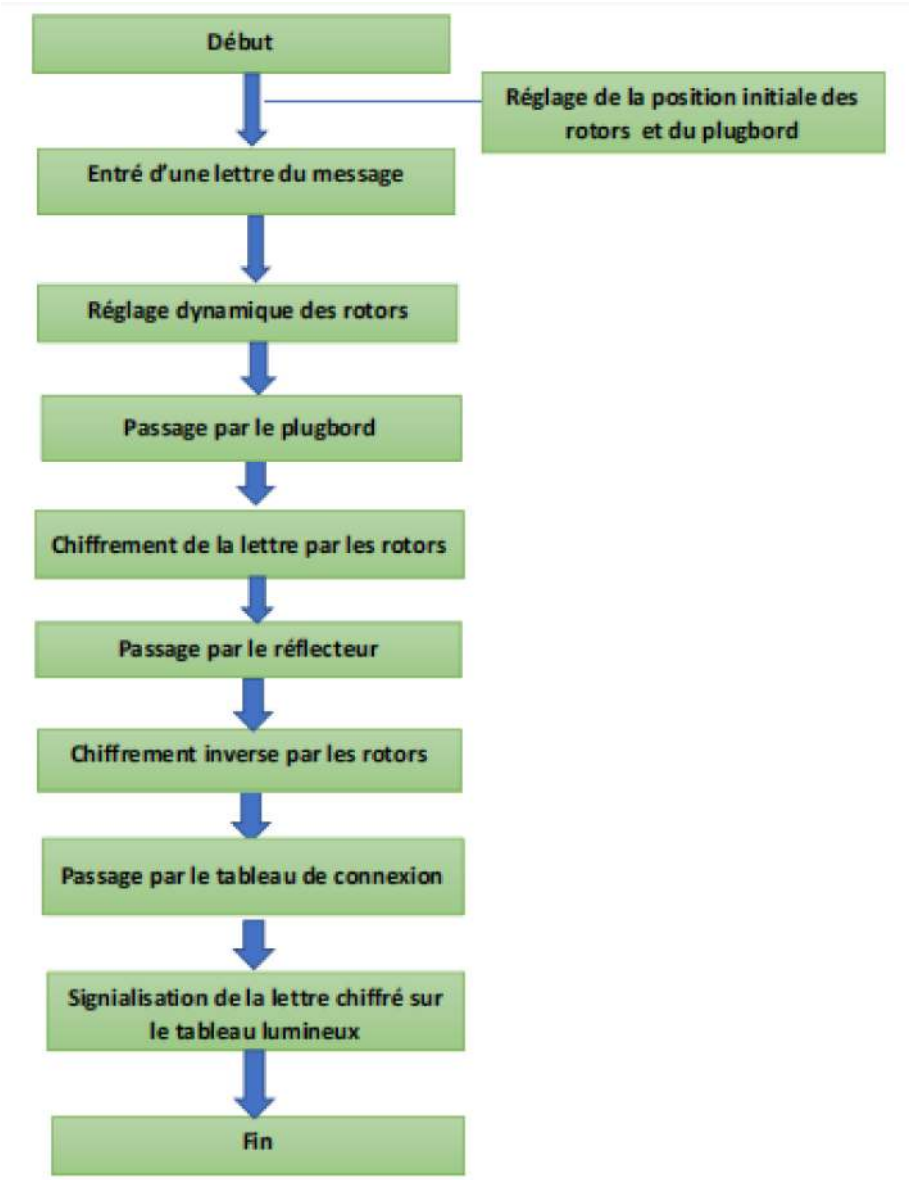

FIGURE 2.7 – Organigramme du chiffrement de la machine Enigma

2.2.4.2 Contributions a la cryptanalyse moderne et son impact sur la guerre

La machine de chiffrement allemande Enigma a eu un impact innovant sur la cryptanalyse moderne durant la Seconde Guerre mondiale. Grâce aux efforts des Polonais en premier puis des Britanniques, à Bletchley Park, notamment des figures telles que Marian Rejewski et Alan Turing, que l'Enigma a été déchiffrer. Ces progrès incluent aussi l'utilisation des "Bombes", des machines électromécaniques, et l'invention du premier calculateur électronique "Colossus", ces efforts ont eu un effet décisif sur les communications contribuant ainsi a la victoire des alliés durant la guerre. [25]

2.2.5 DES(Data Encryption Standard)

2.2.5.1 Norme de chiffrement des années 70 et 80

Le DES (Data Encryption Standard), émergeant en 1970, est l'un des algorithmes de chiffrement moderne les plus répandus de son époque, proposé par l'IBM au nom de « Lucifer » puis nomée par la suite DES. Il fut standardisé par l'American National Standard Institute-USA (ANSI). C'est une norme de chiffrement américaine et même internationale, du Gouvernement américain approuvé aussi par l'armée américaine pour le chiffrement des informations de nature sensible mais non secrète. [25]

2.2.5.2 Fonctionnement global du DES

La figure ci-dessous représente le fonctionnement global du DES elle englobe le processus de chiffrement et le processus de génération des clés en parallèle indiquant que pour chaque tour correspond une sous clé

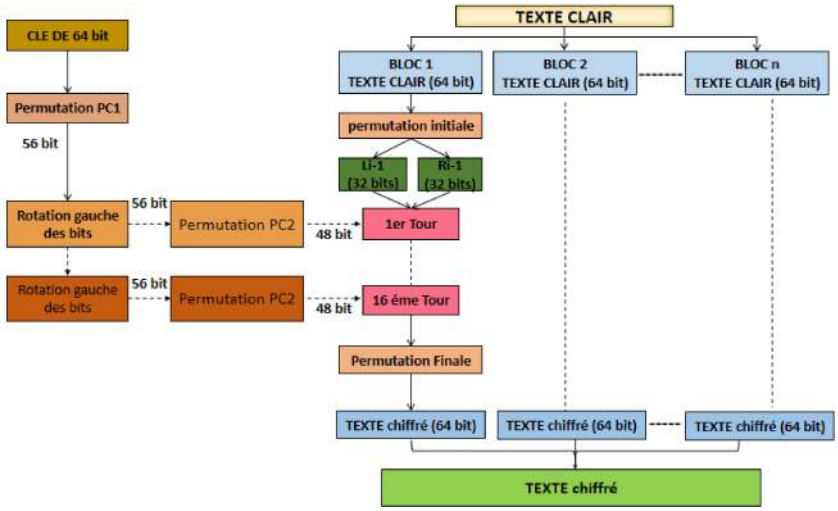

FIGURE 2.8 – Schéma du fonctionnement du DES

2.2.5.3 Processus de génération des clés du DES

La figure 2.9 représente un organigramme qui illustre les différentes étapes du processus de génération des clés dans l'algorithme de chiffrement DES (Data Encryption Standard).

Il commence par la génération des sous-clés à partir de la clé principale de 64 bits. Ensuite, la clé est convertie en binaire (64 bits) et subit une permutation PC-1 (Permuted Choice 1) selon une table prédéfinie, réduisant la clé à 56 bits.

La clé permutée est ensuite divisée en deux parties égales, R0 et L0, de 28 bits chacune. Chaque moitié (R0 et L0) subit alors une rotation gauche selon un tableau de shifts

prédéterminé.

Après la rotation, les deux moitiés sont concaténées pour former une nouvelle clé K1 de 56 bits. Cette clé subit une deuxième permutation PC-2 (Permuted Choice 2) selon une autre table prédéfinie pour obtenir la sous-clé finale K1 de 48 bits.

Ce processus est répété 16 fois pour générer les 16 sous-clés nécessaires pour l'algorithme DES. Les tables PC-1, PC-2 et le tableau des shifts sont fournis pour référence.

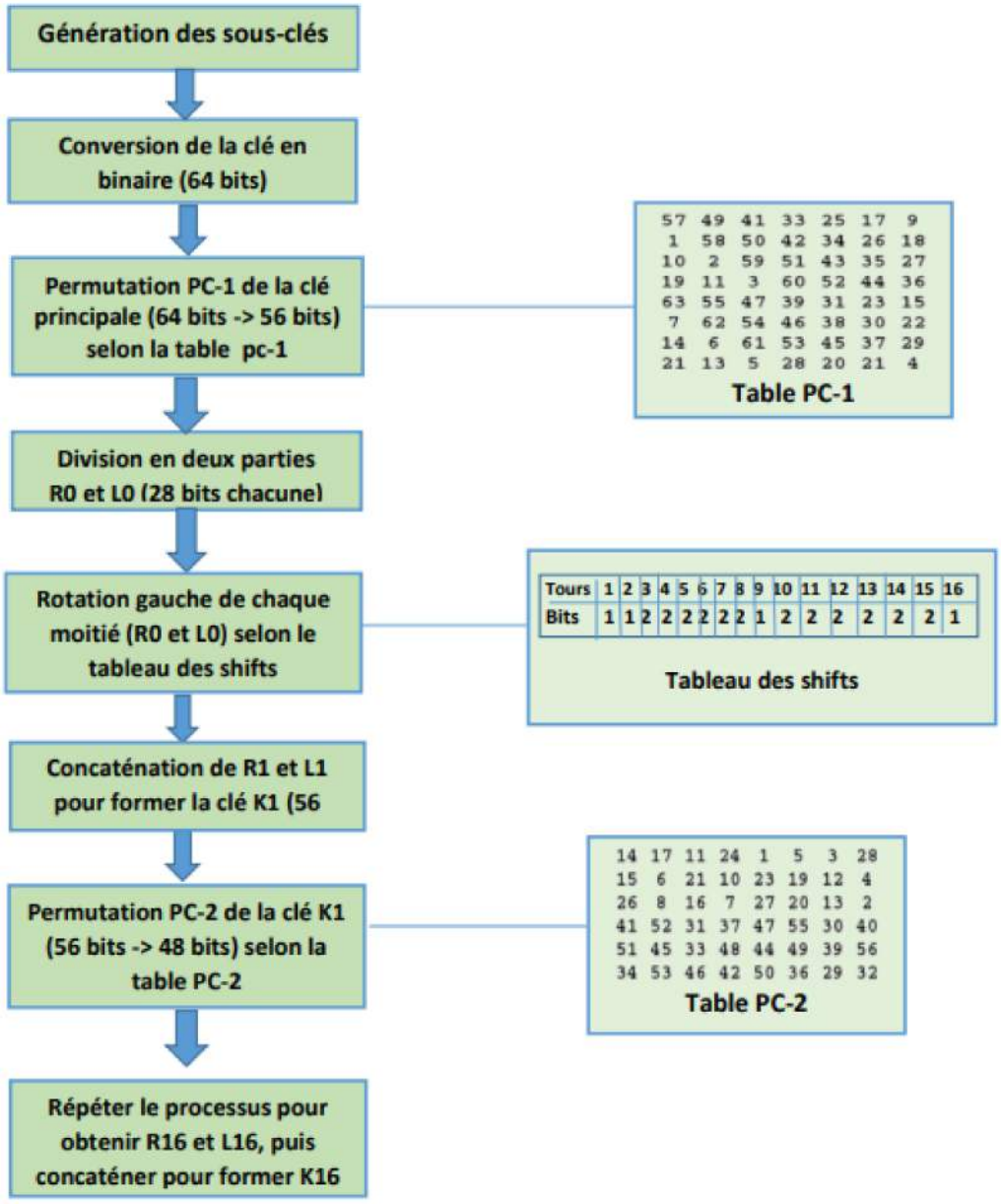

Figure 2.9 – Organigramme de génération des clés du DES

2.2.5.4 Structure et processus de chiffrement

L'organigramme représenté dans la figure 2.10 détaille la structure et le processus complet du chiffrement de l'algorithme DES (Data Encryption Standard).

Il débute par la génération des sous-clés à partir de la clé principale de 64 bits, comme illustré dans la figure précédente. Ensuite, le texte en clair est converti en binaire, séparé en blocs de 64 bits, et passe par une permutation initiale. Pour chacun des 16 tours de chiffrement, les opérations suivantes sont effectuées :

- Le bloc de 64 bits est divisé en deux moitiés, L0 (32 bits de gauche) et R0 (32 bits de droite).
- R0 est étendu à 48 bits à l'aide de la table d'expansion E.
- La sous-clé correspondante au tour (48 bits) est combinée avec le résultat de l'étape 2 par une opération OU exclusif.
- Le résultat de l'étape 3 est divisé en 8 blocs de 6 bits.
- Chaque bloc de 6 bits est substitué par 4 bits en utilisant les boîtes de substitution S (S-boxes).
- Les 8 blocs de 4 bits résultants sont combinés en un seul bloc de 32 bits.
- Ce bloc de 32 bits subit une permutation à l'aide de la table P.
- Le résultat de l'étape 7 est combiné avec L0 par une opération OU exclusif pour former la nouvelle moitié droite R1.
- L1 = R0 (la moitié gauche du tour précédent devient la nouvelle moitié droite).

Ce processus est répété 16 fois, en utilisant une sous-clé différente à chaque tour. Après le 16e tour, les moitiés finales L16 et R16 sont concaténées, puis subissent une permutation inverse pour produire le texte chiffré final.

