وزارة التعليم العالي والبحث العلمي

Ministere de l'enseignement superieur et de la recherche scientifique

جامعة عبد الحميد بنا باديس مستغانم

Universite abdl hamid ibn badis mostaganem

كلية العلوم والتكنولوجيا

Faculté des sciences de la technologiques

قسم الهندسة الكهربائية

Département de Génie Electrique

MEMOIRE DE FIN D'ETUDEMASTER ACADEMIQUE

Filière : Télécommunications
Spécialité : Système des Télécommunications

Thème

Etude Et Implémentation De L'algorithme De Chiffrement AES

Présenté par :

- M^elle^ Chaabane Nour El Houda
- M^elle^ Yaagoub Nihad

Devant le jury composé de :

Président : ***Mr.*** *BENAOUALI MOHAMED*

Encadrant : ***Mr.*** *ZELLAGUI AMINE*

Examinateur : ***Mme.*** *BENCHALLEL AMEL*

Soutenu le 27 /06/2024

Année Universitaire :2023/2024

REMERCIEMENTS

Nous remercions Dieu le tout puissant et miséricordieux, qui nous a donné durant toutes ces années, la santé, le courage et foi en nous-même pour pouvoir avancer.

Nous tenons à remercier notre encadreur, Monsieur ***Zellagui Amine****, pour ses précieux conseils et son aide durant toute la période de notre travail. Nous voudrions également lui témoigner notre gratitude pour sa patience et son soutien, qui nous ont été précieux pour mener notre travail à bon port.*

Nous exprimons également nos gratitudes aux membres du jury ***Mr BENAOUALI MOHAMED*** *et* ***Mme BENCHALLEL AMEL*** *qui nous ont honorés en acceptant de juger ce modeste travail.*

En fin, nous tenons également à remercier toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

Dédicace

Je dédie Ce mémoire à …

***Je** remercie ALLAH qui est toujours présent avec moi dans*

***Le** Meilleur et dans la pire.*

***A** ce qu'est toujours mon meilleur exemple dans La vie : mon grand-père « **Ben dhaiba** », pour les sacrifices qu'il a consentis pour mon éducation et pour l'avenir qu'il n'a cessé d'offrir,*

***Au** symbole de douceur, de tendresse, d'amour et affection, et grâce au sens de devoir et aux sacrifices immenses qu'elle*

A consentis : ma mère « Signe d'amour et de tendresse », j'ai pu arriver à réaliser ce travail.

A** mes chères tantes **: Noura, Hafida

***A** ma binôme « **Yaagoub Nihad** » j'ai trouvé l'entente que j'ai besoin*

A** Mes copines d'amour: **Meriem, Razika

***A** tous mes collègues de génie électrique, et à tous mes enseignants*

Houda.

Dédicace

Je dédie Ce mémoire à …

Je** remercie ALLAH qui est toujours présent avec moi dans **Le** Meilleur **et dans la pire

***J**e dédie cette mémoire à ma cher mère et mes chers frères qui m'ont tous donner et qui m'ont aidé dans la réalisation de ce mémoire que dieux les protège.*

***À** mes grands-parents ma chère tante et à toute la famille du grand au petit*

***A** ma binôme « **Chaabane Nour El Houda** » j'ai trouvé l'entente que j'ai besoin*

***A** tous mes collègues de génie électrique, et à tous mes enseignants*

Nihad.

Résumé

Avec l'évolution d'Internet, la sécurité des informations numériques est devenue cruciale. Le chiffrement, notamment via l'algorithme AES (Advanced Encryption Standard), protège les données contre tout accès non autorisé en les rendant illisibles sans la clé appropriée. L'efficacité des algorithmes de chiffrement dépend de la taille de la clé, de la complexité mathématique, de la vitesse et de la résistance aux attaques. Dans ce travail, nous avons amélioré la sécurité de l'AES en utilisant des cartes chaotiques, en modifiant deux étapes du processus de chiffrement, et avons comparé ses performances et sa sécurité avec celles de l'AES, RC4 (Rivest Cipher 4) et RC6 (Rivest Cipher 6). L'AES modifié, l'AES et le RC6 offrent un bon compromis entre performance et sécurité, tandis que le RC4, bien que rapide, présente des vulnérabilités. Le choix de l'algorithme dépend des exigences spécifiques de l'application, mettant en évidence l'équilibre nécessaire entre performance et sécurité.

Mots-clés : Cryptographie, AES, chaotique, chiffrement, sécurité.

ملخص

مع تطور الإنترنت أصبح أمن المعلومات الرقمية أمرًا بالغ الأهمية. التشفير، وخاصة عبر خوارزمية AES، يحمي البيانات من الوصول غير المصرح به عن طريق جعلها غير قابلة للقراءة بدون المفتاح المناسب. تعتمد فعالية خوارزميات التشفير على حجم المفتاح والتعقيد الرياضي والسرعة ومقاومة الهجمات. في هذا العمل، قمنا بتحسين أمان AES باستخدام خرائط فوضوية لتعديل خطوتين من عملية التشفير، ومقارنة أدائها وأمانها مع AES RC4 وRC6. توفر AES وAES وRC6 المعدلة حلاً وسطًا جيدًا بين الأداء والأمان، في حين أن RC4، على الرغم من سرعته، به نقاط ضعف. يعتمد اختيار الخوارزمية على متطلبات التطبيق المحددة، مع تسليط الضوء على التوازن الضروري بين الأداء والأمان.

الكلمات المفتاحية: التشفير، AES، الفوضى، التشفير، الأمن.

Abstract

With the evolution of the Internet, the security of digital information has become crucial. Encryption, notably through the AES (Advanced Encryption Standard) algorithm, protects data from unauthorized access by making it unreadable without the appropriate key. The effectiveness of encryption algorithms depends on key size, mathematical complexity, speed, and resistance to attacks. In this work, we have improved the security of AES using chaotic maps, modifying two steps of the encryption process, and compared its performance and security with AES, RC4 (Rivest Cipher 4), and RC6 (Rivest Cipher 6). The modified AES, AES (Advanced Encryption Standard), and RC6 offer a good compromise between performance and

security, while RC4, although fast, presents vulnerabilities. The choice of algorithm depends on the specific requirements of the application, highlighting the necessary balance between performance and security.

Keywords: Cryptography, AES, chaotic, encryption, security.

Sommaire

Chapitre 3 : L'ALGORITHME DE CHIFREMENT AES

Les Listes des figures

Les Listes des tableaux

Glossaire

DES Data Encryption Standard

AES Advanced Encryption Standard

RSA Revenu de Solidarité Active

ECB Electronic Code Book

CBC Cipher Block Chaining

OFB Output Feedback

CFB Cipher Feedback

XOR Exclusive OR

IDEA International Data Encryption Algorithm

FIPS Federal Information Processing

RC4 Rivest Cipher 4

RC6 Rivest Cipher 6

S-Box Substitution-Box

Introduction générale

Introduction général :

Dans les temps anciens, la communication et l'échange d'informations étaient des défis majeurs pour les humains, avec la protection des informations comme préoccupation primordiale. Avec l'avènement de la technologie et des communications, cette complexité s'est accrue, menant à une révolution dans ce domaine et à l'émergence de nouveaux défis en matière de sécurité. Les menaces telles que les virus informatiques et l'accès non autorisé sont devenues des préoccupations majeures. Pour répondre à ces défis, il est crucial de développer des systèmes de sécurité robustes et efficaces adaptés à notre époque et à l'évolution technologique. Ces systèmes permettent aux humains d'échanger des informations de manière sûre et confidentielle, en utilisant des technologies avancées de protection et de cryptage pour assurer la confidentialité et la sécurité des données, que ce soit sur des appareils électroniques ou lors de leur transmission sur Internet.

La cryptographie joue un rôle central dans la sécurisation des communications et des données contre ces menaces. Elle propose des techniques pour garantir la confidentialité, l'authentification et l'intégrité des informations. Fondée sur des concepts mathématiques et des paradigmes informatiques, la cryptographie résiste aux attaques potentielles et assure l'incorruptibilité des procédures. En fournissant des outils pour l'authentification, la confidentialité et la protection de l'intégrité, la cryptographie contribue de manière significative à la sécurité informatique.

Les algorithmes de chiffrement se divisent principalement en deux catégories : ceux à clé publique (comme RSA) et ceux à clé privée (comme DES, 3DES, AES). La cryptographie hybride combine les avantages de ces deux techniques.

Cependant, avec l'augmentation de la puissance de calcul, la cryptographie évolue continuellement pour faire face aux avancées des techniques de cryptanalyse. Certains algorithmes ont été remplacés par des versions plus sécurisées, telles que le remplacement du DES par le 3DES puis par l'AES. La sécurité du RSA est également soumise aux progrès en factorisation d'entiers.

L'objectif de ce travail est de mettre en œuvre l'algorithme de chiffrement avancé (AES), reconnu pour sa robustesse et sa popularité, afin d'assurer la sécurité et la protection des données sur les réseaux et dans les communications électroniques. L'AES représente l'un des algorithmes de chiffrement les plus puissants disponibles aujourd'hui. En résumé, l'étude et l'application de l'AES nous permettent d'assurer une protection efficace des données sensibles et confidentielles, renforçant ainsi la cybersécurité dans divers scénarios et applications.

Ce mémoire est composé de quatre chapitres :

Dans le premier chapitre, une introduction à la cryptographie.

Dans le deuxième chapitre, les algorithmes de chiffrement standards.

Dans Le troisième chapitre, l'algorithme de chiffrement AES

Dans le chapitre quatre, Implémentation de l'algorithme de chiffrement AES

Enfin, nous présentons une **conclusion générale** et quelques perspectives qui peuvent aider à l'amélioration des systèmes actuels.

Chapitre 1 : Introduction à la cryptographie

1.1 Introduction :

La cryptographie permet une communication sécurisée en présence de tiers malveillants, appelés adversaires. Le chiffrement utilise un algorithme et une clé pour transformer une entrée en une sortie chiffrée. La cryptologie apparaît ainsi comme un moyen adéquat pour garantir la sécurité de l'information en utilisant des principes mathématiques afin d'assurer la confidentialité, l'authenticité et l'intégrité des messages échangés, notamment sur des canaux de communication non sécurisés comme Internet.

Dans ce chapitre, nous présentons un aperçu des techniques de cryptographie classique et moderne, en mettant en lumière ses deux principales catégories : la cryptographie symétrique et la cryptographie asymétrique. Nous définissons les algorithmes de chiffrement les plus répandus, puis les outils et les techniques mathématiques utilisés pour concevoir un algorithme cryptographique, tels que la théorie des nombres et les cartes chaotiques.

1.2 Définition de cryptologie :

La cryptologie est un terme dérivé du grec : cryptos, qui signifie "secret", et logia, qui signifie "science". En réalité, il s'agit de la science du secret et elle ne peut être réellement considérée comme telle que depuis peu. La cryptologie englobe la cryptographie, l'écriture secrète, et la cryptanalyse, l'analyse des systèmes cryptographiques.

La cryptologie est considérée à la fois comme un art ancien et une science nouvelle. Elle était déjà utilisée par Jules César et apparaît dans l'Ancien Testament sous la forme du code Atbash. Cependant, elle est également une discipline nouvelle, car elle n'est devenue un sujet de recherche scientifique que depuis les années 1970.

La théorie des nombres, l'algèbre, la théorie de la complexité, la théorie de l'information, ainsi que les codes correcteurs sont des disciplines étroitement liées à la cryptologie. [1]

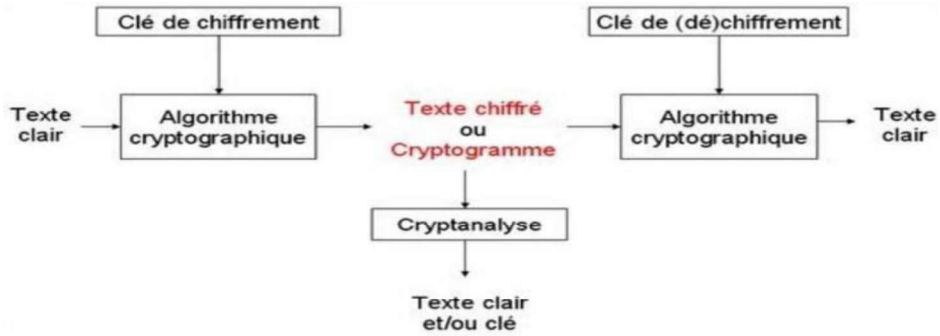


Figure 1.1 Protocole de chiffrement [36]

1.3 Définition de la cryptographie :

La cryptographie est une discipline mathématique qui utilise les mathématiques pour protéger et déchiffrer des données. De cette façon, elle permet de conserver des données confidentielles ou de les transmettre sur des réseaux non sécurisés (comme Internet), de manière à ce qu'aucune autre personne que le destinataire ne puisse les consulter.

Si la cryptographie vise à garantir la sécurité des données, la cryptanalyse se concentre sur l'analyse des informations chiffrées pour en dévoiler le secret. La cryptanalyse traditionnelle combine habilement raisonnement analytique, utilisation d'outils mathématiques, recherche de modèles, patience, détermination et chance. On désigne également ces cryptanalystes sous le nom de pirates [2]

Cryptologie = Cryptographie + Cryptanalyse.

1.4 L'usage de la cryptographie :

Traditionnellement, la cryptographie a été employée pour dissimuler les messages de certains utilisateurs. Cette pratique est d'autant plus captivante de nos jours, car les échanges en ligne se déroulent dans des infrastructures où la fiabilité et la confidentialité ne peuvent être assurées. Actuellement, la cryptographie est utilisée non seulement pour préserver la confidentialité des données, mais également pour assurer l'intégrité et l'authenticité des informations. [3]

- **Confidentialité** : Cela implique de rendre les données inaccessibles à toute personne autre que les participants à la transaction.
- **Intégrité** : La vérification de l'intégrité des données implique de s'assurer que les informations n'ont pas été altérées lors de la transmission.
- **Authentification** : L'objectif est de garantir l'identité de l'utilisateur, c'est-à-dire de s'assurer que chaque correspondant est bien celui qu'il prétend être. Le contrôle d'accès peut être assuré, par exemple, en utilisant un mot de passe crypté pour permettre seulement aux personnes autorisées d'accéder à la ressource.
- **Non-répudiation** : Cela garantit qu'aucun des interlocuteurs ne peut nier la transaction. [4]

1.5 Histoire de la cryptographie :

L'histoire de la cryptographie remonte à l'Antiquité, où les premières techniques de chiffrement étaient utilisées pour protéger les secrets militaires et diplomatiques. L'une des plus anciennes méthodes connues est le chiffre de César, employé par Jules César pour sécuriser ses communications. Cette méthode simple consistait à décaler les lettres de l'alphabet d'un nombre

fixe de positions. Au Moyen Âge, des systèmes plus sophistiqués ont été développés, tels que le chiffre de Vigenère, qui utilisait une série de différentes substitutions pour augmenter la complexité du message chiffré. La cryptographie a connu une transformation radicale au XXe siècle avec l'invention de la machine Enigma par les Allemands, utilisée pendant la Seconde Guerre mondiale. La rupture du code Enigma par les Alliés, notamment grâce aux travaux d'Alan Turing et de son équipe, a marqué un tournant dans l'histoire de la cryptographie. À l'ère moderne, l'avènement de l'informatique a révolutionné le domaine, avec des algorithmes de chiffrement avancés comme RSA et AES, basés sur des concepts mathématiques complexes. Aujourd'hui, la cryptographie est essentielle pour la sécurité des communications numériques, garantissant la confidentialité, l'intégrité et l'authenticité des informations échangées sur des réseaux mondiaux tels qu'Internet. [5]

1.6 L'évolution de la Cryptographie :

Depuis les premiers systèmes de codage utilisés par les civilisations anciennes jusqu'aux algorithmes sophistiqués d'aujourd'hui, la cryptographie a toujours été un outil essentiel pour sécuriser les communications. Cette évolution peut être divisée en trois grandes périodes : la cryptographie classique, la cryptographie moderne et la cryptographie quantique. Chacune de ces périodes a apporté des innovations majeures qui ont transformé la manière dont nous protégeons et échangeons des informations. [6]

1.6.1 Cryptographie classique :

Avant l'apparition des ordinateurs, la cryptographie classique a été développée et a posé les fondements de nombreux algorithmes symétriques encore utilisés aujourd'hui. Les crypto-systèmes classiques sont classés principalement en deux catégories : le chiffrement par substitution et le chiffrement par transposition. [7]

1.6.1.1 Chiffrement par substitution :

Les substitutions consistent à remplacer des symboles ou des groupes de symboles par d'autres symboles ou groupes de symboles dans le but de créer de la confusion.

Le **chiffrement par substitution** se divise en deux sous-catégories : le chiffrement mon alphabétique et le chiffrement poly alphabétique [8]

- **Chiffrement mono alphabétique** : Ce type de chiffrement, comme le chiffrement de César, implique une substitution fixe pour chaque lettre du texte clair, décalant chaque lettre d'un certain nombre de positions dans l'alphabet. [10]
- **Chiffrement poly alphabétique** : Plus complexe, ce type utilise plusieurs alphabets de substitution. La substitution change en fonction de la position de chaque lettre dans le

texte clair, souvent déterminée par un mot-clé répétitif. Un exemple célèbre est le chiffre de Vigenère, où différentes substitutions sont appliquées selon un mot-clé. [10]

A. Chiffrement de César :

Le chiffrement par décalage est l'un des systèmes les plus anciens et simples. La clé secrète de chiffrement, aussi connue sous le nom de code de César, détermine le nombre de positions de décalage appliqué à chaque lettre de l'alphabet. Bien que ce système de chiffrement soit facile à mettre en œuvre, comme illustré dans la figure 1.2, il est également facile à décrypter. En effet, il est possible d'utiliser une méthode exhaustive qui teste toutes les valeurs possibles de la clé d. De plus, une analyse des fréquences d'apparition des lettres dans le message chiffré comparées aux fréquences typiques dans un texte en français, par exemple, peut faciliter la détermination de la valeur de d. Étant donné que la lettre la plus fréquemment utilisée en français est « e », il est souvent possible de la relier à la lettre la plus fréquente dans le message chiffré pour identifier d. [7]

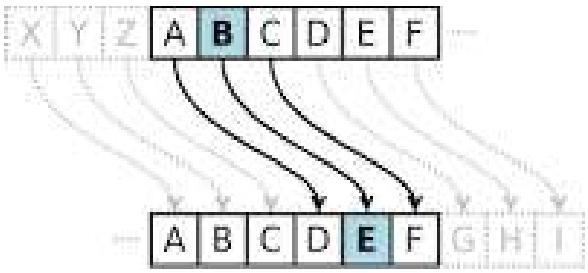


Figure 1.2 Chiffrement de César. [9]

B. Chiffrement de Vigenère :

Le chiffre de Vigenère représente une amélioration significative par rapport au chiffre de César. Il est puissant car il utilise non pas un, mais 26 alphabets décalés pour chiffrer un message. Ces décalages sont organisés dans ce qu'on appelle le carré de Vigenère. Ce système utilise une clé qui établit le décalage pour chaque lettre du message. [5]

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 1.3 Table de Vigenère [49]

Exemple : En utilisant la clé "BACHELIER", il est possible de chiffrer le texte "CHIFFRE DE VIGENÈRE" de manière à obtenir une longueur équivalente à celle du texte clair.

Clair	C	H	I	F	F	R	E	D	E	V	I	G	E	N	E	R	E
Clef	B	A	C	H	E	L	I	E	R	B	A	C	H	E	L	I	E
Décalage	1	0	2	7	4	11	8	4	17	1	0	2	7	4	11	8	4
Chiffré	D	H	K	M	J	C	M	H	V	W	I	I	L	R	P	Z	I

Tableau 1.1 Exemple de chiffrement de Vigenère [5]

1.6.1.2 Chiffrement par transposition :

Les transpositions consistent à mélanger les symboles ou les groupes de symboles d'un message clair suivant des règles prédéfinies pour créer de la diffusion. Ces règles sont déterminées par la clé de chiffrement. Une suite de transpositions forme une permutation. [11]

Les transpositions rectangulaires

Pour effectuer un chiffrement par transposition rectangulaire, on commence par se mettre d'accord sur un mot-clé. Choisissons pour notre exemple le mot TELECOM. On classe alors les lettres du mot TELECOM par ordre alphabétique, et on attribue à chaque lettre son numéro dans l'ordre alphabétique. Ainsi, on donne à A le numéro 1, au premier B le numéro 2, au deuxième B le numéro 3, au H le numéro 4, etc…. [11]

On crée ensuite un tableau de la façon suivante :

- La première ligne est constituée par les lettres de la clé ;
- La deuxième ligne est constituée par les numéros qui leur sont associés ;
- On complète ensuite le tableau en le remplissant avec les lettres du message à chiffrer. On écrit sur chaque ligne autant de lettres que de lettres dans la clé. Eventuellement, la dernière ligne n'est pas complète.

Par exemple, si on veut chiffrer " Je suis en Mostaganem ", le tableau que l'on construit est le suivant :

T	E	L	E	C	O	M
7	2	4	3	1	6	5
J	E	S	U	I	S	E
N	M	O	S	T	A	G
A	N	E	M			

Tableau 1.2 Les transpositions rectangulaires

Ensuite, on écrit d'abord le contenu de la colonne numérotée 1, puis le contenu de la colonne numérotée 2, etc… Le message chiffré obtenu est alors :

IT EMN USM SOE EG SA JNA

1.6.2 Cryptographie moderne :

La cryptographie moderne joue un rôle essentiel en garantissant que les communications et les transactions électroniques restent confidentielles et sécurisées. En combinant des mathématiques avancées avec des technologies informatiques innovantes, la cryptographie moderne propose des solutions sophistiquées pour protéger l'intégrité des données et assurer

l'authenticité des échanges dans un monde interconnecté et souvent vulnérable aux cybermenaces. [8]

Différents types de cryptographie moderne définissent ces algorithmes, et on peut distinguer deux approches :

- La cryptographie symétrique.
- La cryptographie asymétrique. [12]

1.6.2.1 Algorithme symétrique (chiffrement a clé privée) :

La cryptographie symétrique (ou cryptographie à clé secrète ou cryptographie conventionnelle) est la plus ancienne dans l'histoire de la cryptographie. Elle est très courante en raison de sa capacité à chiffrer rapidement en utilisant des clés de petite taille. On utilise la même clé pour chiffrer et déchiffrer, comme illustré dans la figure (1.4).