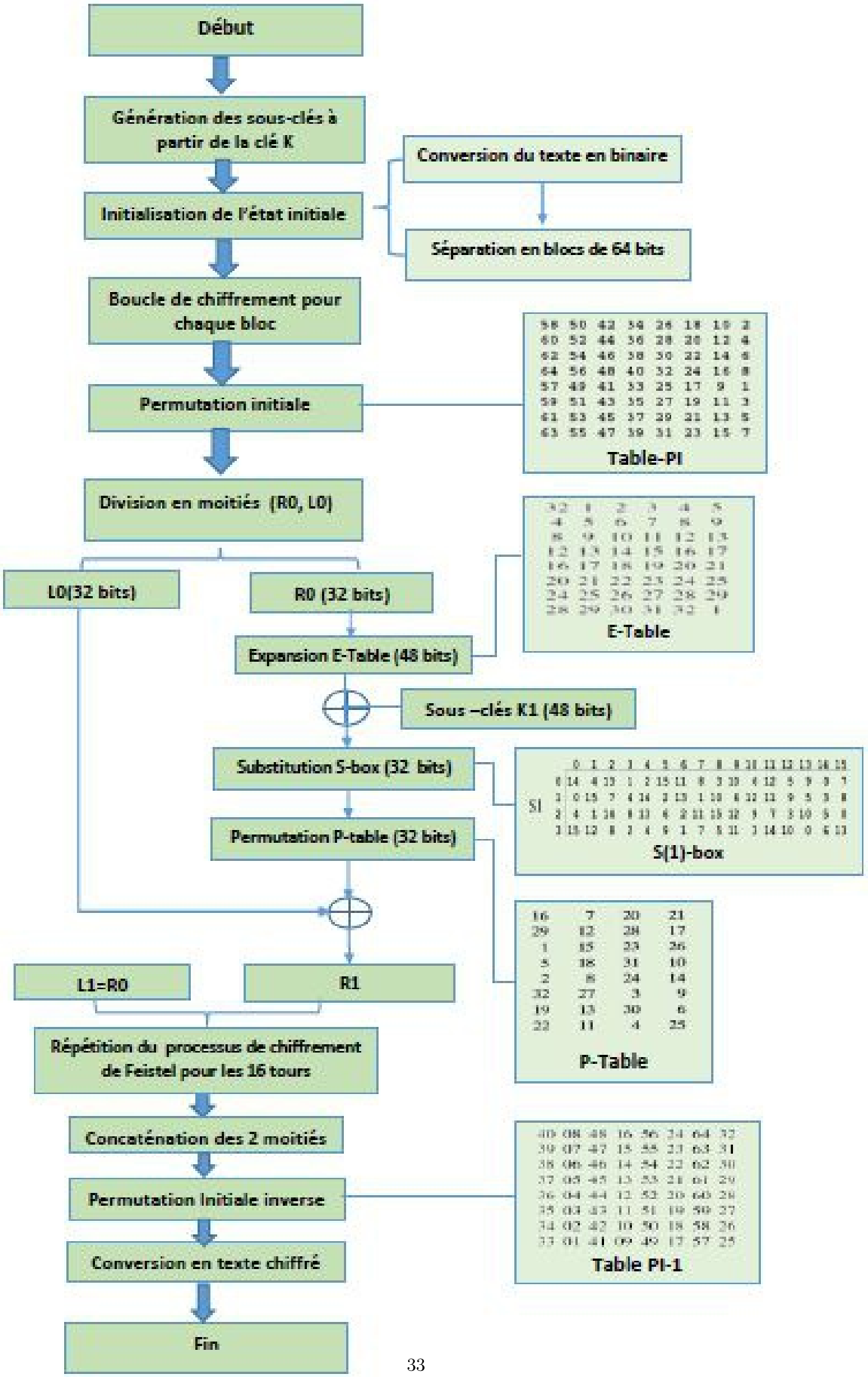

FIGURE 2.10 – Structure et processus de chiffrement du DES

2.2.5.5 Remplacement par l'AES

Le DES a été soumis a de nombreuse attaques en raison de sa longueur de clé relativement restreinte, mais c'est l'essor de la technologie et des ordinateurs qui l'ont rendu obsolète. Ce qui a mené a son remplacement par l'AES pour garantir une meilleure sécurité et une plus grande efficacité avec des clés de tailles plus grande et une vitesse de chiffrement accrue.

2.2.6 AES (Advanced Encryption Standard)

2.2.6.1 Évolution depuis le DES

L'évolution de la cryptographie a été marquée par l'émergence de l'AES comme successeur du DES. Voici un survol des différences clés et des améliorations apportées qui ont conduit à l'adoption de l'AES en tant que norme de chiffrement.

1. Taille de la clé :
   - DES : Limitée a une clé de 56bits.
   - AES : Utilise des clés de 128, 192 et 256 bits.
2. Structure et complexité :
   - DES : Fonctionne sur le réseaux Feistel sur des blocs de 64 bits.
   - AES : opte pour une structure de réseau de substitution-permutation (SPN) simplifié et performante sur des blocs de 128 bits,il est plus rapide.
3. Sécurité :
   - DES : vulnérable aux attaques par force brute, à la cryptanalyse linéaire et différentielle.
   - AES : Robuste et résistant aux attaques connues.
4. Algorithmes :
   - DES : fonctionne en effectuant 16 tours de traitement, où le bloc de données est divisé en deux moitiés et traité individuellement.
   - AES : utilise différents nombre de tours 10 pour les clés de 128 bits, 12 pour celles de 192 bits et 14 pour celle de 256 bits.

Cette évolution marque un progrès significatif dans la sécurisation des systèmes d'information grâce à sa fiabilité et à sa puissance dans la protection des informations critiques. [21]

2.2.6.2 Processus d'expansion de clés

L'algorithme AES utilise une clé initiale (généralement de 128, 192 ou 256 bits) pour chiffrer et déchiffrer les données. Le processus d'expansion de clés, tel qu'organisé dans l'organigramme de la figure 2.11, se déroule en plusieurs étapes.

D'abord, la clé de chiffrement est divisée en mots de 4 octets. Ensuite, des mots de clé supplémentaires sont générés à partir de la clé de départ pour chaque tour de chiffrement. Ces mots de clé subissent une rotation et une substitution pour générer a la fin des sous clé pour chaque tour.

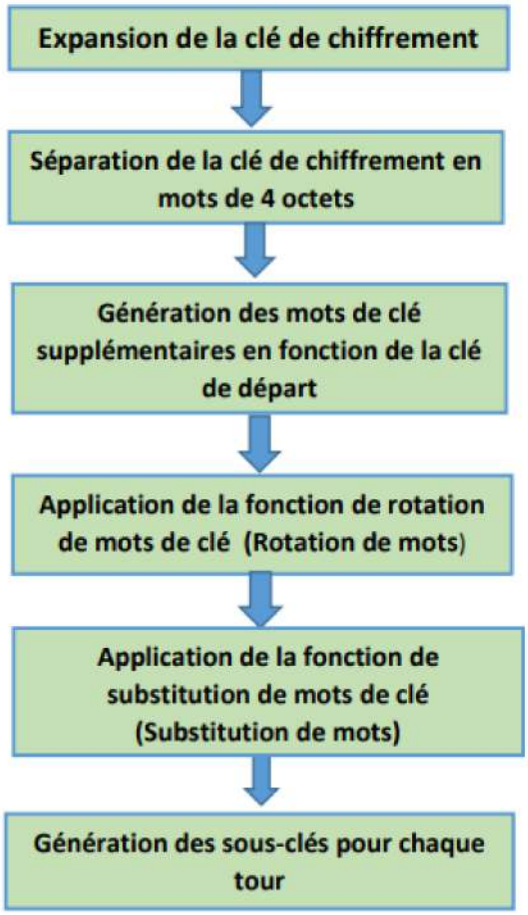

Figure 2.11 – Organigramme du système d'expansion de clé

2.2.6.3 Mécanisme de chiffrement symétrique avancé

— fonctionnement globale du processus de chiffrement de l'AES

La figure ci-dessous représente de manière simplifiée le fonctionnement de l'algorithme de chiffrement avancé AES (Advanced Encryption Standard). Elle illustre les principales

étapes du processus de chiffrement des données en utilisant une clé et des permutations de bits sur une matrice d'état.

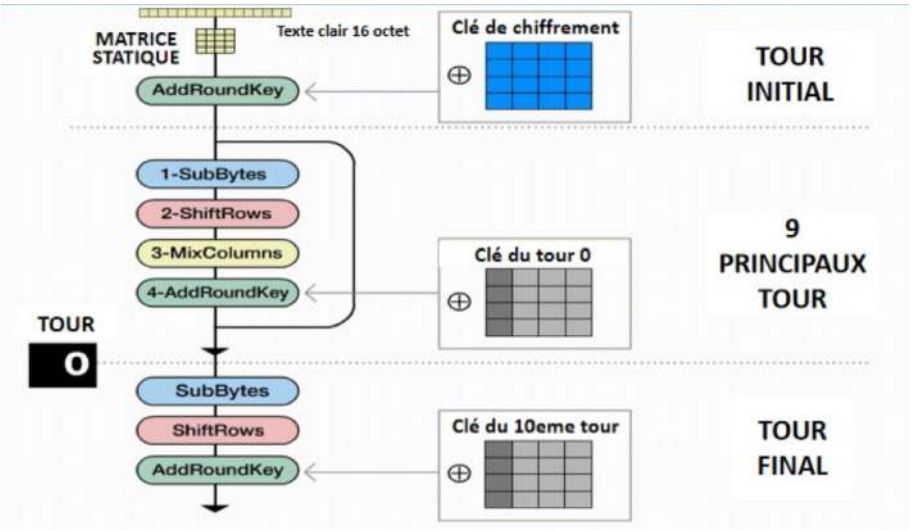

FIGURE 2.12 – Schéma de fonctionnement de l'AES

— Tour unique du fonctionnement de l'AES

La figure ci-dessous représente un tour unique du fonctionnement de l'AES en détaillant les principaux opérations que subis la matrice du texte clair.

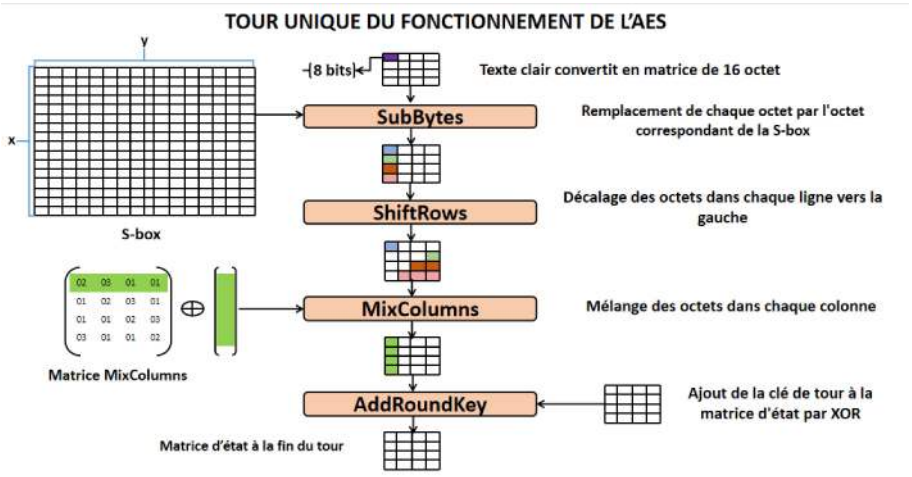

FIGURE 2.13 – Tour unique du fonctionnement de l'AES

L'organigramme de la figure 2.18 illustre le processus de chiffrement AES, expliquant les étapes de transformation d'un message en clair en un message chiffré sécurisé.

Le processus commence par l'expansion de la clé, où la clé secrète est transformée en plusieurs sous-clés pour le chiffrement. Ensuite, lors du tour initial, le message est divisé en blocs et le premier bloc est préparé pour le chiffrement.

Le processus se poursuit par plusieurs tours, au cours desquels des opérations mathématiques complexes sont appliquées au bloc, le mélangeant avec des sous-clés, le nombre de tours dépendant de la longueur de la clé. Chaque tour inclut plusieurs opérations :

— La substitution des octets (SubBytes), où les octets du bloc sont remplacés selon une table définie. Comme le montre la figure ci-dessous qui représente l'étape de substitution SubBytes.

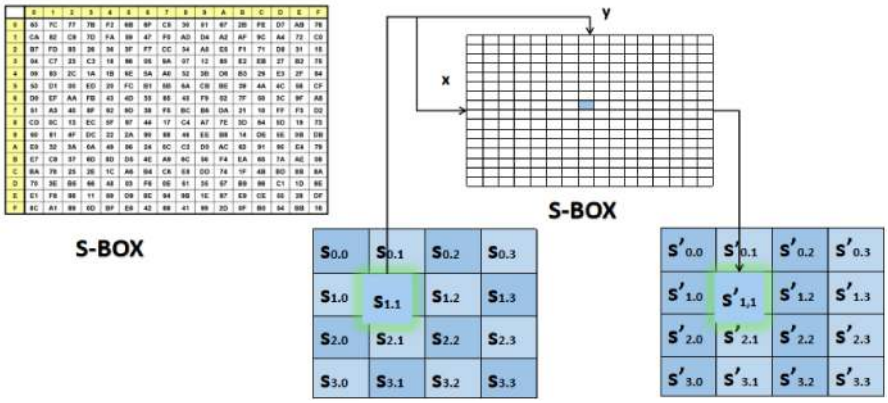

FIGURE 2.14 – L'étape SubBytes

— La permutation des lignes (ShiftRows), les lignes du bloc sont permutées selon un schéma défini, on les décalée vers la gauche par un nombre différent de positions comme le montre la figure **??** qui illustre l'étape shiftRows.