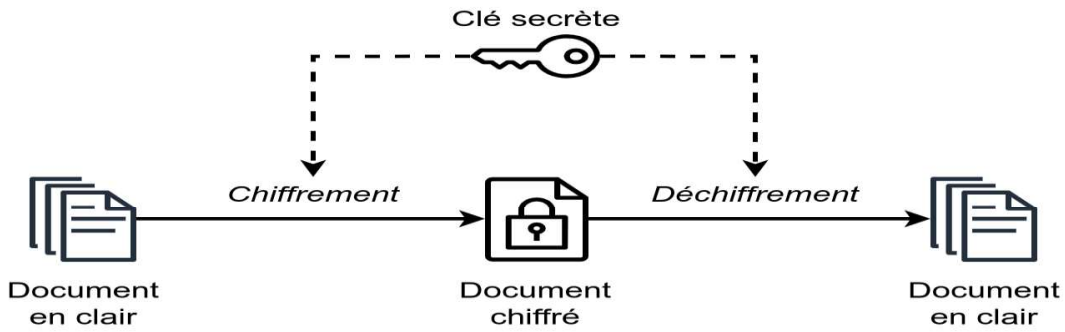


Figure 1.4 Chiffrement symétrique. [50]

La cryptographie symétrique se distingue par sa rapidité et sa facilité d'exécution, ainsi que par des mises en œuvre à la fois logicielles et matérielles, ce qui permet une accélération significative des débits et une utilisation massive. [13] En outre, le message (clair) est divisé en blocs distincts, chacun étant chiffré séparément. Cela donne accès à des systèmes de chiffrement extrêmement performants et extrêmement simples.

Les techniques de cryptage symétrique sont naturellement classées en deux catégories, le cryptage par bloc et le cryptage par flot, comme décrit ci-dessous :

- **Le cryptage par bloc** : Le chiffrement est réalisé sur des blocs de bits.
- **Le cryptage par flot (ou flux) :** Le chiffrement se déroule en permanence, bit par bit. [14]

A. Cryptage par flot (Stream Cipher) :

Dans un crypto système par flots, le cryptage des messages se fait caractère par caractère ou bit par bit, au moyen de substitutions générées aléatoirement, la taille de la clé est donc égale à la

taille du message utilisés lorsque l'information ne peut être traitée qu'avec de petites quantités de symboles à la fois, comme par exemple dans le cas où l'équipement n'a pas de mémoire physique ou une mémoire tampon très limitée. [12]

Quelques algorithmes de cryptographie symétrique par flot :

- A5 dans les smartphones afin de sécuriser les échanges sans fil entre le téléphone mobile et l'antenne relais la plus proche. [15]
- RC4 Un élément très utilisé dans les protocoles de sécurité mobile. [12]

B. Cryptage par blocs (Block Cipher) :

Contrairement aux chiffrements par flot qui agissent bit par bit, le chiffrement par bloc consiste à diviser chaque texte en clair en blocs de longueur fixe (typiquement de 64 ou 128 bits) et à les chiffrer avec une seule clé. Ces algorithmes sont généralement basés sur un modèle itératif. Leur fonctionnement repose sur une fonction F qui utilise une clé secrète k et un texte T de n bits. La fonction F est appliquée plusieurs fois (connue sous le nom de nombre de tours). À chaque itération, la valeur de la clé k change et le texte obtenu à partir de l'itération précédente est chiffré. La clé secrète k est utilisée pour dériver différentes sous-clés $k[i]$. [16]

Il existe quatre modes d'opération pour le chiffrement par bloc :

- Mode ECB (Electronic Codebook)
- Mode CBC (Cipher Block Chaining)
- Mode OFB (Output Feedback)
- Mode CFB (Cipher Feedback)

Ces modes déterminent la manière dont les blocs de texte sont encryptés et interconnectés pour renforcer la sécurité et la confidentialité des données.

Mode ECB :

Le mode de fonctionnement ECB (Electronic Codebook) ou livre de codes électronique est le plus simple et le plus rapide, ce qui constitue son principal avantage. Chaque bloc de bits du message est chiffré indépendamment des autres avec la même clé, comme illustré dans la figure (1.5).

Cependant, ce mode présente l'inconvénient majeur de produire des blocs chiffrés identiques pour des blocs de texte clair identiques. De plus, un attaquant potentiel peut altérer le message en chiffrant ou en supprimant des blocs pour modifier le sens du message initial. [16]

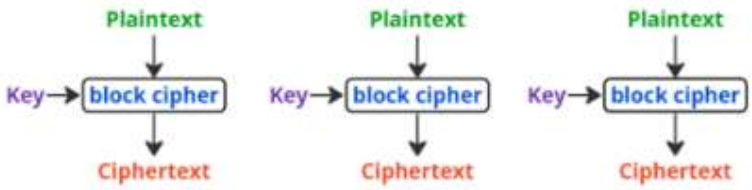


Figure 1.5 Le mode ECB **[35]**

Mode CBC :

Le mode CBC (Chaining of Cipher Blocks) ou enchaînement de blocs. La figure 1.6 illustre qu'il s'agit d'un vecteur initial aléatoire (IV) qui est d'abord sélectionné pour être associé au premier bloc par une opération XOR avant de chiffrer. Après que l'algorithme de chiffrement a chiffré le premier bloc de texte en clair, le texte chiffré obtenu est conservé dans un registre de rétroaction pour qu'il puisse être associé comme un vecteur initial avec le deuxième bloc de texte avant le chiffrement. [17]

En mode CBC, on utilise l'algorithme de déchiffrement pour déchiffrer un bloc , puis on le combine par ou exclusif avec le bloc chiffré précédent, respectivement avec le vecteur initial.

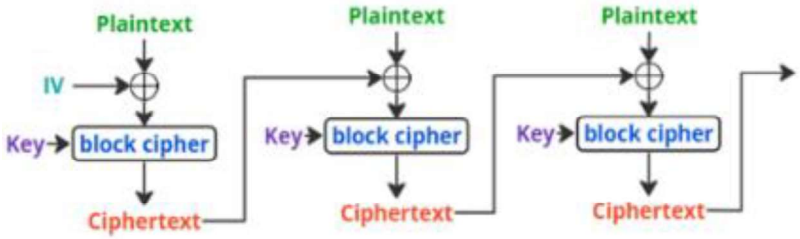


Figure 1.6 Le mode CBC**. [35]**

Mode OFB :

Le chiffrement à rétroaction de sortie, également connu sous le nom de mode OFB, est une variante du mode CFB. [19] Il est essentiel d'avoir un vecteur d'initialisation (IV) unique pour chaque utilisation du mode avec une clé spécifique. L'algorithme de chiffrement utilise cet IV pour générer le premier bloc de sortie. En appliquant un XOR entre ce bloc de sortie et le premier bloc de texte clair, on obtient le premier bloc de texte chiffré. Ensuite, la fonction de chiffrement est appliquée au bloc de sortie pour générer le bloc suivant de sortie. Ce processus de chiffrement est illustré dans la figure 1.7 [1]. Le processus de décryptage est identique à celui du chiffrement.

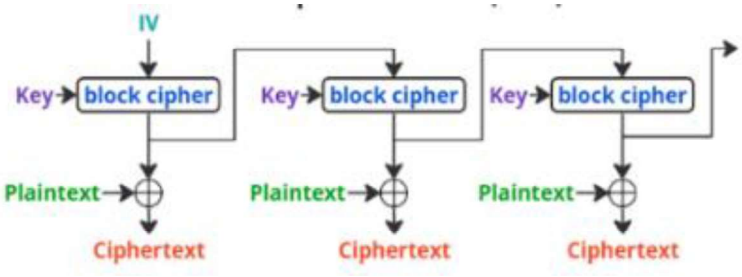


Figure 1.7 Le mode OFB**. [35]**

Mode CFB :

Le mode CFB (Cipher Feedback) ou chiffrement à rétroaction est similaire au mode OFB, mais cette fois-ci, le flot de bits pseudo-aléatoires est généré à partir des blocs chiffrés eux-mêmes, comme illustré dans la figure 1.8. Dans cette méthode, le premier bloc est un vecteur d'initialisation (VI) qui est chiffré à l'aide de l'algorithme de chiffrement choisi, puis combiné avec le texte clair via un XOR pour produire le texte chiffré à transmettre. Le résultat de ce bloc chiffré est ensuite réutilisé pour calculer le chiffrement du bloc suivant. En d'autres termes, chaque bloc est chiffré avec l'algorithme de chiffrement, puis rétrodiffusé dans la chaîne avec la clé via un XOR, comme illustré dans la figure 1.8. [18]

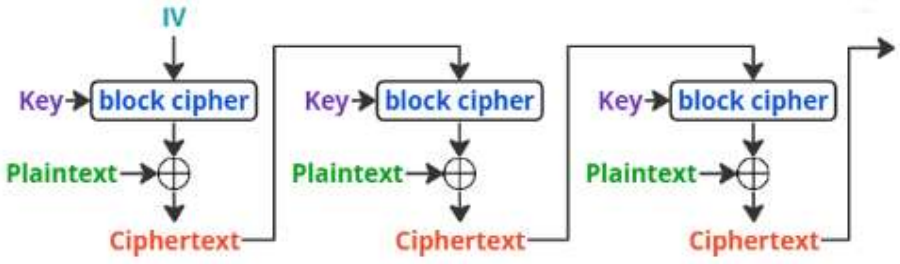


Figure 1.8 Le mode CFB. [35]

Exemple : Le chiffrement de Feistel.

Dans cette méthode de chiffrement, on divise un bloc de texte clair en deux parties ; on applique la transformation de ronde à l'une des deux parties, et le résultat est combiné avec l'autre moitié par ou exclusif. Les deux moitiés sont ensuite inversées afin de mettre en œuvre la prochaine ronde. Un avantage de ce genre d'algorithmes réside dans le fait que leur structure de chiffrement et de déchiffrement est similaire. [45] Selon nous, pour une clé d'entrée spécifique, ces fonctions sont les suivantes : [5]

entrée	f_1	sortie		entrée	f_2	sortie
00	→	01		00	→	11
01	→	11		01	→	00
10	→	10		10	→	00
11	→	01		11	→	01

Tableau 1.3 Les Fonctions de Feistel [20]

La fonction XOR (représentée par un + entouré d'un cercle) est utilisée pour "additionner" deux bits, comme indiqué dans le tableau ci-dessous. Il convient de souligner que l'opérateur XOR est son inverse. [20]

XOR	0	1
0	0	1
1	1	0

Tableau 1.4 fonction XOR [20]

Il convient de souligner que ni f1 ni f2 sont des bijections. Prenons l'exemple du message 1101 pour le chiffrer. G est la partie gauche du message à chiffrer, tandis que D est la partie droite. [20]

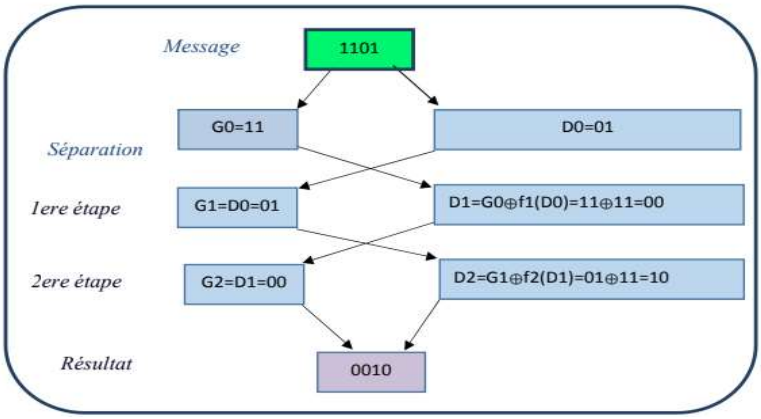


Figure 1.9 Schémas de Feistel. [20]

1.6.2.2 Algorithmes asymétrique (chiffrement a clé public) :

La cryptographie asymétrique ne nécessite pas d'échange préalable de secrets. Chaque utilisateur dispose de deux clés mathématiquement liées : l'une est privée et l'autre est publique.

Ce processus asymétrique signifie que ce qui est chiffré avec l'une des clés ne peut être déchiffré qu'avec l'autre. Le chiffrement des messages s'effectue avec la clé publique, tandis que la clé privée permet de déchiffrer les messages chiffrés à l'aide de la clé publique correspondante, comme illustré dans la figure 1.10. La clé publique est largement connue, tandis que la clé privée doit rester secrète, car elle seule permet de déchiffrer les messages. De plus, la sécurité de la clé privée ne peut pas être déduite de la validité de la clé publique. Lorsqu'une clé publique est validée, elle est utilisée pour générer une pré-clé partagée avec le serveur. Ce résultat est ensuite transmis au serveur. [21]

Dans le domaine mathématique, le chiffrement asymétrique repose sur des fonctions à sens unique, simples à calculer mais difficiles à inverser. Cela signifie qu'il n'est pas nécessaire de casser ce type de chiffrement par une attaque exhaustive essayant toutes les clés possibles. Au lieu de cela, il s'agit de trouver l'inverse de la fonction utilisée. La complexité calculatoire de cette tâche dépend des ressources de calcul disponibles et du temps nécessaire pour effectuer les calculs. [22]

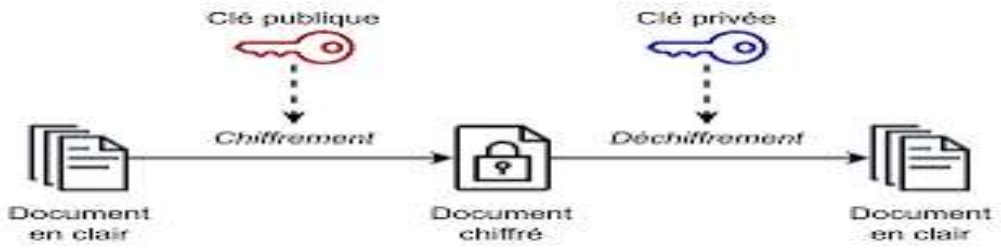


Figure 1.10 Chiffrement asymétrique [50]

Les principaux algorithmes utilisés dans la cryptographie asymétrique sont :

- **RSA :** Le RSA (Ron Rivest, Adi Shamir et Leonard Adleman) est le premier algorithme de chiffrement asymétrique à avoir été créé. Il a été mis au jour en 1977. Le RSA repose sur la théorie des nombres premiers et sa force réside dans le fait qu'il n'y a aucun algorithme qui serait capable de décomposer un nombre en facteurs premiers. [23]
- **ElGamal :** l'algorithme ElGamal est utilisé pour le chiffrement des données. Il est basé sur la difficulté de calculer les logarithmes discrets. [24]

1.6.3 Cryptographie quantique :

La cryptographie quantique est l'utilisation des caractéristiques de la physique quantique pour développer des protocoles de cryptographie qui permettent d'atteindre des niveaux de sécurité qui sont démontrés ou conjecturés en utilisant uniquement des phénomènes classiques

(c'est-à-dire non-quantiques). La distribution quantique de clés est un exemple majeur de cryptographie quantique, qui permet de transmettre une clé de chiffrement secrète entre deux interlocuteurs distants tout en garantissant la sécurité de la transmission grâce aux lois de la physique quantique et de la théorie de l'information. Par la suite, cette clé secrète peut être employée dans un algorithme de chiffrement symétrique pour protéger et décrypter des informations confidentielles.

La cryptographie quantique utilise des quantités massives d'information et des clés de chiffrement extrêmement longues afin d'assurer une sécurité inégalée dans les communications quantiques. Elle est révolutionnaire pour l'avenir, car elle promet de transformer radicalement la sécurité des communications. [25]

1.7 Les outils mathématique utilisent pour la cryptographie :

1.7.1 La théorie des nombres :

La théorie des nombres est une grande et intéressante discipline mathématique, parfois connue sous le nom d'arithmétique supérieure, qui concerne l'étude des caractéristiques des nombres entiers. Les premiers nombres et la factorisation des premiers nombres jouent un rôle crucial. Dans la théorie des nombres, de la même manière que plusieurs fonctions.

La cryptographie consiste à transférer des informations de manière sécurisée, permettant ainsi à aucun tiers indésirable de comprendre le message. On l'emploie depuis des milliers d'années. La théorie des nombres est inséparable de la cryptographie.

Dans les prochaines lignes, nous allons mentionner certaines caractéristiques de la théorie des nombres, telles que les opérations binaires, les nombres premiers, la congruence, l'algorithme euclidien et l'exponentiel. [26]

1.7.1.1 Les opérations binaires :

Nous nous concentrerons principalement sur trois opérations en cryptographie : ET, OU et XOR**. [46]**

Operateur ET :

La fonction ET est obtenue par la multiplication des Entrées F= A • B. La table de vérité et le symbole associés à cette fonction sont **:**

A	B	F= A • B
0	0	0
0	1	0
1	0	0
1	1	1

Tableau 1.5 L'opérateur ET [46]

Operateur OU :

La fonction OU est obtenue par la somme des entrées du système F = A + B. La table de vérité et le symbole associés à cette fonction sont : [46]

A	B	F= A + B
0	0	0
0	1	1
1	0	1
1	1	1

Tableau 1.6 L'opérateur OU. [46]

Operateur XOR (OU EXCLUSIF) :

La fonction OU EXCLUSIF ne vaut 1 que si les deux entrées sont différentes. Elle s'écrit :

$$F=A \oplus B = A \bullet B + A \bullet B$$

La table de vérité et le symbole associés à cette fonction sont : [46]

A	B	F= A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

Tableau 1.7. L'opérateur XOR. [46]

1.7.1.2. Les nombres premiers :

Les premiers nombres sont les entiers positifs dont les facteurs sont seulement 1, et le nombre entier lui-même. Par exemple, les facteurs de 6 sont 1, 2, 3 et 6, ce qui représente un total de quatre facteurs. Cependant, les facteurs de 7 ne sont que 1 et 7, ce qui représente deux facteurs au total. Donc, 7 est un premier nombre, tandis que 6 ne l'est pas, c'est un nombre composite. Il est important de se rappeler que 1 n'est ni premier ni composite.

Les premiers nombres sont aussi les nombres qui ne sont divisibles que par 1 ou le nombre lui-même. On le définit également comme un nombre positif ou entier, qui n'est pas le produit de deux autres entiers positifs. Il n'y a pas de formule spécifique pour cela.

À part la recherche de ses facteurs, il est impossible de déterminer si un nombre est premier ou non (à moins d'une certaine limite). [26]

1.7.1.3 Congruence :

La question de la congruence des opérations sur le module revêt une grande importance et est souvent utilisée dans les algorithmes cryptographiques contemporains. En mathématiques, le symbole ≡ est employé pour exprimer la congruence.

Deux nombres a et b sont appelés "congruents modulo "n :

- Si a mod n = b mod n, alors a ≡ b (mod n).
- En cas de congruence entre deux nombres modulo n, la différence entre a et b sera un multiple de n. [26]

1.7.1.4. Algorithme Euclidien :

L'algorithme d'Euclide est probablement l'un des plus anciens et les plus célèbres.

Ce calcul du plus grand diviseur commun (PGCD) de deux entiers a et b est une méthode. Il offre aux ordinateurs la possibilité d'accomplir diverses tâches simples en théorie des nombres.et constitue aussi la fondation d'algorithmes plus complexes en théorie des nombres. Cela consiste principalement à répéter en permanence l'algorithme de division pour les entiers. Le but est de répartir le diviseur par le reste de manière répétée jusqu'à ce que le reste soit égal

à 0. Le dernier élément non nul de cet algorithme est le PGCD. L'exemple suivant présente l'algorithme qui permet de déterminer le PGCD de 102 et 38. [26]

1.7.1.5 Algorithme d'Euclide étendu :

L'algorithme euclidien étendu est une méthode qui permet de déterminer les entiers x et y, comme :

$$ax + by = PGCD(a,b).$$

Étant donné a et b, le lemme de Bézout assure l'existence de tels entiers.

On peut considérer l'algorithme euclidien étendu comme l'opposé de l'exponentiation modulaire. C'est en inversant les étapes de l'algorithme euclidien que ces nombres entiers x et y peuvent naître. Le concept consiste à débuter par le PGCD et à revenir en arrière de manière récurrente. Afin d'accomplir cela, il faut traiter les nombres comme des variables jusqu'à ce qu'une expression soit une combinaison linéaire des nombres initiaux. [26]

1.7.1.5 Exponentielle modulaire :

La fonction exponentielle croissant très rapidement, on dépasse facilement les limites de l'arithmétique entière de la machine, et on ne profite guère de cette accélération. Calculons plutôt $x^e \bmod m$, de façon à rester dans ces limites. Il suffit de placer un %m après chaque opération

L'exponentielle modulaire s'avère utile à certaines méthodes de cryptographie. [47]

1.7.2 Les systèmes chaotique :

1.7.2.1 Description du chaos :

Un chaotique est un système déterministe qui adopte un comportement similaire à celui des systèmes non linéaires, mais qui possède certaines caractéristiques distinctes .Ce système se distingue par sa grande sensibilité aux conditions initiales et par certaines caractéristiques telles que l'absence de périodicité, le comportement en apparence aléatoire et la complexité élevée .Le système chaotique est souvent défini comme « un système qui devient apériodique (non-linéaire) lorsque son paramètre, la variable interne, les signaux externes, la variable de contrôle ou même la valeur initiale est choisi de manière spécifique ». Nous connaissons ce comportement imprévisible d'un système déterministe sous le nom de théorie du chaos ou système du chaos. [20]

1.7.2.2 Les cartes chaotiques :

En réalité, les cartes chaotiques sont des systèmes dynamiques qui sont caractérisés par des relations de récurrence. Elles servent habituellement à représenter des phénomènes

complexes et non linéaires dans différents domaines tels que la physique, les mathématiques, l'économie et l'informatique. [20]

Quelques caractéristiques seront présentées afin de mieux comprendre les points clés d'un système chaotique. [27]

- **Non-linéarité :** Un système chaotique est un système en mouvement qui n'est pas linéaire. Il est impossible qu'un système linéaire soit chaotique. Lorsque l'entrée d'un système n'est pas proportionnelle à sa sortie, ou lorsque l'événement présente des imprévisibles à long terme [28], on parle de non linéarité. La notion de système dynamique chaotique concerne tous les systèmes dont la progression est liée au temps. Les non linéarités sont responsables de l'évolution irrégulière du comportement d'un système chaotique.
- **Déterminisme** : Un système chaotique possède des règles essentielles qui sont déterministes plutôt que probabilistes.
  Le terme "déterminisme" désigne la capacité de "prédire" le futur d'un phénomène à partir d'un événement passé ou présent.
- **Sensibilité aux conditions initiales :**
  Certains phénomènes non linéaires dynamiques sont si sensibles aux conditions initiales que, même s'ils sont soumis à des lois rigoureuses et parfaitement déterministes, il est impossible de prédire avec précision. [29]

Nous citons brièvement quelques cartes chaotiques utilisées dans les applications cryptographiques.

A. Carte PWLCM :

Le système PWLCM possède une grande ergodicité et une grande sensibilité aux valeurs initiales, ce qui le rend parfaitement adapté à la cryptographie. L'équation fournit une présentation du système PWLCM.

$$x(i+1) = Fp(x_i) = \begin{cases} x_i/p & 0 \le xi \le p \\ (x_i - p)/(0{,}5 - p), & p \le xi \le 0.5 \\ F_p(1 - x_j), & 0.5 \le xi \le 1 \end{cases}$$

Si x ∈ [0,1) et le paramètre de contrôle p ∈ (0,5. 0), la carte est chaotique, les valeurs initiales de cette carte x et les paramètres de contrôle p sont utilisés comme des clés secrètes.