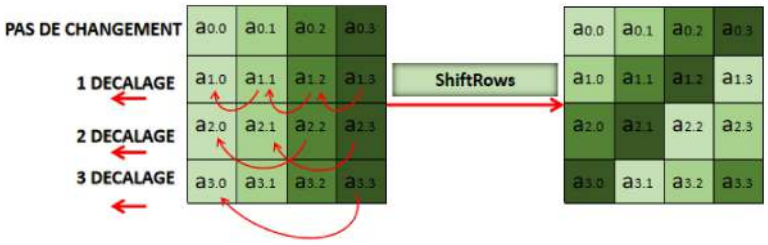

FIGURE 2.15 – L'Etape ShiftRows

— Le mélange des colonnes (MixColumns), où les colonnes du bloc sont mélangées à l'aide d'une opération mathématique une matrice de transformation spécifique. la figure ci dessus représente l'étape MixColumns.

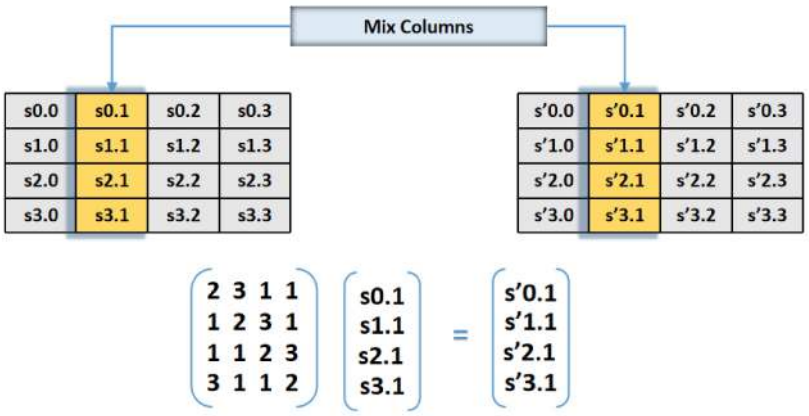

FIGURE 2.16 – L'Etape MixColumns

— L'ajout de la clé de tour (AddRoundKey), une sous-clé est ajoutée au bloc, où chaque octet de la matrice d'état est combiné avec un octet de celle-ci par un XOR bit à bit, tel que l'expose la figure qui représente clairement cette étape.

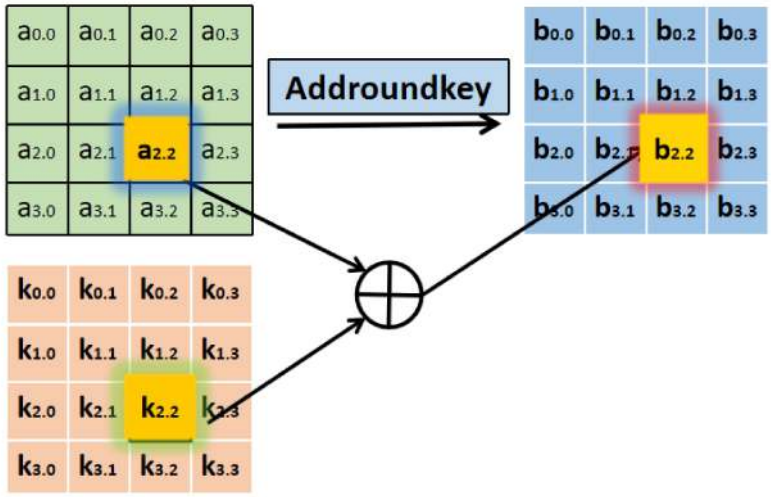

FIGURE 2.17 – L'étape AddRoundKey

Un tour final optionnel peut simplifier le processus en variant légèrement le dernier tour. Enfin, le bloc de données chiffré résultant est produit, complétant ainsi le processus de chiffrement AES.

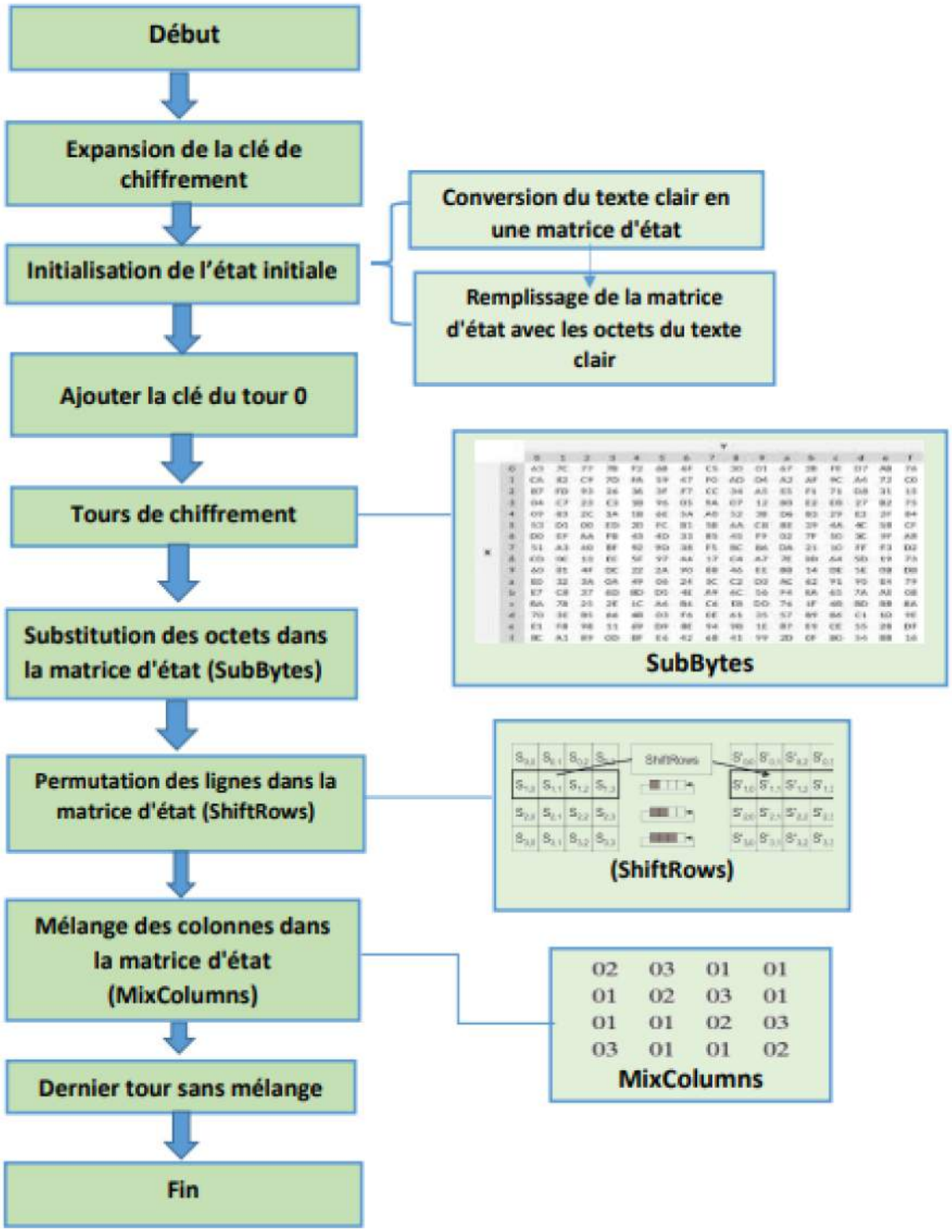

Figure 2.18 – Organigramme du chiffrement AES

2.2.6.4 Applications dans la sécurité des données et des réseaux

L'Advanced Encryption Standard s'impose comme un élément essentiel dans le domaine de la sécurité informatique. en vertu de ses performance et de sa sécurité élevé. voici quelques utilisations clés de l'AES dans ce domaine :

- Stockage sécurisé des données des disques durs, des serveur et des périphériques portables.
- Chiffrement des communications électroniques, telles que les e-mails, les messages instantanés et les transactions en ligne, les protocoles de sécurités (https ssl) s'appuie sur l'AES.
- Authentification des utilisateurs par des mots de passe et des informations d'authentification dans les systèmes informatiques et les réseaux.
- Sécurisation des infrastructures importante telles que les réseaux électriques, les systèmes de transport et les installations médicales contre les cybermenaces.
- Chiffrement des données et contrôle d'accès au niveaux granulaires, limitant ainsi l'accès aux utilisateurs et aux applications autorisés.
- Protection des clés privées et sécurisation des transactions numériques des portefeuilles de cryptomonnaies. [17]

2.3 Les algorithmes de cryptographie asymétrique

2.3.1 Processus de génération des clés et d'encodage

La figure 2.19 représente l'organigramme de génération des clés et d'encodage du RSA. La génération des clés RSA commence par le choix de deux nombres premiers aléatoires, notés p et q.

Ensuite, on calcule le module de chiffrement n $n = p * q$ et la fonction d'Euler de n $(= (p-1) * (q-1)$.

Un exposant de chiffrement e est sélectionné de sorte qu'il soit positif, et premier a la fonction d'Euler de n. On calcule ensuite l'exposant de déchiffrement d comme l'inverse multiplicatif de e modulo la fonction d'Euler de(n).

Les clés résultantes sont : la clé publique (e, n) et la clé privée (d, n). Grâce à cette paire de clés, les messages peuvent être chiffrés avec la clé publique et déchiffrés avec la clé privée.

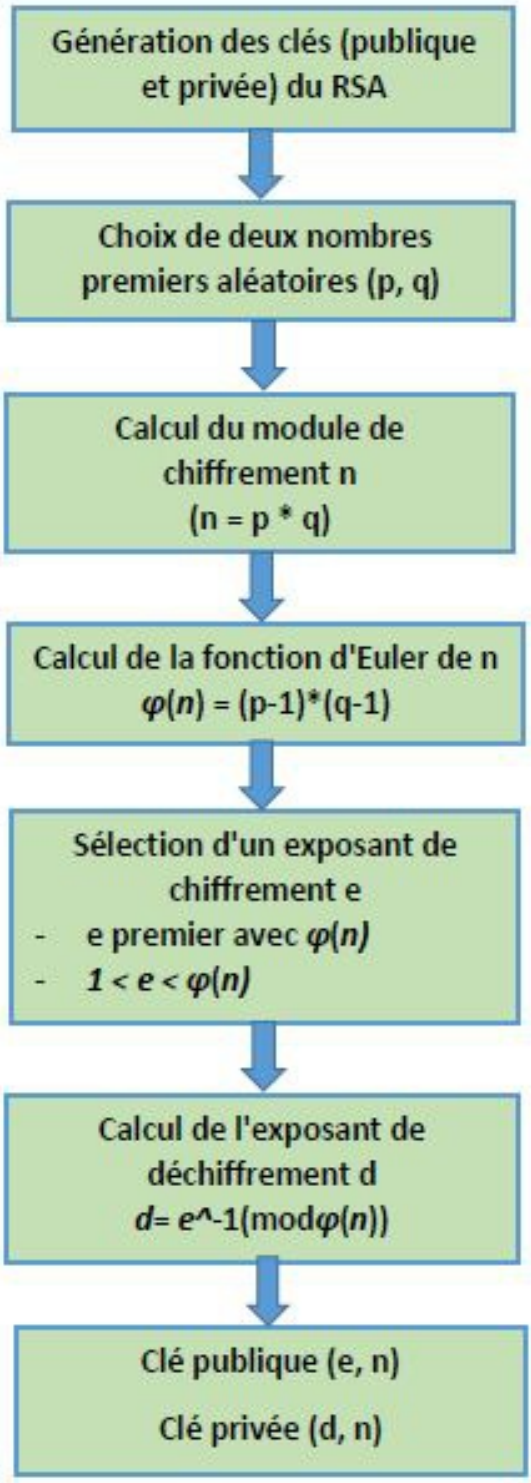

FIGURE 2.19 – Processus de génération des clés du RSA

2.3.2 Fondement mathématique de la cryptographie a clé publique

L'organigramme présenté dans la figure 2.21 démontre le processus de chiffrement et de déchiffrement des données à l'aide de l'algorithme RSA, qui se décompose en trois étapes principales.

La première étape est la génération des clés cryptographiques nécessaires au chiffrement et au déchiffrement des messages, où les clés publique et privée sont respectivement (e, n) et (d, n).