Le système chaotique PWLCM est utilisé à la fois dans la permutation et la diffusion. Dans la phase de confusion, nous transformons une séquence aléatoire en une séquence entière en utilisant l'équation suivante : [30]

$$a_i = \text{fix}(\text{bitsll}\ (x_i, 8))$$

- a_i **:** c'est la variable que va stocker le résultat des opération appliquées a x_i.
- **Fix :** cette fonction convertir le résultat en un nombre entier.
- **Bitsll :** est une fonction qui réalise un décalage logique à gauche (bitwise left shift) sur un nombre.
- $\mathbf{x_i}$: est le nombre sur lequel l'opération est effectuée.
- **8 :** le nombre de position de décalage vers la gauche.

L'expression fixe(n) renvoie la totalité de n, tandis que bits (n,i) renvoie la valeur totale du décalage logique à gauche de l'entrée n par i bits.

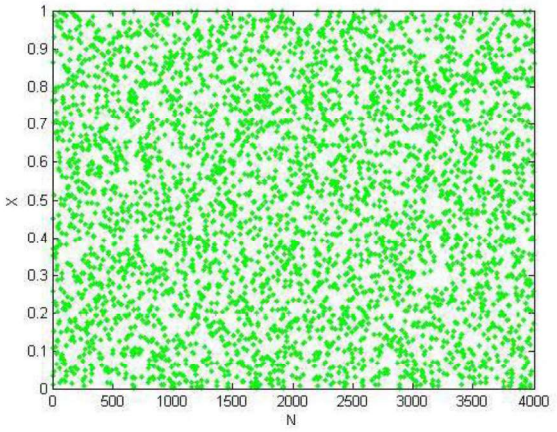


Figure 1.11 Distribution x (n) avec N = 1000 itérations [26]

B. Carte logistique :

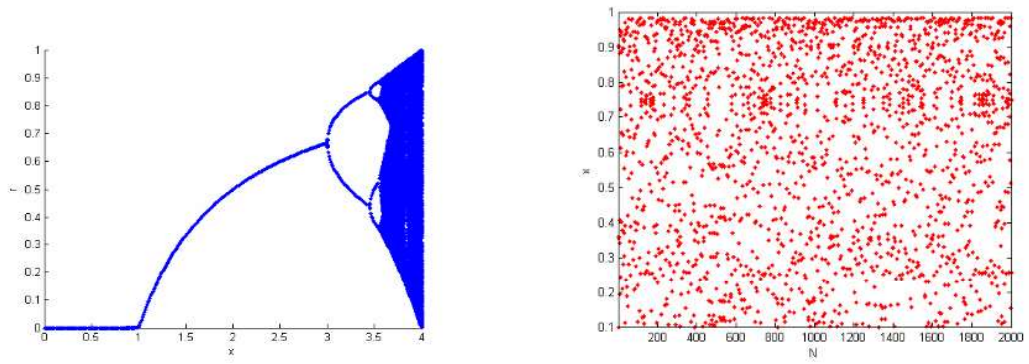
La carte logistique représente une représentation polynomiale, où le comportement de cette carte repose sur une équation dynamique non linéaire très simple.

La formule de la carte logistique chaotique est fournie par :

$$X_{(n+1)} = rX_{(n)}\ (1 - X_{(n)})$$

Dans l'intervalle [0, 1], x est une variable et n est le nombre d'itérations, tandis que r est un nombre défini dans. L'espace entre [0, 4].

On utilise la carte logistique comme une méthode chaotique pour générer la clé de la logistique. Cette clé est une paire de deux variables $(x_{(0)}, r)$ en cryptographie. Le choix de cette paire a un impact sur la conclusion du cryptage. [31]



(a) **(b)**

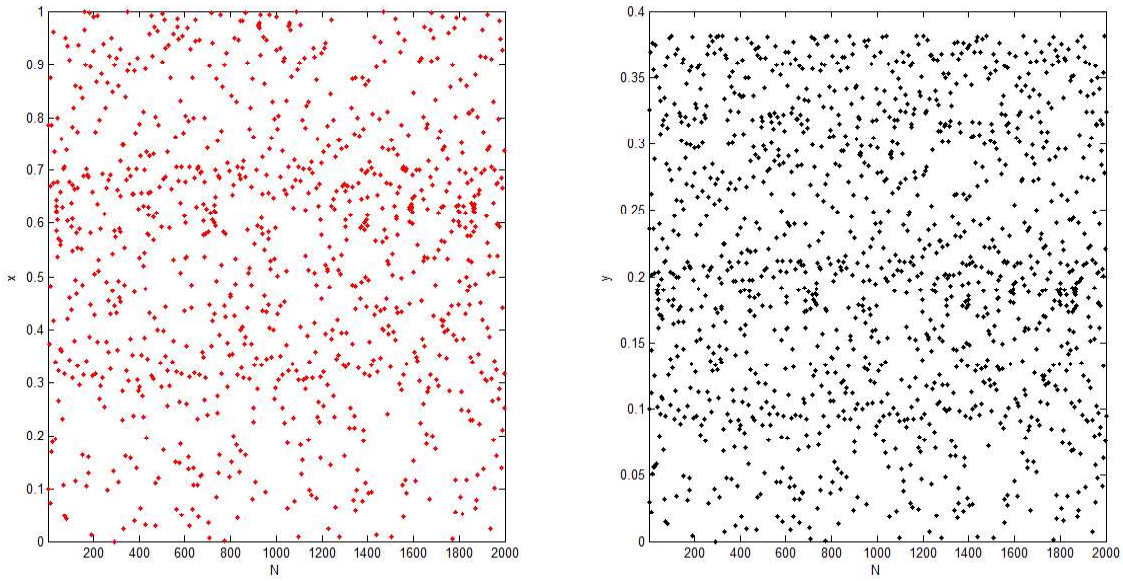
Figure 1.12 : (a) diagramme de bifurcation (b) tracée de x. [26]

C. Carte du Henon :

La récurrence de Hénon est un modèle proposé en 1976 par le mathématicien Michel Hénon, [32] le modèle d'Etat associé est :

$$\begin{cases} X_{n+1} = y_n + 1 - ax_n^2 \\ y_{n+1} = bx_n \end{cases}$$

En utilisant $(x_n, y_n) \epsilon \mathbb{R}^2$, a et b sont des paramètres. [29]



(a) (b)

Figure 1.13. (a) tracée de x**.** (b) tracé de y [26]

1.8 Conclusion :

Dans ce chapitre, nous avons présenté les fondements essentiels de la cryptographie, débutant par sa définition et explorant son évolution historique à travers les méthodes classiques et modernes de chiffrement. Nous avons également abordé les concepts révolutionnaires de la cryptographie quantique et examiné les outils mathématiques cruciaux tels que la théorie des nombres et les systèmes chaotiques, qui sous-tendent bon nombre des algorithmes cryptographiques contemporains. Ce chapitre nous a permis de comprendre comment la cryptographie garantit la sécurité et la confidentialité des informations dans un monde numérique en constante expansion, tout en anticipant les futures avancées technologiques et les défis qui y sont associés.

Chapitre 2 : Les algorithmes de chiffrement standards

2.1 Introduction :

Les algorithmes de chiffrement standard comme DES (Data Encryption Standard), 3DES (Triple DES) et IDEA (International Data Encryption Algorithm) ont été largement utilisés dans le passé pour assurer la confidentialité des données. Ils reposent sur des principes de substitution et de permutation pour transformer les données en un format illisible sans la clé appropriée. Parallèlement, les algorithmes basés sur les cartes chaotiques exploitent les propriétés du chaos déterministe pour générer des clés de chiffrement. Ces techniques offrent une alternative fascinante aux approches traditionnelles en introduisant un élément imprévisible et complexe dans le processus de chiffrement.

Ce chapitre explore les diverses techniques utilisées dans le domaine de la cryptographie pour sécuriser les informations sensibles. Nous nous concentrerons sur une variété d'algorithmes de chiffrement, depuis les standards bien établis tels que DES, 3DES, IDEA, RC4 et RC6, jusqu'aux approches innovantes basées sur les cartes chaotiques.

2.2 DES (Data inscription standard)

Le DES (Data Encryption Standard) est un algorithme de chiffrement à clé symétrique largement utilisé qui a joué un rôle majeur dans l'histoire de la cryptographie. Au début des années 1970, le développement des communications entre ordinateurs a nécessité la création d'un standard de cryptage des données pour permettre l'interopérabilité entre différents systèmes. Pour répondre à ce besoin, la National Security Agency (NSA) américaine a lancé un appel d'offres. IBM a développé un algorithme nommé Lucifer, complexe et sophistiqué. Après plusieurs années de discussions et de modifications, incluant l'ajout des S-Box et la réduction de la clé à 56 bits, cet algorithme est devenu le DES. [34] Il opère sur des blocs de données en clair de 64 bits et utilise une clé pour chiffrer ces blocs en texte chiffré de 64 bits. Initialement dotée de 64 bits, la taille de la clé du DES est réduite à 56 bits en retirant chaque huitième bit avant le chiffrement. Le processus de chiffrement implique plusieurs opérations, incluant des substitutions, des transpositions et des mélanges de clés, utilisant une structure de chiffrement Feistel répartie en 16 tours. En raison de la taille relativement petite de sa clé, le DES a été considéré comme vulnérable aux attaques par force brute Il a été adopté comme standard fédéral le 23 novembre 1976. Aujourd'hui, il a largement été remplacé par des algorithmes de chiffrement plus sécurisés offrant des clés plus longues et une meilleure protection contre les attaques. [33]

2.2.1 Principe de fonctionnement

Les données brutes sont divisées en blocs de 64 bits par le DES, puis chiffrées à l'aide d'une clé pour obtenir des blocs de texte chiffré de 64 bits. La clé initiale utilisée dans le DES est de 64 bits, mais 8 bits sont supprimés avant le chiffrement, réduisant ainsi la clé à 56 bits. Ensuite, cette clé de 56 bits est subdivisée en sous-clés de 48 bits, qui sont utilisées dans les opérations de substitution, de transposition et de mélange des clés constituant la structure de chiffrement de Feistel [34].

Le chiffrement Feistel utilise 16 tours, chaque tour employant une sous-clé différente générée à partir de la clé de 56 bits par l'algorithme de génération de clé. Toutefois, dans le DES, la même sous-clé est utilisée pour les tours 2 à 16.

Les opérations de substitution, de transposition et de mélange des clés sont intégrées dans le processus de chiffrement, utilisant la structure de chiffrement de Feistel.

2.2.2 Génération des clés :

L'algorithme ci-dessous montre comment obtenir à partir d'une clé de 64 bits (composé de 64 caractères alphanumériques quelconques) 8 clés diversifiées de 48 bits chacune servant dans l'algorithme du DES. [48]

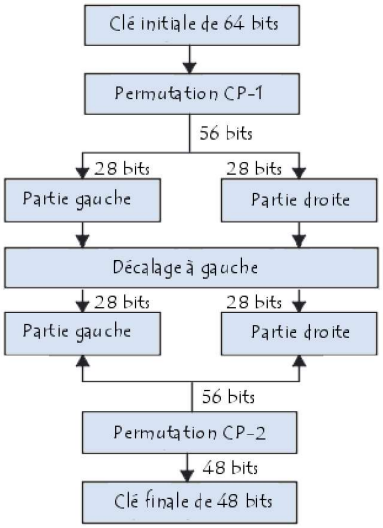


Figure 2.1 Génération des clés [48]

-Dans un premier temps les bits de parité de la clé sont éliminés afin d'obtenir une clé d'une longueur utile de 56 bits.

-La première étape consiste en une permutation notée CP-1 dont la matrice est présentée ci-dessous :

CP-1	57	49	41	33	25	17	9	1	58	50	42	34	26	18
	10	2	59	51	43	35	27	19	11	3	60	52	44	36
	63	55	47	39	31	23	15	7	62	54	46	38	30	22
	14	6	61	53	45	37	29	21	13	5	28	20	12	4

-Cette matrice peut en fait s'écrire sous la forme de deux matrice Gi et Di (pour gauche et droite) composées chacune de 28 bits :

G_i	57	49	41	33	25	17	9
	1	58	50	42	34	26	18
	10	2	59	51	43	35	27
	19	11	3	60	52	44	36

D_i	63	55	47	39	31	23	15
	7	62	54	46	38	30	22
	14	6	61	53	45	37	29
	21	13	5	28	20	12	4

-On note G0 et D0 le résultat de cette première permutation. [48]

-Ces deux blocs subissent ensuite une rotation à gauche, de telles façons que les bits en seconde position prennent la première position, ceux en troisième position la seconde, ...

-Les bits en première position passent en dernière position.

-Les 2 blocs de 28 bits sont ensuite regroupés en un bloc de 56 bits. Celui-ci passe par une permutation, notée CP-2, fournissant en sortie un bloc de 48 bits, représentant la clé Ki.

CP-2	14	17	11	24	1	5	3	28	15	6	21	10
	23	19	12	4	26	8	16	7	27	20	13	2
	41	52	31	37	47	55	30	40	51	45	33	48
	44	49	39	56	34	53	46	42	50	36	29	32

-Des itérations de l'algorithme permettent de donner les 16 clés K1 à K16 utilisées dans l'algorithme du DES. [48]

LS	1	2	4	6	8	10	12	14	15	17	19	21	23	25	27	28

2.2.3 Chiffrement :

Le DES (Data Encryption Standard) est un système de chiffrement basé sur les blocs. Cela signifie que DES ne chiffre pas les données dès qu'elles arrivent, mais divise virtuellement le texte en clair en blocs de 64 bits qu'il chiffre individuellement, puis concatène les blocs chiffrés. Un bloc de texte en clair de 64 bits entre dans l'algorithme et un bloc de texte chiffré en ressort. Le principe de l'algorithme est relativement simple, combinant uniquement des permutations et des substitutions. [48]

Il s'agit d'un algorithme de chiffrement symétrique, ce qui signifie que la même clé est utilisée pour chiffrer et déchiffrer le message. La longueur initiale de la clé est de 64 bits, soit 8 caractères, mais seuls 56 bits sont utilisés pour le chiffrement, les 8 bits restants servant de bits de contrôle de parité.

La figure suivante illustre l'algorithme de chiffrement de DES.

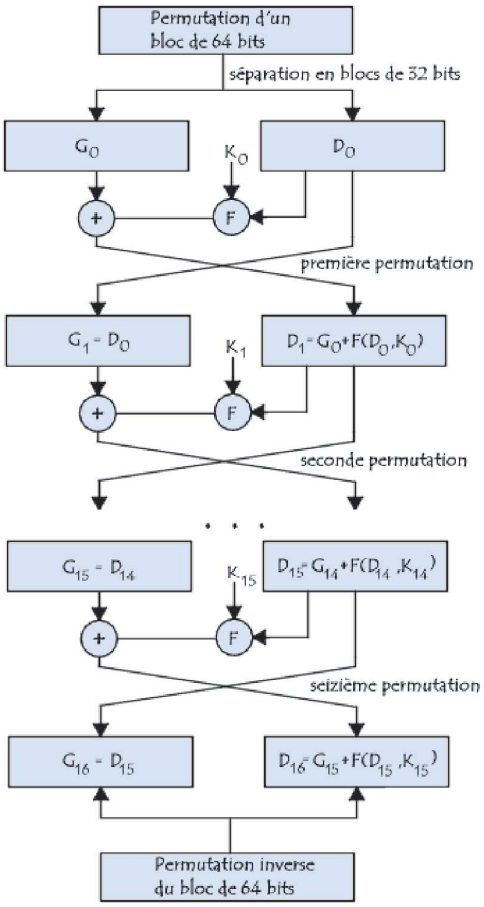


Figure 2.2 Algorithme de chiffrement [48]

Les grandes lignes de l'algorithme sont les suivantes :

- Fractionnement du texte en blocs de 64 bits (8 octets) ;
- Permutation initiale des blocs ;
- Découpage des blocs en deux parties : gauche et droite, nommées G et D ;
- Etapes de permutation et de substitution répétées 16 fois (appelées rondes) ;
- Recollement des parties gauche et droite puis permutation initiale inverse.

2.2.3.1 Fractionnement du texte :

Le processus de fractionnement du texte en blocs de 64 bits (8 octets) consiste à diviser le texte clair en segments de longueur fixe afin de les chiffrer individuellement, facilitant ainsi le traitement par l'algorithme DES. [48]

2.2.3.2 Permutation initiale :

Dans un premier temps, chaque bit d'un bloc est soumis à la permutation initiale, pouvant être représentée par la matrice de permutation initiale (notée PI) suivante :

PI							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Cette matrice de permutation indique, en parcourant la matrice de gauche à droite puis de haut en bas, que le 58ème bit du bloc de texte de 64 bits se retrouve en première position, le 50ème en seconde position et ainsi de suite. [48]

2.2.3.3 Scindement en blocs de 32 bits :

Une fois la permutation initiale réalisée, le bloc de 64 bits est scindé en deux blocs de 32 bits, notés respectivement G et D (pour gauche et droite, la notation anglo-saxonne étant L et R pour Left and Right). On note G0 et D0 l'état initial de ces deux blocs :

G_0							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8

D_0							
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Il est intéressant de remarquer que G0 contient tous les bits possédant une position paire dans le message initial, tandis que D0 contient les bits de position impaire [48]

2.2.3.4 Rondes :

Les blocs Gn et Dn sont soumis à un ensemble de transformation itératives appelées rondes, explicitées dans ce schéma, et dont les détails sont donnés plus bas : [48]

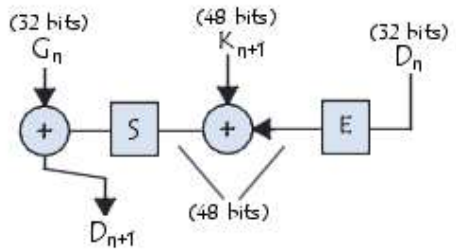


Figure 2.3 Fonction F [48]

- **Fonction d'expansion :** Les 32 bits du bloc D0 sont étendus à 48 bits grâce à une table (matrice) appelé table d'expansion (notée E), dans laquelle les 48 bits sont mélangés et 16 d'entre eux sont dupliqués : [48]

E						
	32	1	2	3	4	5
	4	5	6	7	8	9
	8	9	10	11	12	13
	12	13	14	15	16	17
	16	17	18	19	20	21
	20	21	22	23	24	25
	24	25	26	27	28	29
	28	29	30	31	32	1

Ainsi, le dernier bit de D0 (c'est-à-dire le 7ème bit du bloc d'origine) devient le premier, le premier devient le second, ...

De plus, les bits 1,4,5,8,9,12,13,16,17,20,21,24,25,28 et 29 de D0 (respectivement 57, 33, 25, l, 59, 35, 27, 3, 6l, 37, 29, 5, 63, 39, 31 et 7 du bloc d'origine) sont dupliqués et disséminés dans la matrice. [48]

- **OU exclusif avec la clé :** La matrice résultante de 48 bits est appelée D'0 ou bien E[D0]. L'algorithme DES procède ensuite à un OU exclusif entre la première clé K1 et E[D0]. Le résultat de ce OU exclusif est une matrice de 48 bits que nous appellerons D0 par commodité (il ne s'agit pas du D0 de départ).

- **Fonction de substitution S :** D0 est ensuite scindé en 8 blocs de 6 bits, noté D0i. Chacun de ces blocs passe par des fonctions de sélection (appelées parfois boîtes de substitution ou fonctions de compression), notées généralement Si.

Les premiers et derniers bits de chaque D0i déterminent (en binaire) la ligne de la fonction de sélection, les autres bits (respectivement 2, 3, 4 et 5) déterminent la colonne. La sélection de la ligne se faisant sur deux bits, il y a 4 possibilités (0,1,2,3). La sélection de la colonne se faisant sur 4 bits, il y a 16 possibilités (0 à 15). Grâce à cette information, la fonction de sélection "sélectionne" une valeur codée sur 4 bits.

Voici la première fonction de substitution, représentée par une matrice de 4 par 16 :

S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Soit D01 égal à 101110. Les premiers et derniers bits donnent 10, c'est-à-dire 2 en binaire. Les bits 2,3,4 et 5 donnent 0111, soit 7 en binaire. Le résultat de la fonction de sélection est donc la valeur située à la ligne n°2, dans la colonne n°7. Il s'agit de la valeur 11, soit en binaire 111.

Chacun des 8 blocs de 6 bits est passé dans la fonction de sélection correspondante, ce qui donne en sortie 8 valeurs de 4 bits chacune.

Les sept autres matrices de substitution (S2 à S8) traitent également 6 bits chacune, couvrant ainsi les 48 bits d'entrée. Le résultat total des substitutions est une sortie combinée de 4 bits par S-box, ce qui donne 32 bits au total (4 bits * 8 S-boxes = 32 bits). [48]

- **Permutation :** Le bloc de 32 bits obtenu est enfin soumis à une permutation P dont voici la table :

P								
	16	7	20	21	29	12	28	17
	1	15	23	26	5	18	31	10
	2	8	24	14	32	27	3	9
	19	13	30	6	22	11	4	25

L'ensemble de ces résultats en sortie de P est soumis à un OU Exclusif avec le G0 de départ (comme indiqué sur le premier schéma) pour donner D1, tandis que le D0 initial donne G1.

- **Itération :** L'ensemble des étapes précédentes (rondes) est réitéré 16 fois.

2.2.3.5 Permutation initiale inverse :

A la fin des itérations, les deux blocs G16 et D16 sont "recollés, puis soumis à la permutation initiale inverse :

PI-1								
	40	8	48	16	56	24	64	32
	39	7	47	15	55	23	63	31
	38	6	46	14	54	22	62	30
	37	5	45	13	53	21	61	29
	36	4	44	12	52	20	60	28
	35	3	43	11	51	19	59	27
	34	2	42	10	50	18	58	26
	33	1	41	9	49	17	57	25

Le résultat en sortie est un texte codé de 64 bits.

2.2.4 Déchiffrement :

Le déchiffrement du DES est réalisé en appliquant l'algorithme inverse de chiffrement, utilisant les mêmes sous-clés mais dans l'ordre inverse. [48]

2.2.5 Sécurité du DES :

Le Data Encryption Standard (DES) a été un pilier de la cryptographie symétrique depuis son adoption en 1977. Cependant, malgré sa simplicité et son efficacité initiale, le DES a suscité des préoccupations croissantes concernant sa sécurité au fil du temps [49].

Le principal point faible du DES réside dans la taille de sa clé de 56 bits. À l'époque de sa création, cette longueur de clé était considérée comme suffisamment robuste. Toutefois, avec les avancées technologiques et l'augmentation de la puissance de calcul, il est devenu de plus en plus vulnérable aux attaques par force brute. En effet, il est désormais possible de tester toutes les combinaisons possibles de clés en un temps raisonnable, rendant le DES insuffisant pour protéger les informations sensibles.