La deuxième étape concerne le chiffrement du message : le message clair M est d'abord converti en un entier m, puis le chiffrement est réalisé en calculant $m^e(mod(n))$,produisant ainsi le message chiffré C, prêt à être transmis au destinataire, comme le montre la figure ci-dessous qui représente le principe mathématique du chiffrement et du déchiffrement en utilisant le RSA

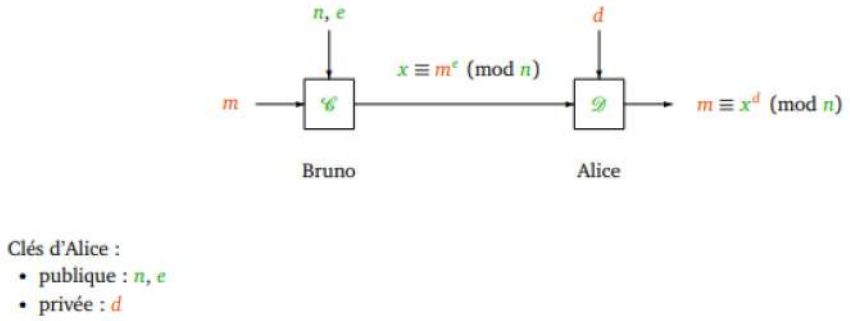

Figure 2.20 – Principe mathématique du RSA

La troisième étape est le déchiffrement du message : le destinataire reçoit le message chiffré C et le déchiffre en calculant $C^d(mod(n))$ pour obtenir le nombre m, qui est ensuite reconverti en message original M, reconstituant ainsi le contenu initial tel qu'illustré dans la figure précédente.

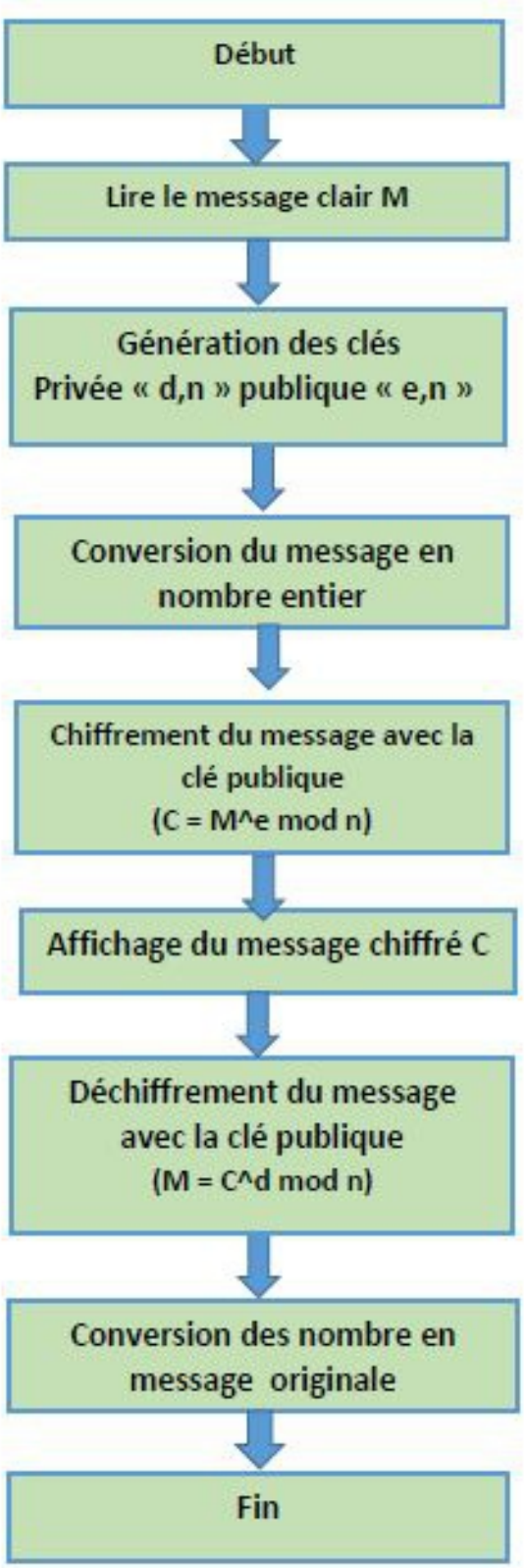

FIGURE 2.21 – Organigramme du processus de chiffrement et de déchiffrement du RSA

2.3.3 Sécurité et utilisation dans les communication modernes

le RSA fait partie des algorithmes les plus performants en matière de cryptographie, Malgré les défis important de la cryptanalyse, le RSA a prouvé sa résistance au fil du temps. Sa sécurité repose sur la difficulté de factoriser de grands nombres premiers, il n'y a aucune solution efficace pour ce résoudre ce problème mathématique complexe pour les clés de taille courante. Néanmoins, il faut augmenter la taille des nombres utilisés dans la mesure où les ordinateurs deviennent plus puissants et les algorithmes de factorisation plus efficaces afin garantir cette sécurité à long terme.

C'est un système universel utilisé dans de nombreuses application telles que :

— L'authentification et la sécurité des sites Web.
— Les échanges bancaires et financières.
— Les réseaux privés virtuels (VPN).
— La messagerie instantanée et vocale.
— La vérification d'identité numérique.
— La signature de code et l'authentification de logicielle. [?, 24]

2.4 Conclusion

Ce chapitre a offert une exploration approfondie des mécanismes de chiffrement et de déchiffrement, allant des algorithmes symétriques classiques aux méthodes modernes et aux machines historiques. En détaillant le fonctionnement, les caractéristiques et les faiblesses des méthodes comme celles de César, Vigenère, Enigma, DES, AES, et RSA, nous avons mis en lumière l'évolution de la cryptographie en réponse aux progrès technologiques et aux nouvelles menaces. Dans le chapitre suivant, on mettra en œuvre une interface graphique en implémentant les divers algorithmes mentionnés, offrant ainsi une application pratique de ces techniques.

Chapitre 3

Implémentations des Algorithmes de Chiffrement dans l'interface Graphique

3.1 Introduction

Dans ce chapitre, nous opérons la transition de l'abstrait au concret en implémentant les différents algorithmes de chiffrement, explorés dans le chapitre précédent au sein d'une interface graphique conviviale dont le but est de fournir a l'utilisateur un outil interactif, permettant de chiffrer et de déchiffrer des messages en expérimentant les différentes méthodes cryptographiques, sans avoir à plonger dans les détails techniques complexes.

Nous explorerons en détail la conception, l'implémentation et les fonctionnalités de cette interface en testant chaque algorithme, du classique comme César et Vigenère, jusqu'aux plus modernes tels que AES, DES et RSA, sans oublier la machine historique Enigma et la substitution.

3.2 Conception de l'interface

3.2.1 Environnement matériel

L'implémentation de notre application a été réalisée sur un micro-portable HP personnel, fonctionnant sur le système d'exploitation Windows 10 Professionnel de 64 bits. Les caractéristiques techniques de l'ordinateur utilisé sont les suivantes :

— Un processeur Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz, avec une fréquence maximale de 2.50 GHz, appartenant à la sixième génération de processeurs Intel.

— Une mémoire RAM de 8,00 Go, pour le développement et l'exécution de l'application.

— Un disque dur de 237,58 Go NTFS, suffisamment grand pour stocker le système d'exploitation, les outils de développement, les bibliothèques nécessaires, et les données de projet.
— Une carte graphique Intel(R) HD Graphics 520, adaptée pour l'affichage de l'interface utilisateur de l'application. Malgré sa modestie elle est suffisante pour supporter les besoins graphiques de notre projet.

3.2.2 Environnement logiciel

Notre projet a été développé en utilisant la version Python 3.11.7 et Spyder 5 Comme environnement de développement intégré, offrant des fonctionnalités avancées telles que la gestion de projet, l'éditeur de code avec une coloration syntaxique et des outils de profilage et de débogage, facilitant ainsi la réalisation de celui-ci.

3.2.3 Outils et bibliothèque

Pour la mise en œuvre de notre interface graphique, et pour la gestion des opérations de chiffrement et de déchiffrement nous avons utilisé les outils et bibliothèques représentait dans la figure 3.1 .

```
1  import tkinter as tk
2  from tkinter import ttk, messagebox
3  from Crypto.Cipher import DES, AES
4  from Crypto.Util.Padding import pad, unpad
5  import string
6  import math
7  from PIL import Image, ImageTk
8  import os
9
10
```

FIGURE 3.1 – Script des bibliothèque importé

— Tkinter : cette bibliothèque standard permet de développer des applications GUI (Graphical User Interface), en utilisant les outils spécifié a la créations des fenêtres, des labels, des menus, des boutons et d'autres composants.
— Ttk :une extension de la bibliothèque Tkinter, qui permet d'ajouter des widgets pour amélioré l'esthétique des interfaces créés avec celle-ci.
— Messagebox :un module de Tkinter Utilisé pour l'affichage des boîtes de dialogue dans Tkinter tel que les messages d'information, d'avertissement ou d'erreurs a l'utilisateur.

— Crypto (PyCryptodome) : une bibliothèque de cryptographie Utilisé pour les opérations de chiffrement robuste, tel que le DES et l'AES, ainsi que pour les fonctions de padding et unpadding.
— Les modules string et math intégrés a python, sont utilisés pour la manipulation des chaînes de caractères et pour les opérations mathématiques, tel que les calculs logarithmiques, trigonométriques et de puissance.
— Pillow (PIL) : une version améliorée de la bibliothèque PIL (Python Imaging Library) utilisé pour la manipulation, l'ouverture et l'affichage des images dans divers formats.
— Os : cette bibliothèque est utilisée pour la gestion des fichiers et des répertoires, en interagissant avec le système d'exploitation.

Ces bibliothèques forment le squelette du développement de notre application GUI.

3.2.4 Vue d'ensemble et architecture de l'interface

L'application offre une interface graphique simple et intuitive, permettant a l'utilisateur de choisir et d'exploiter les différents algorithmes de chiffrement présent dans la fenêtre principale. On y trouve la Substitution, César, Vigenère, Enigma, DES, AES et RSA, le tout représenté en une liste dont Le choix se fait via des boutons radio. Comme le montre la figure 3.2. Elle comprend aussi un bouton « ouvrir » pour accéder à la fenêtre dédiée à l'algorithme choisi afin de saisir le texte en clair, d'ajuster les paramètres spécifiques et d'effectuer les opérations de chiffrement et de déchiffrement.

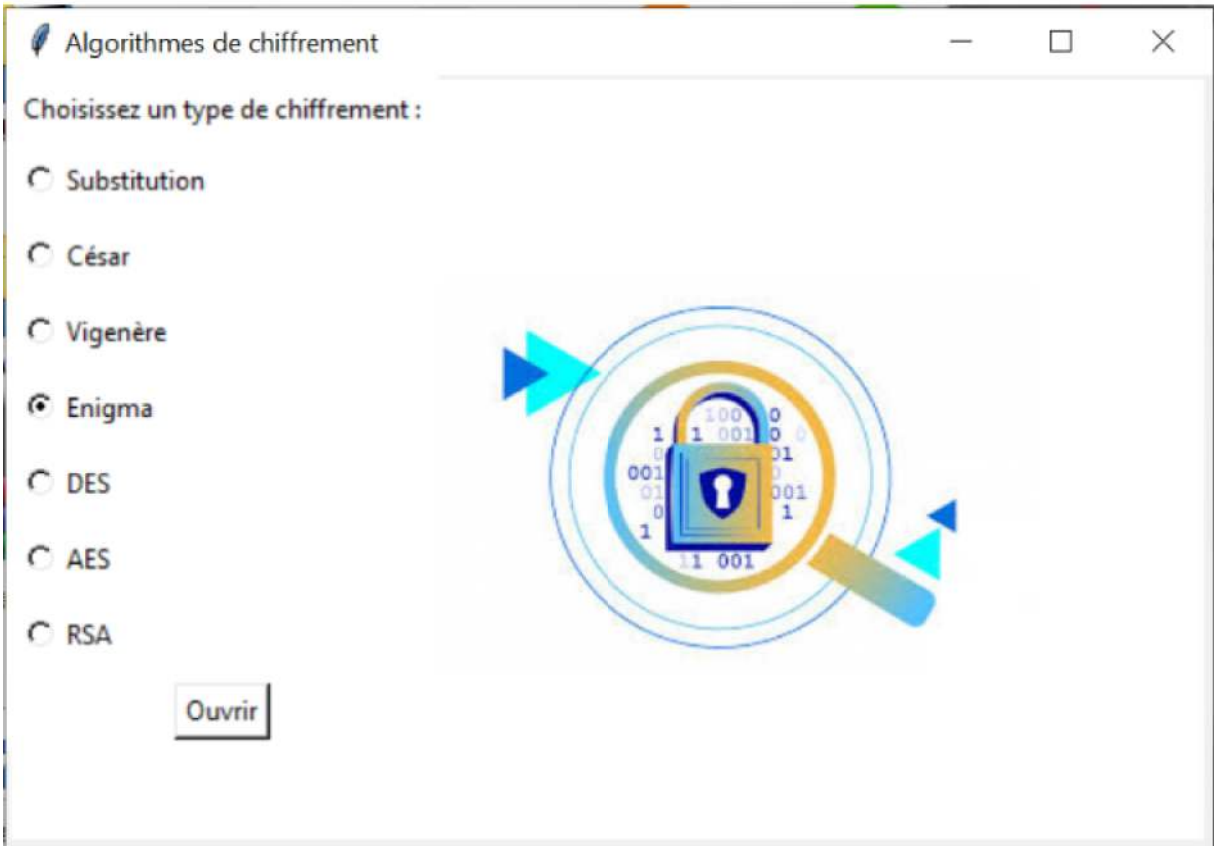

FIGURE 3.2 – Écran principale de l"interface graphique

La figure ci-dessus représente la fenêtre principale de notre interface graphique.