En outre, certaines faiblesses structurelles du DES, telles que les S-boxes et les permutations, ont été examinées et exploitées dans divers types d'attaques cryptanalytiques, comme les attaques linéaires et différentielles. Ces méthodes exploitent les schémas de redondance et de prévisibilité au sein de l'algorithme pour réduire le nombre de tentatives nécessaires pour trouver la clé correcte.

Pour pallier ces vulnérabilités, le Triple DES (3DES) a été introduit. Ce mécanisme applique le DES trois fois de suite avec deux ou trois clés différentes, augmentant ainsi la taille effective de la clé et améliorant la sécurité. Cependant, 3DES est également plus lent et a ses propres limitations.

Aujourd'hui, le DES est largement remplacé par des algorithmes plus sécurisés, tels que l'AES (Advanced Encryption Standard), qui offre des longueurs de clé de 128, 192 et 256 bits, assurant une protection bien plus robuste contre les attaques modernes. [49]

2.3 TDES (Triple DES) :

La cryptanalyse différentielle a été développée par Eli Biham et Adi Shamir en 1990. Elle permet de décrypter des versions limitées du DES, où le nombre de tours de la boucle principale est réduit. De cette façon, un DES à 8 ou 10 tours pouvait être facilement brisé. Par ailleurs, un ordinateur connu sous le nom de « DES cracker », qui comprend 1536 puces et a la capacité de chercher 88 milliards de clés par seconde, a réussi à rivaliser avec les laboratoires RSA en craquant une clé DES en 56 heures. Par conséquent, l'algorithme de TDES a été développé afin d'améliorer la sécurité du DES [35].

Le Triple DES, également appelé TDES ou 3-DES, est un algorithme de chiffrement symétrique par bloc. Comme son nom l'indique, chaque bloc de données est chiffré trois fois en utilisant trois clés différentes de 56 bits chacune, soit une clé totale de 168 bits (112 bits effectifs de sécurité) , comme illustré dans la figure 2.4.

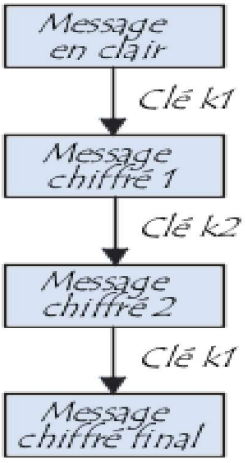


Figure 2.4 : Algorithme du chiffrement Triple DES. [48]

Le TDES implique l'utilisation de plusieurs algorithmes de chiffrement et de déchiffrement, et chaque application DES utilise une clé différente de 56 bits. [35]

2.4 RC4 (Rivest Cipher 4) :

Ron Rivest de RSA Security a développé un algorithme de chiffrement de flux et de clé de longueur variable en 1987. Cet algorithme utilise un octet simultanément (ou des unités plus grandes simultanément). Un générateur de bits pseudo-aléatoire est un dispositif qui génère un flux de 8 bits inattendu sans avoir connaissance de la clé d'entrée. La sortie du générateur est connue sous le nom de flux de clé, et elle est chiffrée à la fois avec un octet et le flux de texte en clair en utilisant l'opération X-OR. [36]

2.4.1 Description générale

RC4 est un algorithme de chiffrement à flot synchrone prenant en entrée une clé secrète pouvant varier de 40 à 1024 bits. En pratique, on choisit souvent une taille de clé de 128 bits. En revanche, cet algorithme ne prend pas de vecteur d'initialisation en entrée [36].

RC4 fonctionne de la façon suivante : la clef RC4 permet d'initialiser un tableau de 256 octets en répétant la clef autant de fois que nécessaire pour remplir le tableau. Par la suite, des opérations très simples sont effectuées : les octets sont déplacés dans le tableau, des additions sont effectuées, etc. Le but est de mélanger autant que possible le tableau. Finalement on obtient

une suite de bits pseudo-aléatoires qui peuvent être utilisés pour chiffrer les données via un XOR.

L'état interne se compose de 258 octets répartis de la manière suivante :

- Une permutation S de toutes les 256 valeurs possibles d'un octet,
- Deux pointeurs i et j servant d'index dans le tableau de permutation.

Après la phase d'initialisation de la clé, l'algorithme généré un octet de suite chiffrant par itération de la mise à jour d'état interne. Les fonctions d'initialisation et de mise à jour d'état interne sont décrites dans les paragraphes suivants.

L'algorithme RC4 se compose de :

- Un algorithme génération des clés **;**
- Un algorithme de chiffrement ;
- Un algorithme de déchiffrement.

Les deux algorithmes (chiffrement et déchiffrement) sont les même néanmoins la seule différence en remplace le message clair par le message crypté. [36]

2.4.2 Génération des clés :

2.4.2.1 Initialisation de la matrice S

Avant de générer les premiers octets de suite chiffrant, l'algorithme exécute une phase d'initialisation.

Cette étape consiste à utiliser la clé secrète comme une graine pour la génération d'une suite chiffrant aléatoire. Dans un premier temps, tous les octets de la permutation S sont initialisés avec leurs indices respectifs. Ainsi,

S [0] = 0, S [1] = 1,..., S [255] = 255.

Algorithme 1

- Pour i = 0 à 255 faire
- S[i] ← i
- Fin Pour

2.4.2.2 Génération de la suite chiffrant

Une fois ces opérations effectuées, l'algorithme utilise la clé secrète pour calculer des indices qui serviront à mélanger la permutation S de manière « aléatoire ». Le principe de l'initialisation est décrit dans l'algorithme 2. De cette manière, l'ordre du mélange réalisé dans de la permutation S dépend de la clé et l'on est sûr que chacun des octets de S a bien été permuté au moins une fois. Un fois cette initialisation terminée, l'algorithme est prêt pour la génération de la suite chiffrant proprement dite.

L'algorithme RC4 généré un octet de suite chiffrant par itération de la fonction de mise à jour de l'état interne. Contrairement à la phase d'initialisation. [36]

Algorithme 2

Entrées : Etat interne S, clé secrète K, taille de clé n

- j ← 0
- Pour i = 0 ` a 255 faire
- j ← (j + S[i] + K[i mod n]) mod 256
- Temp ← S[i]
- S[i] ← S[j]
- S[j] ← Temp
- Fin Pour

2.4.3 Chiffrement

La clé n'est plus utilisée pour mettre à jour les registres d'état interne. Au début de l'exécution, les pointeurs de l'état interne. i et j sont remis à zéro, puis leur valeur est incrémentée de la façon suivante :

- i ← i + 1 mod 256
- j ← j + S[i] mod 256

Ces pointeurs servent ensuite à réaliser la mise à jour de l'état de la permutation S. En effet, ils définissent les octets de S qui seront additionnés modulo 256 puis permutés. Le résultat de cette addition définit l'octet de la permutation S qui sera retourné en guise d'octet de la suite chiffrant.

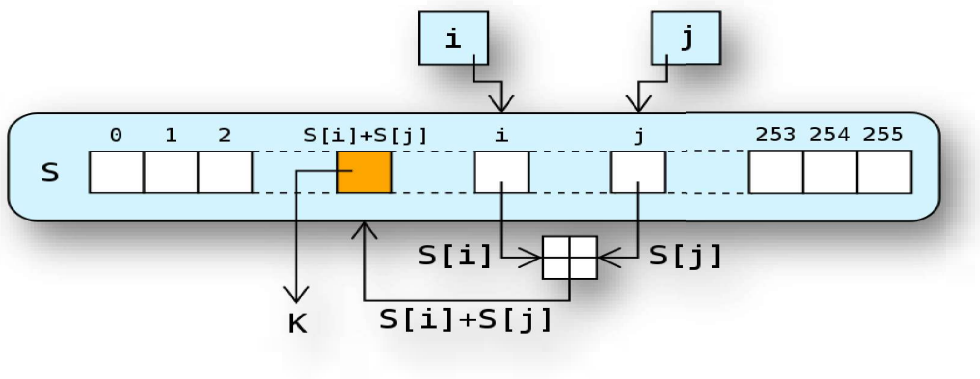


Figure 2.5 Schéma de mise à jour de l'état interne de RC4 [36]

Ainsi, pour chaque itération de la mise à jour de l'état interne, un octet de suite chiffrant est généré. Cet octet est ensuite combiné avec un octet de message via un « ou exclusif » pour donner un octet du chiffré. [36]

Algorithme de chiffrement :

- i := 0
- j := 0
- tant que générer une sortie:
- i := (i + 1) mod 256
- j := (j + S[i]) mod 256
- échanger(S[i], S[j])
- octet_chiffrement = S[(S[i] + S[j]) mod 256]
- result_chiffré = octet_chiffrement XOR octet_message
- fin tant que

2.4.4 Déchiffrement

RC4 est un générateur de bits pseudo-aléatoires dont le résultat est combiné avec le texte en clair via une opération XOR, alors on peut trouver le déchiffrement avec même algorithme de chiffrement, on remplace le message claire dans l'algorithme par le message chiffré, et on doit trouver le message clair. [36]

Algorithme de déchiffrement :

- i := 0
- j := 0
- tant que générer une sortie:
- i := (i + 1) mod 256
- j := (j + S[i]) mod 256
- échanger(S[i], S[j])
- octet_déchiffrement = S[(S[i] + S[j]) mod 256]
- result_déchiffré = octet_déchiffrementXORoctet_message chiffrer
- fin tant que

2.4.5 Sécurité de RC4 :

Le RC4 a montré des vulnérabilités significatives au fil du temps, notamment des biais dans le flux de clé et des patterns prévisibles dans le flux de sortie. Ces faiblesses ont permis des attaques statistiques qui ont compromis sa sécurité. Les premiers octets du flux de clé étaient particulièrement vulnérables, rendant possible la récupération de la clé secrète ou du texte en clair par des adversaires bien informés. En conséquence, RC4 a été progressivement abandonné dans de nombreux protocoles cryptographiques modernes au profit d'algorithmes plus robustes comme AES, qui offrent une sécurité améliorée et une résilience renforcée contre les attaques. [36]

2.5 RC6 (Rivest cipher 6) :

RC6 est un nouveau chiffre par bloc a été créé en 1998, soumis à NIST devenir le nouveau standard de la cryptographie avancée (Advanced Encryptions Standard - AES). La conception de RC6 a commencé par une considération de RC5 comme un candidat potentiel à une soumission AES. Les modifications ont été alors faites pour satisfaire aux exigences AES, augmenter la sécurité et améliorer la performance. [36]

2.5.1 Description générale :

RC6 est un algorithme de chiffrement par blocs avec de nombreux paramètres : la taille des mots traités w-bits, le nombre de rondes r et la taille de la clef (b octets).

La version proposée pour AES utilise par exemple les paramètres w = 32, r = 20 et b = 16, 24 ou 32. [39]

L'algorithme RC6 utilise six opérations pour chiffrer et déchiffré les unités de quatre mots w bits à savoir :

- « a+b» : l'addition modulo 2^w.
- « a-b » : la soustraction modulo 2^w.
- « $\oplus$ » : OU exclusif entre les mots de w-bits.
- « a*b » : multiplication modulo 2^w.
- « a<<<b »: tourner le mot (a) de taille w-bits à la gauche par la quantité donnée par bits du mot (b).
- « a>>>b »: tourner le mot (a) de taille w-bits à la droit par la quantité donnée par bits du mot (b).

L'utilisateur fournit une clé de longueur k octets et le bloc de texte en clair de 128 bits est chargé dans des mots A; B; C; et D commençant avec l'octet d'ordre bas A.

Ces quatre mots de taille w-bits contiennent le cryptogramme de sortie à la fin. [36]

L'algorithme RC6 se compose de :

- Un algorithme de génération des clés ;
- Un algorithme de chiffrement ;
- Un algorithme de déchiffrement.