L'architecture est modulaire, séparant l'interface utilisateur du mécanisme algorithmique du cryptage, donc chaque algorithme a une fenêtre spécifique qui contient des champs de saisie pour les paramètres requis par ce dernier, ainsi que des boutons pour lancer les opérations de chiffrement et de déchiffrement. Facilitant ainsi l'extension et la maintenance de l'application. On prend pour exemple la figure 3.3.

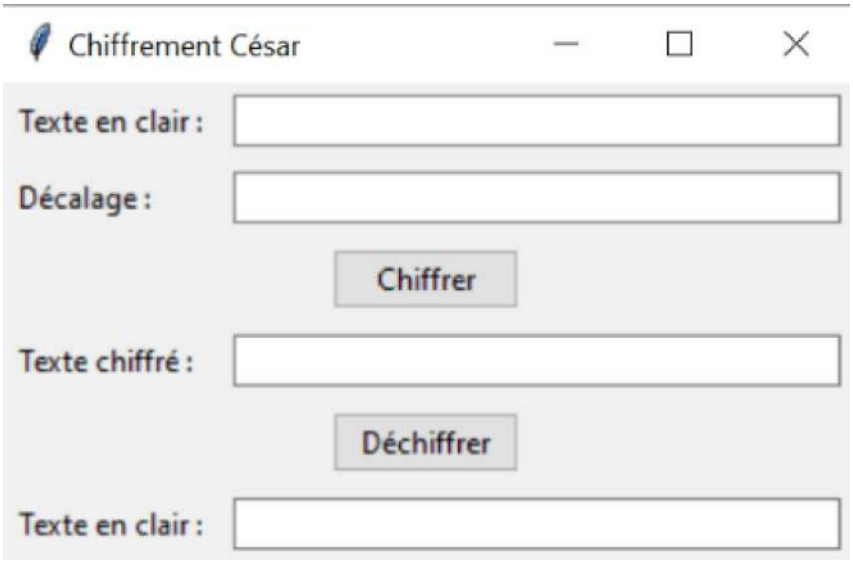

FIGURE 3.3 – Fenêtre du chiffrement de césar

La figure ci dessus représente la fenêtre du chiffrement de César avec un champs de saisie pour le texte clair et un autre pour le décalage, ainsi que des boutons pour le chiffrement et le déchiffrement. Chaque algorithmes a des paramètres spécifié du plus simple au plus complexe.

3.3 Implémentation et test des Algorithmes de Chiffrement

Dans cette partie nous allons testé le fonctionnement de notre interface en chiffrant le même message avec les différents algorithmes symétriques et asymétriques présent sur celle-ci.

3.3.1 Algorithme de chiffrement symétrique

3.3.1.1 Le chiffrement par substitution

Le chiffrement par substitution est assez simple, en sélectionnant le bouton radio de la substitution puis en cliquant sur ouvrir, apparaît une nouvelle fenêtre sur laquelle on trouve des champs de saisie pour le texte clair et pour la clé, qui doit être un alphabet de 26 lettres ordonnées aléatoirement, puis en appuyant sur les boutons chiffré et déchiffré s'effectue les opérations correspondante pour afficher ensuite le texte chiffré ainsi que le texte clair.

La figure 3.4 illustre l'exemple du chiffrement et du déchiffrement du mot "bonjour", avec le chiffrement par substitution mono-alphabétique.

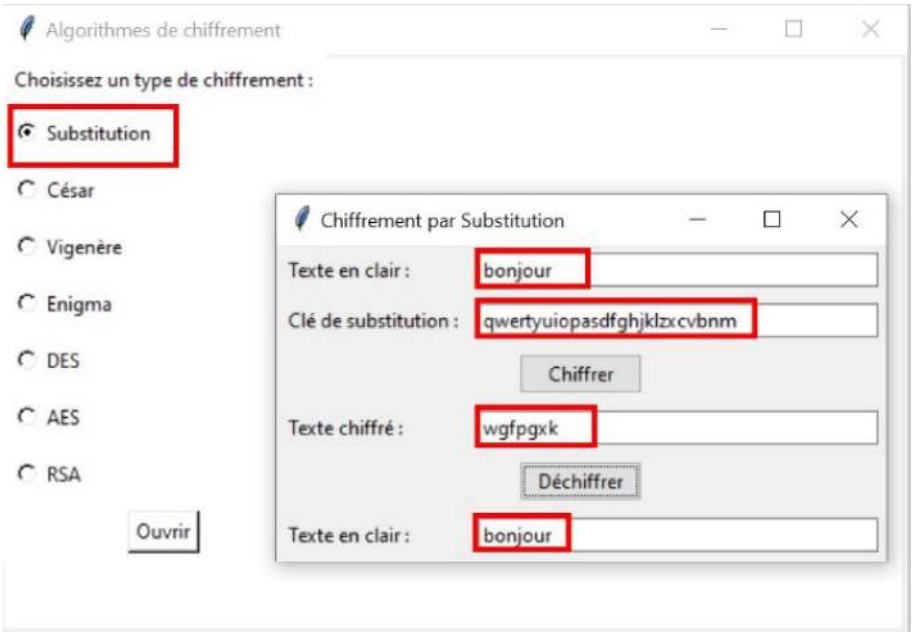

FIGURE 3.4 – Exemple de chiffrement et de déchiffrement par substitution

Pour le chiffrement chaque lettre du mot "bonjour" est substitué par une autre prédéfinit par la clé de substitution "qwertyuiopasdfghjklzxcvbnm", on obtient le mot "wgfpgxk". Tandis que pour le déchiffrement, on utilise l'inversion de la clé pour restaurer le texte original.

Parmi les faiblesses de ce type de chiffrement est qu'une même lettre est toujours chiffré de la même façon, ce qui le rend vulnérable aux attaques par analyse de fréquence et par force brute comme le montre la figure 3.5, qui représente un exemple de chiffrement d'un long message en utilisant la substitution.

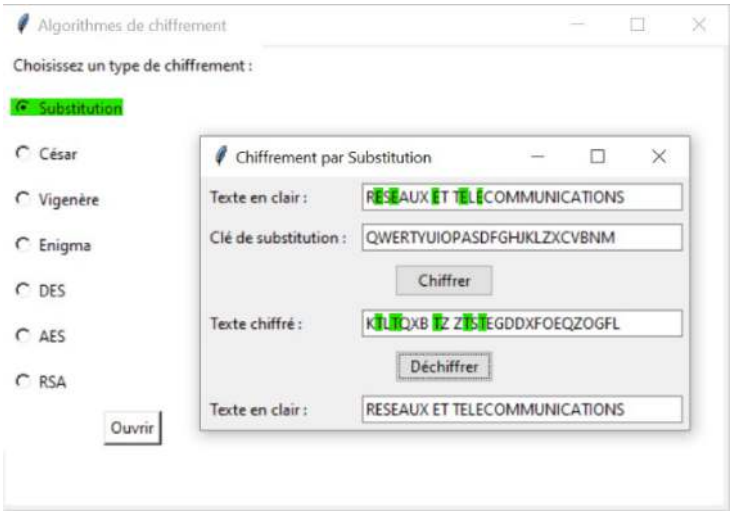

FIGURE 3.5 – Exemple de chiffrement illustrant une des Vulnérabilité de la substitution

En chiffrant un long texte avec cette méthode, on remarque que la lettre E qui se répète 5 fois dans le message clair et toujours chiffré par un T selon la clé de substitution .

3.3.1.2 Le chiffrement de César

Le chiffrement de césar est un chiffrement par substitution, qui repose sur le décalage de chaque lettre du texte clair d'un nombre fixe de positions dans l'alphabet selon la clé choisi.

La figure 3.6 représente l'utilisation de l'algorithme de césar implémenté dans l'interface graphique pour chiffré le mot "bonjour".

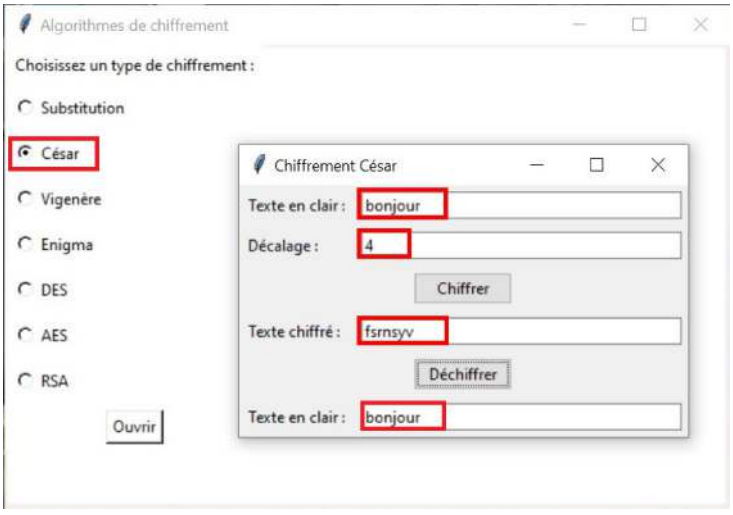

FIGURE 3.6 – Chiffrement et déchiffrement en utilisant l'algorithme de césar

On a utilisé un décalage de 4 positions, ce qui fait que le "b" se décale de 4 position dans l'alphabet et devient f et de même pour les autres lettres, pour le déchiffrement le décalage inverse est appliqué.

3.3.1.3 Le chiffrement de vigenere

Contrairement a César, le chiffrement de vigenere utilise la substitution poly-alphabétique avec une clé qui se répète pour correspondre a la longueur du message et pour accomplir cela. Nous avons utilisé la fonction "extended key" comme le montre la figure ci-dessous, qui représente un script du code d'implémentation qui concerne l'extension de la clé.

```
35      def extend_key(self, key, length):
36          extended_key = ""
37          idx = 0
38          for i in range(length):
39              if idx == len(key):
40                  idx = 0
41              extended_key += key[idx]
42              idx += 1
43          return extended_key
```

Figure 3.7 – Extrait du code concernant l'extension de la clé

La méthode "extended key" définit dans ce script, prend deux argument (key, length) afin de créer une nouvelle chaîne de caractère en répétant la chaîne "key" pour la longueur du message, en utilisant une boucle donc pour le mot "bonjour" la clé devient "keykeyk".

La figure ci-dessous illustre le chiffrement du mot "bonjour" avec la clé "key", en utilisant le chiffrement de vigenère présent sur l'interface.

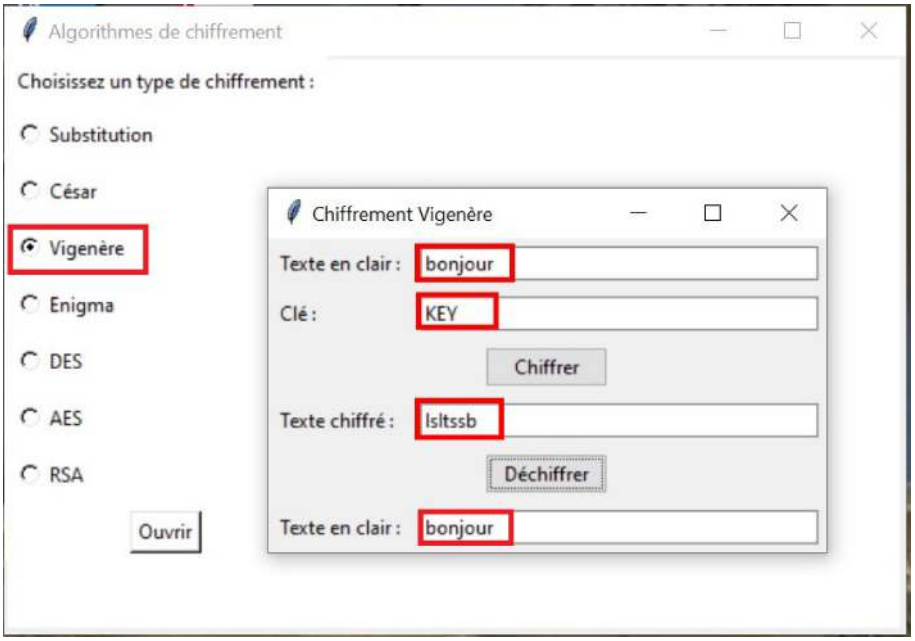

FIGURE 3.8 – Exemple d'application du chiffrement de Vigenère

En appuyant sur le bouton chiffré, une fois la clé et le texte en clair entré on obtient le texte chiffré "lsltssb", et le bouton déchiffrer sert a récupérer le message original .

Le principe de la substitution poly-alphabetique est qu'une lettre n'est pas toujours chiffré de la même manière, mais on remarque que la lettre "o" est chiffré deux fois par un "S" due a la répétitions de la clé qui est relativement petite par rapport au message clair, ce qui peut conduire a la cryptanalyse du message, en devinant la longueur de la clé suite a la répétition d'une séquence de 3 lettres dans le texte chiffré, comme pour l'exemple illustré dans la figure 3.9 qui représente le chiffrement du mot "RESEAUXET-TELECOMMUNICATIONS" avec la clé "KEY" en utilisant le chiffrement de vigenere .

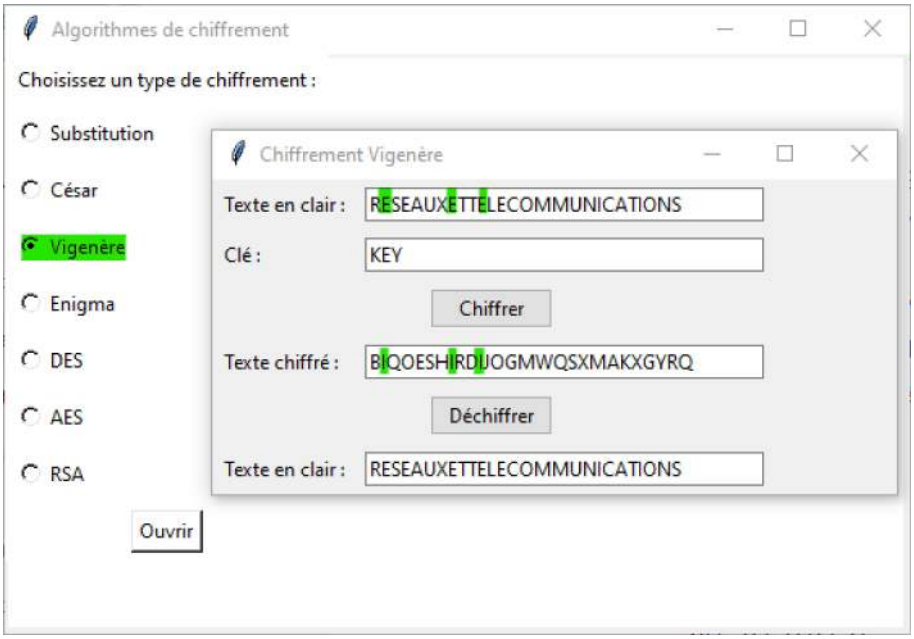

FIGURE 3.9 – Analyse des failles du chiffrement de Vigenère

La lettre "E" dans le texte "RESEAUXETTELECOMMUNICATIONS" est chiffré 3 fois par la lettre "I", le 1er "I" indique que c'est la 2 ème lettre de la clé qui correspond a ces 3 lettre, tandis que le 2ème et le 3ème "I" on 2 lettres entre eux, alors la clé est composé de 3 lettres.

3.3.1.4 La machine Enigma

Le fonctionnement de la machine Enigma repose sur les rotors et le tableau de connexion, pour chiffrer un message avec cette dernière il faut configurer la position des 3 rotors entre 0 et 25, ainsi que le tableau de connexions des lettres. La figure 3.10 représente un script du code d'implementation concernant la machine Enigma

```
87      def set_rotor_positions(self, positions):
88          self.position = positions
89
90      def set_plugboard(self, connections):
91          self.plugboard = {pair[0].upper(): pair[1].upper() for pair in connections}  # Connexions en majuscules
92
```

FIGURE 3.10 – Script du code concernant les configurations de la machine Enigma

La commande "set rotor positions" permet de définir la positions des rotors, tandis que la commande "set plugboard" permet de configurer les connexions du plugbord fonctionnant en majuscules seulement, si le message en clair est en minuscule, il sera convertit et affiché en majuscule comme le montre la figure 3.11 qui représente la fenêtre dédiée a la machine Enigma, sur laquelle on essaie de chiffrer le mot "bonjour".

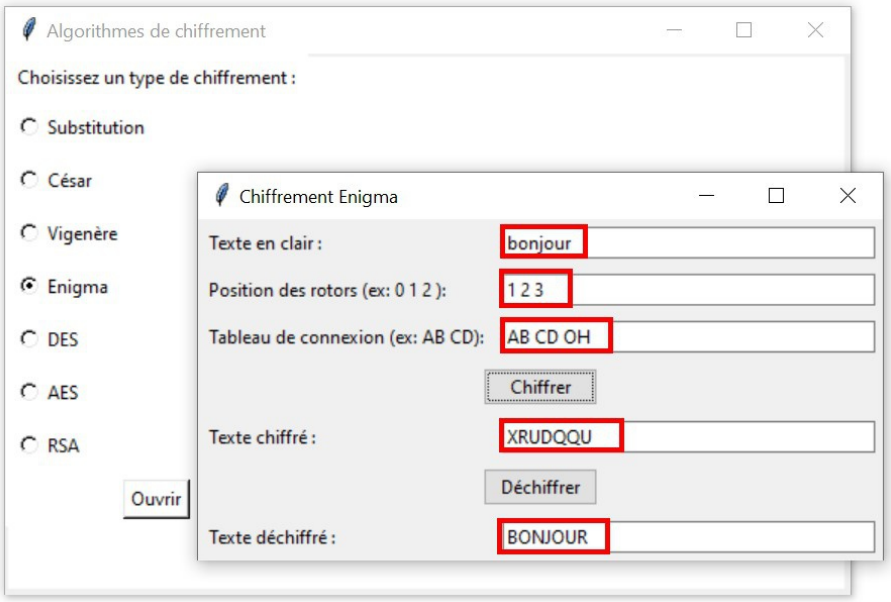

FIGURE 3.11 – Exemple de chiffrement avec la machine Enigma

Nous avons configuré la position des 3 rotors "0 1 2" ainsi que le plugbord avec "A" relié a "B", "C" relié a "D" et "O" relié a "H", pour obtenir le message chiffré en majuscule "AYRYLI". Et pour le déchiffrement on appuie sur le bouton "déchiffrer" et on obtient le message original mais celui-ci est en majuscule, car Enigma fonctionne sur ce principe.

3.3.1.5 Data Encryption Standard DES

Pour faire une démonstration sur l'utilisation du chiffrement DES implémenté dans notre application, on chiffre le mot "bonjour" avec la clé "mykey123", qui doit être de 8 caractère comme le montre la figure.3.12 qui représente cette exemple sur la fenêtre de l'algorithme data encryption standard.

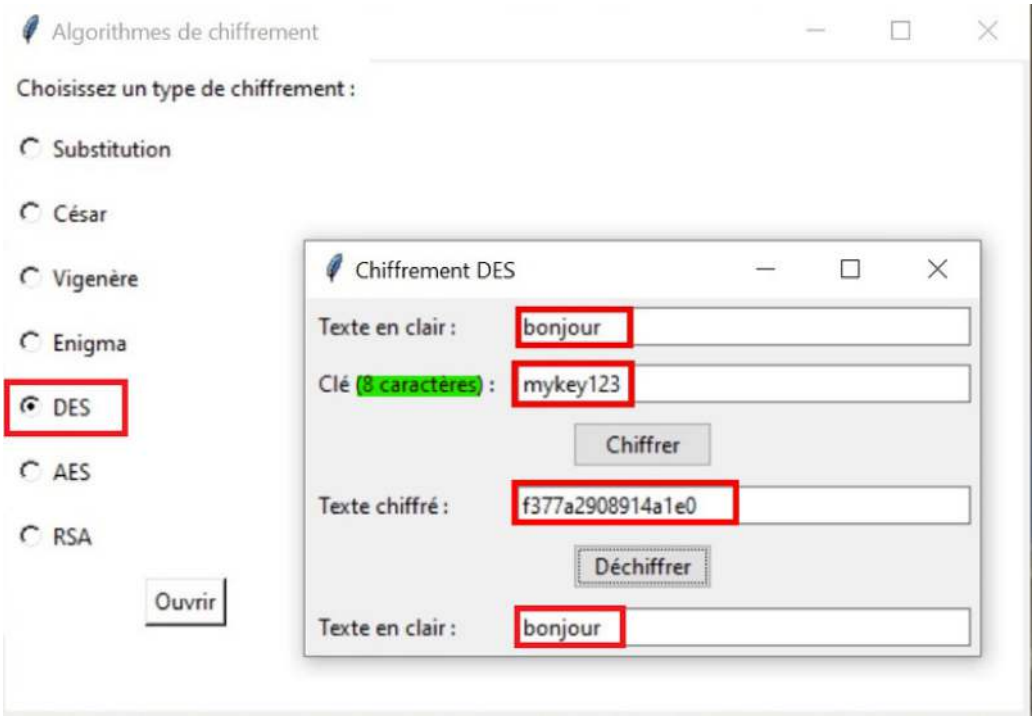

FIGURE 3.12 – Exemple de chiffrement en utilisant le DES

Le texte chiffré obtenue "f377a2908914a1e0", est une chaîne composé de 16 caractère hexadécimale qui représente le texte codé en binaire.

Le DES ne fonctionne pas si la clé n'est pas composé de 8 caractère exactement, c'est pour cela que nous avons introduit cette condition dans le code d'implémentation. La figure ci dessous représente un script de l'algorithme DES.

```
562     def encrypt(self):
563         plaintext = self.plaintext_entry.get()
564         key = self.key_entry.get()
565         if len(key) != 8:
566             messagebox.showerror("Erreur", "La clé doit contenir 8 caractères.")
567             return
```

FIGURE 3.13 – Script concernant la vérification des clés du DES

Dans le cas ou il y' a une erreur concernant la clé, s'ouvre une boite de dialogue sur laquelle s'affiche le message représenté dans la figure 3.14 qui illustre la condition du script précèdent.

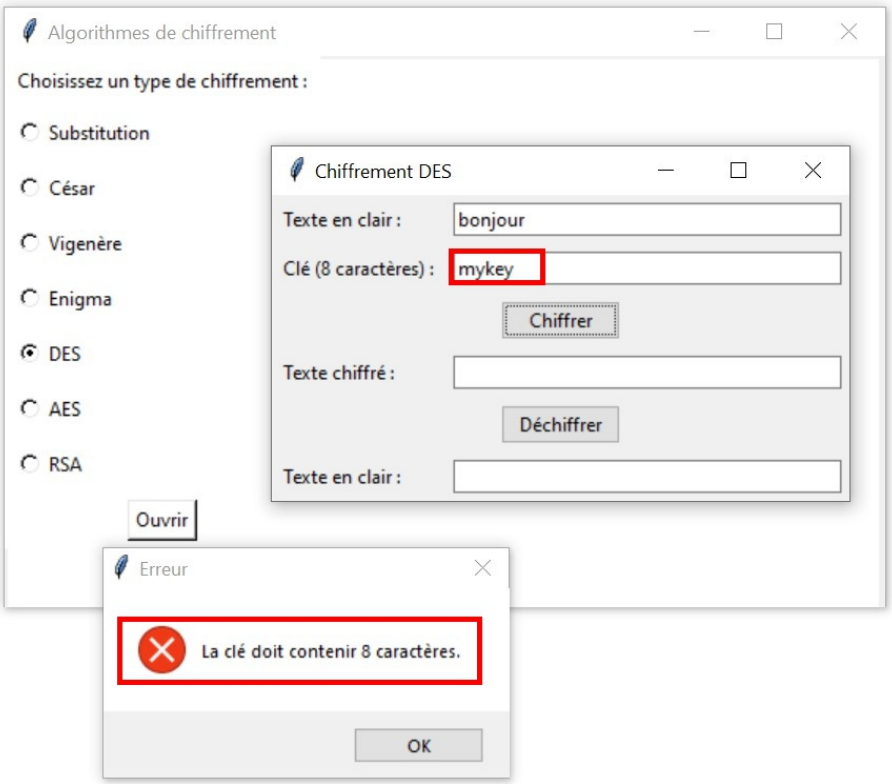

FIGURE 3.14 – Message d'erreur pour la clé invalide du DES

Le message indique que "la clé doit contenir 8 caractère" pour une utilisation facile est correcte de l'interface.

3.3.1.6 Advanced Encryption Standard AES

A l'inverse du DES, l'AES fonctionne avec des clés de taille différentes (128,192 et 256 bits) selon le niveaux de sécurité que l'on requiert. Nous avons implémenter les 3 clés dans notre application que nous allons tester en choisissant l'AES parmi les types d'algorithme présent sur la fenêtre principale, s'ouvre une fenêtre sur laquelle on doit entré le texte en clair, ainsi que la clé spécifié a la taille choisis comme le montre les figures ci-dessous. La figure 3.15 démontre un exemple de chiffrement avec l'algorithme AES présent sur l'interface en utilisant la clé de 128 bits.

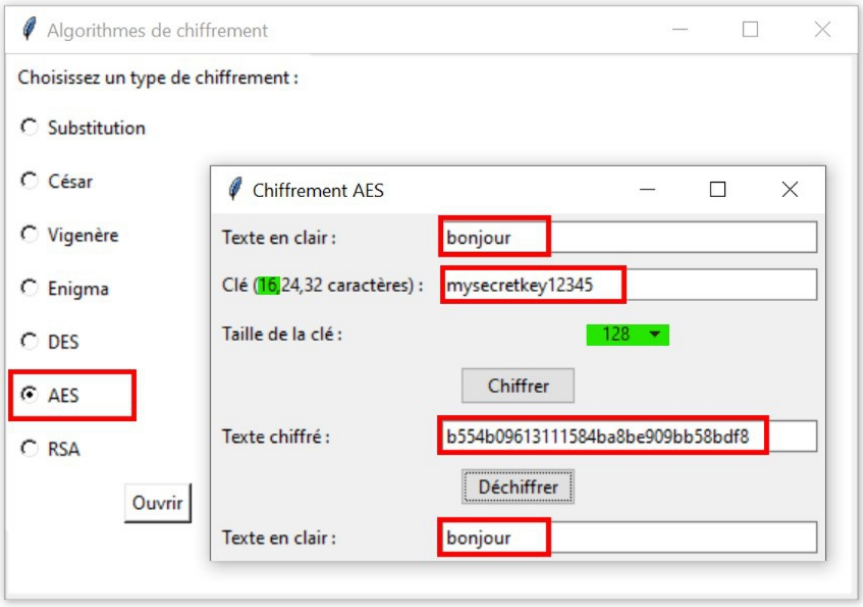

FIGURE 3.15 – Exemple de chiffrement en utilisant la clé de 128 bits de l'AES

Pour le chiffrement du mot "bonjour" avec une clé de 128 bits, il faut qu'elle soit de 16 caractères, dans cette exemple c'est 'mysecretkey12345", pour obtenir le texte chiffré représenté en une chaîne de 32 caractères hexadécimales. Tandis que pour le déchiffrement il suffit de cliquer sur le bouton "déchiffrer" afin d'obtenir le texte original.

La figure ci-dessous représente un exemple de chiffrement en utilisant la clé de 192 bits.

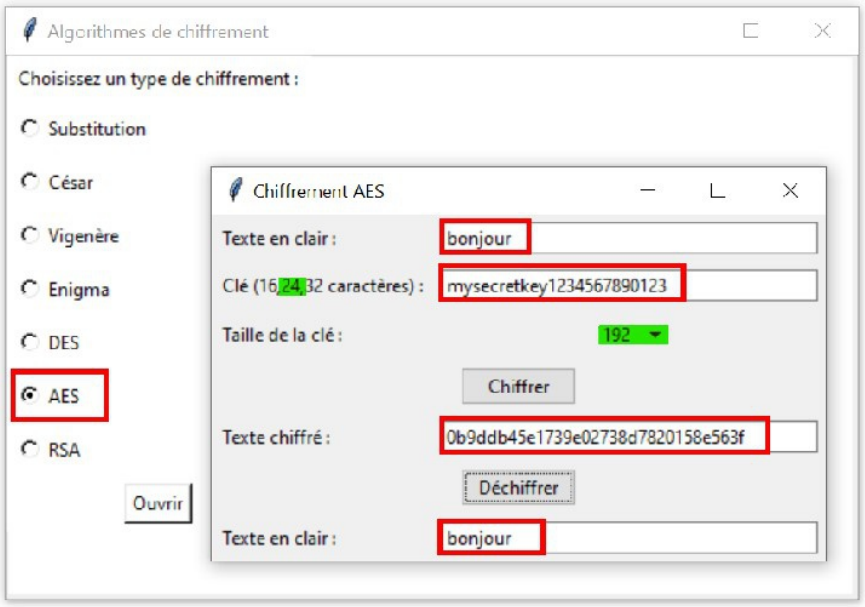

FIGURE 3.16 – Exemple de chiffrement avec la clé de 192 bits

La clé de 192 bits doit contenir 24 caractères. On a utilisé "mysecretkey1234567890123" comme clé pour chiffrer le mot "bonjour", et on obtient une chaîne de 32 caractères hexadécimales différente de celle de 128 bits, pour chiffrer le mot. La figure 3.17 illustre l'utilisation de la clé de 256 bits pour chiffrer le mot "bonjour" avec l'Advanced Encryption Standard.

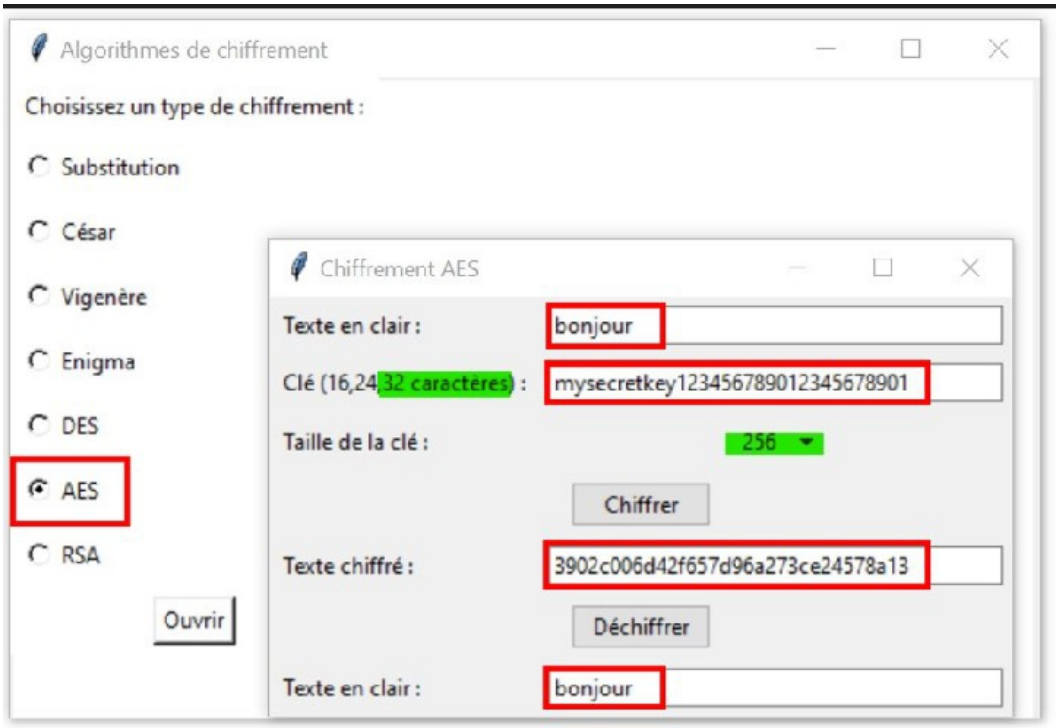

FIGURE 3.17 – Exemple de chiffrement en utilisant la clé de 256 bits

On a utilisé la clé "mysecretkey123456789012345678901" de 32 caractères pour chiffrer le mot "bonjour", et on obtient toujours une chaîne de 32 caractères hexadécimal qui correspond au texte clair chiffré avec les différentes clés, si le message est plus long le texte chiffré le seras aussi d'où vient sa sécurité.

Pour chaque clé, le nombre de caractère défini doit être respecté, c'est pour cela qu'on a défini une condition qui comme le montre la figure ci dessous, est affiché en une boite de dialogue tel que l'illustre la figure 3.18 qui représente le message d'erreur défini pour la clé.

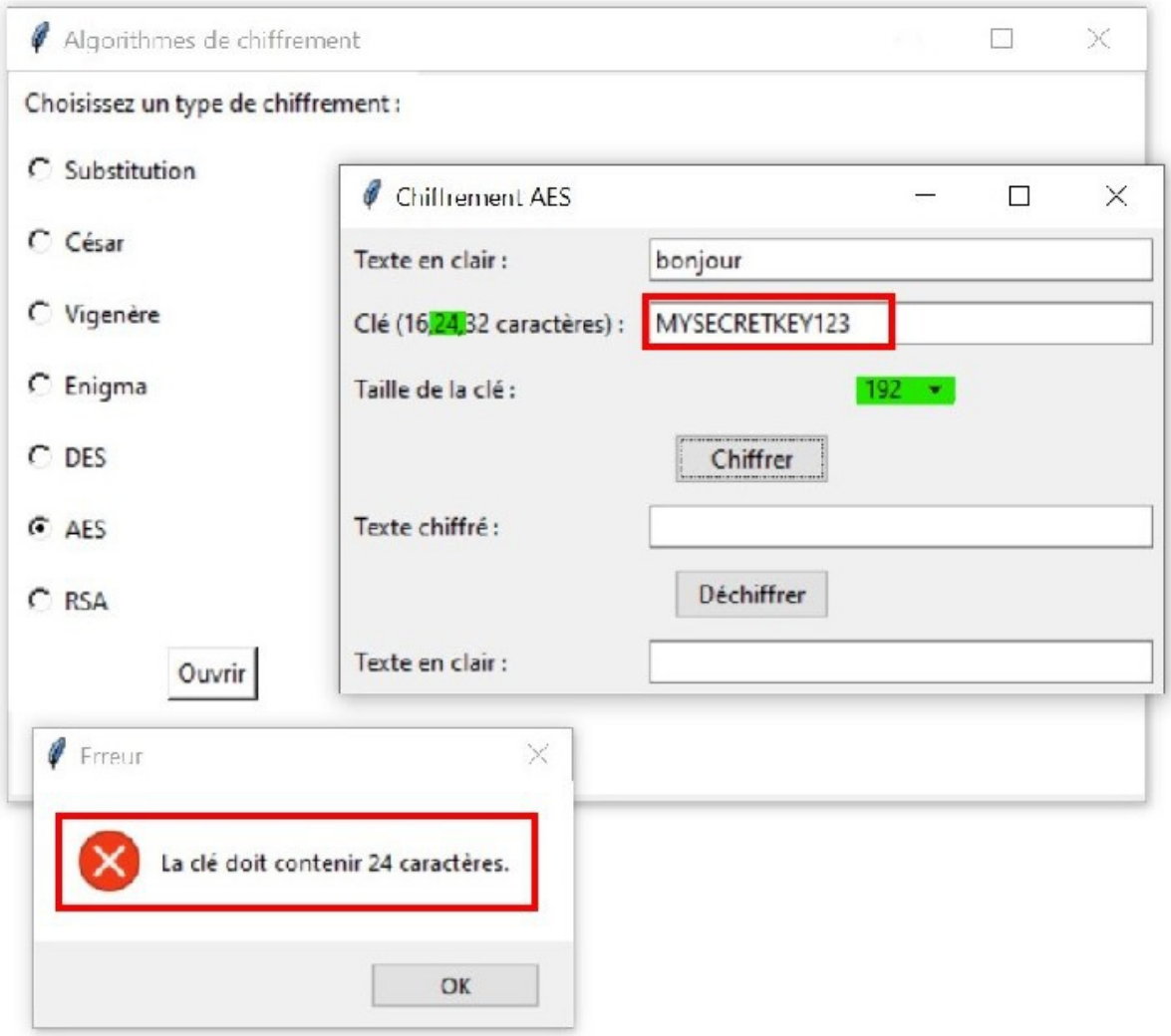

FIGURE 3.18 – Message d'erreur concernant la clé de l'AES

Le message indique que la clé doit être de 24 caractères pour celle de 192 bits, et de même pour les autres. marquant ainsi l'importance de la longueur de la clé de l'AES qui tant qu'elle ne respecte pas le nombre de caractères spécifiés, tant que le processus de chiffrement ne se produit pas.

3.3.2 Algorithme de chiffrement asymétrique

Nous avons implémenté le RSA comme chiffrement asymétrique dans notre application fonctionnant en générant des clés publiques pour le chiffrement et privés pour le déchiffrement. La figure 3.19 illustre un extrait du code Python concernant le processus de génération des clés.

```
n = p * q
phi_n = (p - 1) * (q - 1)

if math.gcd(e, phi_n) != 1:
    messagebox.showerror("Erreur", "L'exposant public e n'est pas premier avec φ(n)")
    return

d = self.mod_inverse(e, phi_n)

messagebox.showinfo("Clés générées", f"Clé publique (e, n): ({e}, {n})\nClé privée (d, n): ({d}, {n})")
```

FIGURE 3.19 – Extrait du code relatif a la Génération des Clés RSA

Cet extrait démontre les calculs nécessaires pour la générations des clés public et privé, a partir des nombres premiers p, q et de l'exposant public e qui doivent répondre a leur condition, dans le cas contraire un message d'erreur est affiché, on prend pour exemple le cas ou p et q ne sont pas relativement premier ou distincts comme le montre la figure 3.20 qui reflète le message d'erreur concernant les nombres premier p et q.

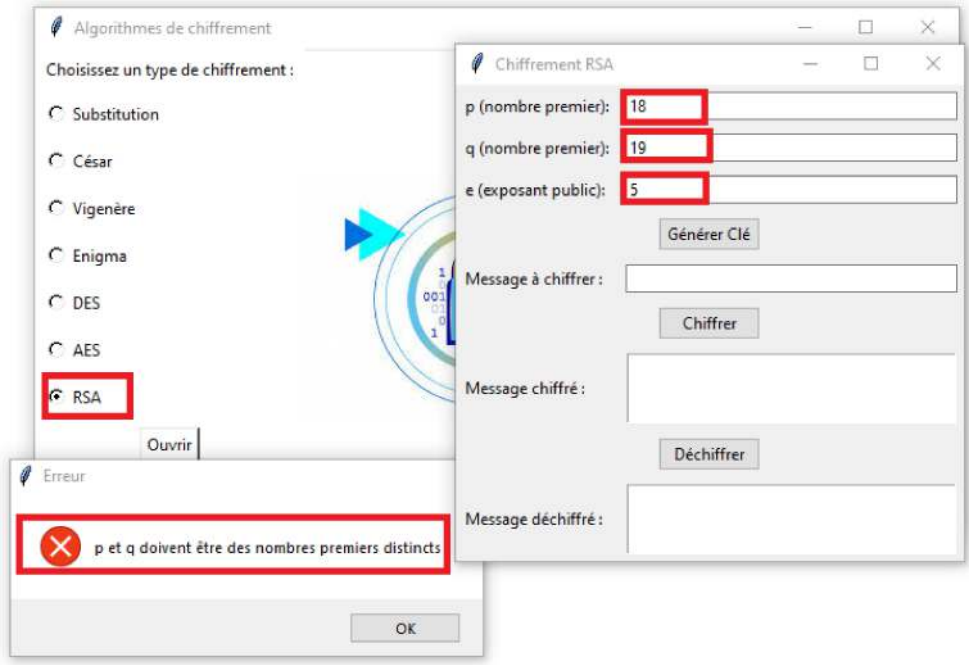

FIGURE 3.20 – Message d'erreur concernant la génération des clés du RSA

La boite de dialogue affiche un message, indiquant que p et q doivent être premiers et distincts pour que l'utilisateur corrige l'erreur. En entrant des nombres validés et en appuyant sur le bouton "générer les clés ". S'affiche une boite de dialogue qui comprend les clés générés tel que l'illustre la figure ci dessous.

La figure 3.21représente les clés générés dans une boite de dialogue.

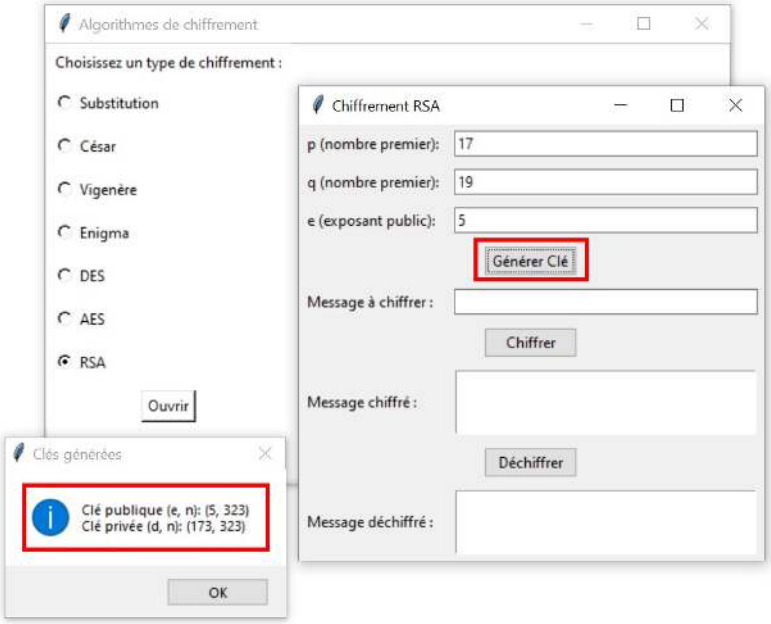

FIGURE 3.21 – Génération des clés du RSA

La boite de dialogue affiche la clé public qui sert a chiffré le message, ainsi que la clé privé permettant de déchiffrer le message selon les nombres premiers entrés, on pourras ainsi chiffré le mot "bonjour" en entrant le texte clair, puis en appuyant sur le bouton chiffrer, s'affiche le texte chiffré comme l'illustre la figure 3.22 qui reflète un exemple de chiffrement en utilisant l'algorithme RSA.

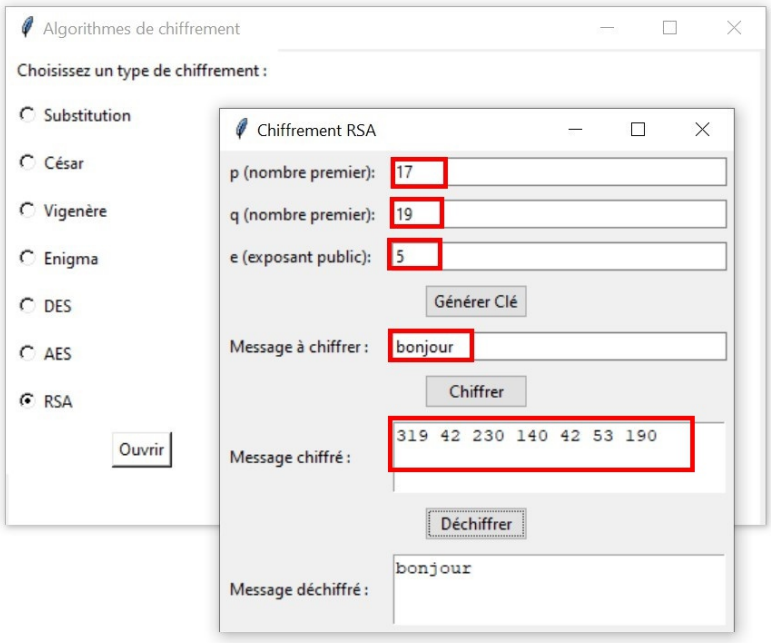

FIGURE 3.22 – Exemple de chiffrement et de Déchiffrement en utilisant le RSA

On obtient une série de nombres qui représente le texte chiffrer et pour récupérer le message original, il suffit d'appuyer sur le bouton déchiffrer.

3.4 Analyse comparative des résultat d'implémentation

L'expérimentation des divers algorithmes de chiffrement présent sur l'interface en chiffrant le même texte clair, a mis en évidence les caractéristiques et les performances de chaque algorithme allant des classiques tel que la substitution, César et Vigenére qui ont produit des textes chiffrés relativement simples et des motifs de répétition, jusqu'aux moderne comme l'Enigma, le DES, l'AES et le RSA qui ont présenter des résultat plus complexes. Ces variations significatives exposent l'évolution de la complexité et de la sécurité, À mesure que l'on évolue des méthodes traditionnelle vers celles plus moderne.

3.5 Conclusion

Dans ce chapitre nous avons abordé la conception, l'implémentation et l'analyse des algorithmes de chiffrement symetriques et aymetriques, prenant en compte les outils matériel et logiciel, ainsi que les bibliothèques utilisées pour le développement de l'application. Nous avons explorer l'environnement d'implémentation en expérimentant les différents algorithmes, mettant ainsi leurs caractéristiques et leur limites en évidence, pour ensuite effectuer une analyse comparative des résultat d'application en soulignant l'évolution des méthodes cryptographiques, tant pour la sécurité et tant pour la complexité. En conclusion, ce chapitre a permis une exploration approfondie de notre application qui s'avère être particulièrement enrichissante, permettant a l'utilisateur une compréhension détaillées du fonctionnement et des caractéristiques des techniques de chiffrement mis en oeuvre.

Conclusion Générale

Ce mémoire a exploré en profondeur l'univers fascinant de la cryptographie, retraçant son riche passé historique, décortiquant ses principes fondateurs et analysant en détail les techniques de chiffrement symétriques et asymétriques, anciennes et modernes. L'implémentation d'une interface graphique interactive en Python a permis de concrétiser ces concepts théoriques, offrant aux utilisateurs une expérience d'apprentissage immersive et ludique.

Au fil des chapitres, nous avons acquis une compréhension approfondie des enjeux cruciaux liés à la sécurité des données dans notre ère numérique interconnectée. L'importance de choisir judicieusement les algorithmes de cryptage les plus robustes et de rester vigilant face aux menaces émergentes a été soulignée.

L'un des objectifs principaux de notre application est de fournir un outil interactif et pédagogique qui facilite la compréhension et la comparaison des différentes méthodes de cryptographie, tout en offrant une expérience pratique aux utilisateurs. Dans le cadre éducatif, cet outil permet aux étudiants d'apprendre et d'utiliser ces algorithmes sans entrer dans les détails techniques complexes. En simplifiant l'accès à ces concepts, notre application vise à rendre l'apprentissage de la cryptographie plus accessible et engageant, aidant ainsi les utilisateurs à développer une solide compréhension des principes de sécurité des données.

Bien que ce projet ait atteint ses objectifs, il convient de garder à l'esprit que le domaine de la cryptanalyse est en constante évolution. L'avenir de la cryptanalyse est passionnant et stimulant. À mesure que la technologie progresse, les outils et techniques utilisés pour briser les algorithmes de chiffrement évoluent également. La bataille constante pour la sécurité des informations se poursuivra, et les cryptographes devront garder une longueur d'avance sur les pirates. L'informatique quantique, l'intelligence artificielle, la blockchain, la cryptographie post-quantique et la cryptographie hybride sont autant de technologies qui joueront un rôle important dans l'avenir de la cryptanalyse. Il appartient aux cryptographes de développer de nouveaux algorithmes de chiffrement capables de résister aux attaques de ces technologies et de garantir la sécurité des données.

En ce qui concerne les perspectives futures, notre application évoluera pour prendre en compte les menaces émergentes et les avancées technologiques. Nous envisageons d'intégrer des modules sur la cryptographie post-quantique et les techniques de cryptanalyse assistées par l'intelligence artificielle. De plus, l'application pourra inclure des scénarios de simulation pour illustrer les attaques potentielles et les méthodes de défense correspondantes. En restant à la pointe des développements en cryptographie, notre outil continuera d'offrir une plateforme éducative pertinente et à jour, préparant les étudiants à relever les défis de la sécurité des données dans un monde en constante évolution.

Bibliographie

[1] Chiffrement par bloc et par flot, consulté le 10/04/2024. Image. URL : https://www.thibautprobst.fr/posts/block-cipher-modes-operation/#chiffrement-par-bloc-et-par-flot.

[2] Chiffrement par substitution, consulté le 09/04/2024. Image. URL : https://www.wikiwand.com/fr/Chiffrement_par_substitution.

[3] Chiffrement par transposition, consulté le 10/04/2024. Image. URL : https://www.apprendre-en-ligne.net/crypto/transpo/paths.gif.

[4] Crypto asymétrique, consulté le 09/04/2024. Image. URL : http://david.carella.free.fr/images/crypto/crypto-asymetric-fr.png.

[5] Crypto symétrique, consulté le 09/04/2024. Image. URL : http://david.carella.free.fr/images/crypto/crypto-symetric-fr.png.

[6] Data encryption standard (des), consulté le 16/04/2024. Image. URL : https://media.geeksforgeeks.org/wp-content/uploads/20200306122641/DES-11.png.

[7] Image de cesar, consulté le 10/04/2024. Image. URL : https://commons.wikimedia.org/wiki/File:ROT13.png.

[8] la machine enigma, consulté le 12/04/2024. Image. URL : https://www.apprendre-en-ligne.net/crypto/Enigma/.

[9] Limitations classical cryptography : Classical ciphers limitations (cours lcs21001), consulté le 12/04/2024. URL : https://www.studocu.com/row/document/lira-university/computer-science/limitations-classical-cryptography/83129192.

[10] Principe rsa, consulté le 18/04/2024. image. URL : https://www.apprendre-en-ligne.net/crypto/rsa/index.html.

[11] Sécurisez vos données avec la cryptographie, consulté le 18/04/2024. URL : https://openclassrooms.com/en/courses/1757741-securisez-vos-donnees-avec-la-cryptographie.

[12] Tableau de chiffrage de vigenere, consulté le 10/04/2024. Image. URL : https://dcoqueux.github.io/vigenere/img/aide/tableau_chiffrage.svg.

[13] Tout savoir sur les différents types de protocoles de sécurité, consulté le 18/04/2024. URL : https://eays-consulting.com/differents-types-protocoles-securite/.

[14] T. Allançon. Le chiffre de césar, consulté le 08/04/2024. Haltode, mai 2014. Modifié le 14 décembre 2015.

[15] T. Allançon. Le chiffre de vigenère, consulté le 08/04/2024. Haltode, mai 2014. Modifié le 20 décembre 2015.

[16] J.-P. Aumasson. *Introduction à la Cryptographie*. 2011. URL : https://www.cryptologie.net/document.php?id=3403.

[17] G. Baumslag, B. Fine, and M. Kreuzer. *A course in mathematical cryptography*. De Gruyter, 2015.

[18] F. Z. Benidris. *Cryptographie : Polycopié de cours et exercices corrigés*. Faculté des Sciences Exactes et d'Informatique, Département de Mathématiques et Informatique, Université de Mostaganem, 2020. 3ème année licence - SI.

[19] M.-A. Bir and L. Dahmouni. Étude et implémentation d'algorithmes de chiffrement à clé secrète et à clé publique : Application au cryptage de la parole. 2018. Spécialité Automatique et Systèmes.

[20] Collège et Lycée International de Valbonne, consulté le 08/04/2024 . L'histoire de la cryptographie, décembre 2021. URL : https://webmedias.ac-nice.fr/colivensemble/2021/12/16/lhistoire-de-la-cryptographie/.

[21] W. Easttom. *Modern Cryptography : Applied Mathematics for Encryption and Information Security*. Georgetown University. Ebook.

[22] H. Ferradi. Cours de cryptographie. URL : https://www.di.ens.fr/~ferradi/cours.pdf.

[23] S. Ferradi. Introduction à la cryptographie (cours 4) : Chiffrement par bloc (aes), consulté le 16/04/2024. URL : https://www.di.ens.fr/~ferradi/coursAES.pdf.

[24] G. Florin and S. Natkin. Les techniques de cryptographie. *Journal de la Cryptographie*, 12(3) :234–256, mars 2002.

[25] P. Guillot. *La cryptologie : L'art des codes secrets*.

[26] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. Version électronique edition, 1997.

[27] NinjaOne. Cryptographie : son utilité et ses 3 types principaux. URL : https://www.ninjaone.com/cryptographie-son-utilite-et-ses-3-types-principaux/.

[28] S. Renard. *Cryptographie et sécurité informatique*. Editions Eyrolles, 2017.

[29] B. Schneier. *Cryptographie appliquée (2ème éd.) : Protocoles, algorithmes et codes sources en C*. Z-Library, 2000.