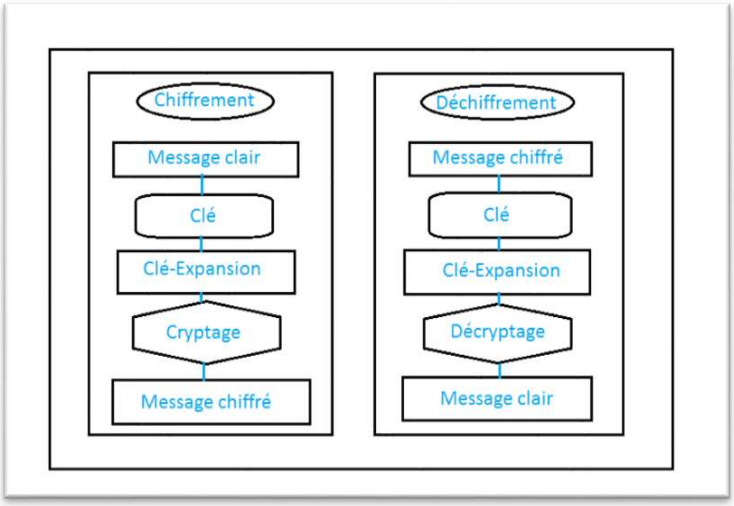


Figure 2.6 Chiffrement et le déchiffrement d'un message avec l'algorithme RC6. [36]

2.5.2 Génération des clés :

Le clé-expansion de RC6-w / r / b est pratiquement identique au clé-expansion RC5-w / r / b. En effet, la seule différence est que, pour RC6-w / r / b, plusieurs mots sont dérivés de la clé fournie par l'utilisateur destiné à être utilisé lors du chiffrement et du déchiffrement. [36]

2.5.2.1 La conversion de la clé secrète k des octets de mots L

La première étape de l'expansion de clé algorithmique est de copier la clé secrète K[0...b-1] dans une matrice L[0...c-1] de c=[b/u] mots, où U=w/8 est le nombre d'octets par mot. Cette opération s'effectue de façon naturelle, à l'aide de u touches consécutives octets de K à remplir chaque mot dans l'octet d'ordre faible d'octet d'ordre haut. [37]

Les positions de l'octet en carnet sont remises à zéro. Dans le cas que b=C=0 nous reset c à 1 et définissez L[0] à zéro

Notant que :

- K : la clé secret
- b : la taille de la clé
- c : la taille de la matrice L
- u : le nombre d'octets par mot

Algorithme 1 :

- for i=b-1 downto 0 do
- L [i/u] = (L [i/u] <<<8) + K [i];

2.5.2.2 L'initialisation de la matrice S :

La deuxième étape de l'expansion algorithmique clé est d'initialiser matrice S à un particulier fixe (clé indépendant) pseudo-aléatoire particulier motif, en utilisant une progression arithmétique modulo 2^w déterminée par la « magie constantes » P_w et Q_w, Depuis Q_w est impair, la progression arithmétique a période 2^w. [36]

Rappel sur les constantes magiques :

On définit deux constants binaires P_w et Q_w qui représentent les parties décimales de deux nombres : e (2.7182….) et le nombre d'or (1.61803….) sur w bits (w=16, 32 ou 64 bits).Ces valeurs arbitraires peuvent être choisies par le concepteur d'un système proposant RC6. Il faut

toutefois que l'auteur d'un message et le destinataire utilisent les mêmes coefficients P_w et Q_w. [36]

Par exemple :

- P_{16} = 1011011111100001 = B7E1
- Q_{16} =1001111000110111 = 9E37
- P_{32} = B7E15163
- Q_{32} =9E3779B9

On définit ensuite des tableaux S qu'on remplit de la façon suivante :

- S [0]= P_w
- for i= 1 to t-1 do
- S [i] = S [i-1] + Q_w ;

2.5.2.3 Le mixage dans la clé secrète :

La troisième étape de d'extension clé algorithmique est de mélanger dans la clé secrète de l'utilisateur dans trois passes au-dessus des baies S et L (mélangez L et S), plus précisément, en raison de l'effet potentiellement différentes tailles de S et L, la gamme plus large sera traité trois fois, et l'autre peuvent être traités plusieurs fois. [36]

- I=j=0 ;
- A=B=0 ;
- Do 3*max (t,c) times :
- A=S [i] = (S[i] +A +B) <<< 3;
- B=L [j] = (L[j] +A +B) << (A +B);
- i= (i+1) mod (t);
- j= (j+1) mod (c);

2.5.3 Chiffrement :

RC6 fonctionne avec quatre registres de w-bits A, B, C et D qui contiennent de l'entrée initiale de texte en clair, ainsi que le texte chiffré de sortie à la fin de chiffrement. Le registre B est initialisé avec S [0] et le registre D est initialisé avec S [1].

Après l'initialisation t = B x (2B + 1) et u = D x (2D + 1) valeurs se trouvent et tourné à gauche par $\log_2$ w. Les registres A et t sont XORés et tournés vers la gauche par la valeur de u et additionnées avec les mêmes entrées de S [i] et cette valeur est affectée à A. Plus tardLes registres C et u sont XORés et tournés vers la gauche par la valeur de t et additionnées par les entrées impaires de S [i]. Cette valeur est affectée à C. Le premier octet de texte en clair ou de texte chiffré est placé dans l'octet le moins significatif de A, Le dernier octet de texte en clair ou de texte chiffré est placé dans l'octet de poids fort de D. [40]

(A, B, C, D) = (B, C, D, A) est utilisé pour signifier l'affectation parallèle de valeurs sur le droit de registres sur la gauche. Et cette routine est répétée pour les rondes r puis, enfin, S [2r + 2] et S [2r + 3] sont ajoutés à A et C, respectivement. (Voir figure 2.7) [36]

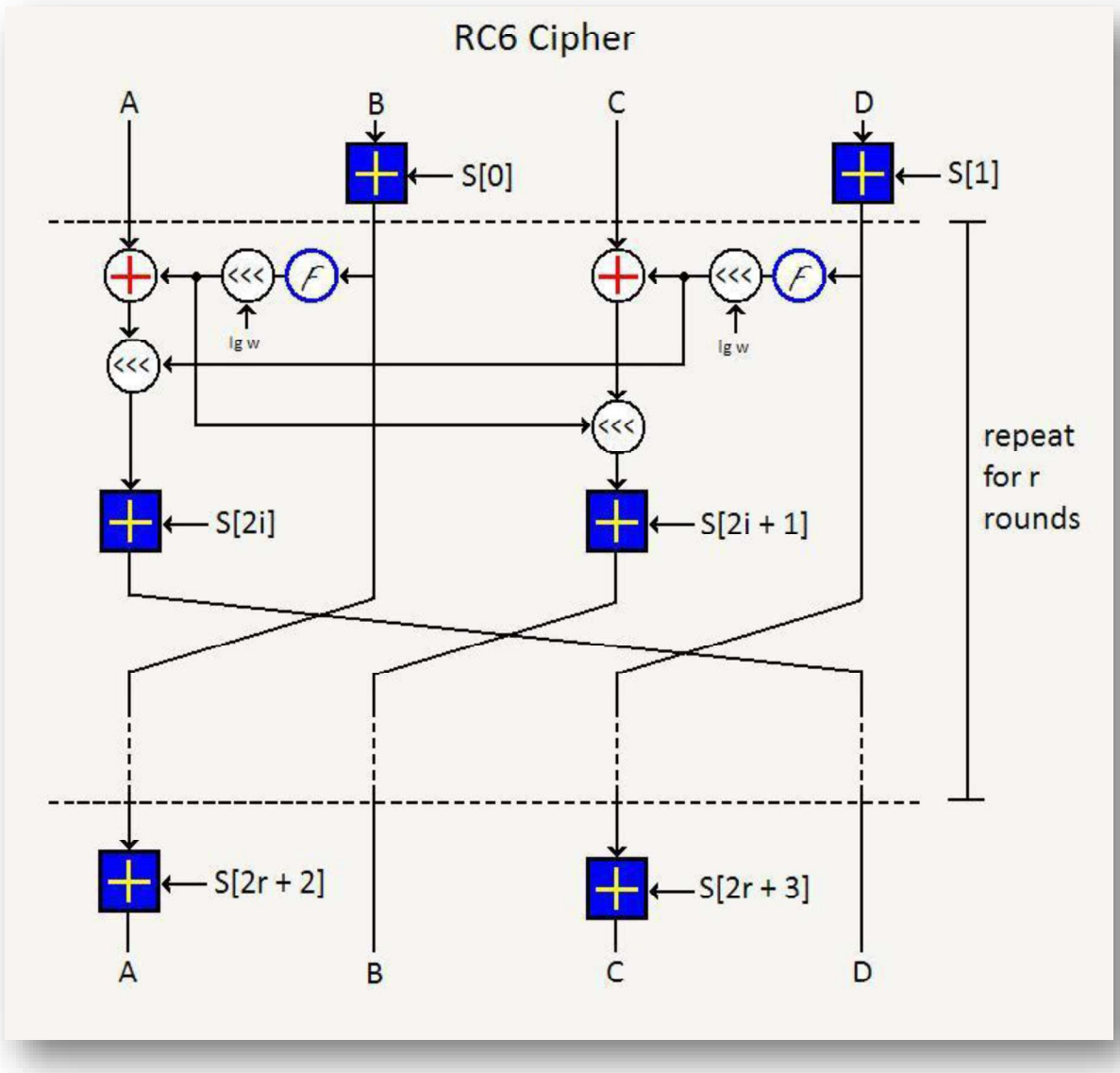


Figure 2.7 : Une Tour d'Algorithme de Chiffrement RC6 [36]

A, B, C, D les données d'entrées sur w bits chacune, en sortie, on aura A, B, C et D.

Le code de cette procédure est le suivant :

- B = B + S [0]
- D = D + S [1]
- pour i = 1 à r faire
- {t = (B x (2B + 1)) <<< $\log_2$ w ……. => ligne 4
- u = (D x (2D + 1)) <<< $\log_2$ w …….=> ligne 5

- A = ((A ⊕ t) <<< u) + S [2i]
- C = ((C ⊕ u) <<< t) + S [2i + 1]
- (A, B, C, D) = (B, C, D, A)}
- A = A + S [2r + 2]
- C = C + S [2r + 3]

Remarque : La version de RC6 proposée pour AES utilise 5 au lieu de lg_2 w (voir les lignes 4 et 5 de procédure).

2.5.4 Déchiffrement :

Après avor effectué le chiffrement en utilisant l'algorithme de cryptage, en doivent restaurer le message clair à partir de message chiffré, Pour cette raison, le processus de déchiffrement utilisant un l'algorithme de décryptage [36].

L'algorithme de déchiffrement est un inverse ou à l'opposé de l'algorithme de chiffrement.

Les opérateurs existants dans l'algorithme de décodage est l'inverse de l'opérateur dans l'algorithme de chiffrement. Les blocs de données sont traités le même que le bloc de données dans le processus de chiffrement, qui est de 128 bits divisés en quatre registres A, B, C et D.

A, B, C, D les données d'entrées sur w bits chacune, en sortie, on aura A, B, C et D. [37]

Le code de cette procédure est le suivant :

- C = C – S [2r + 3]
- A = A – S [2r + 2]
- for i = r downto 1 do
- {(A, B, C, D) = (D, A, B, C)
- u = (D x (2D + 1)) <<< $\log_2$ w
- t = (B x (2B + 1)) <<< $\log_2$ w
- C = ((C – S [2i + 1]) >>> t) ⊕ u
- A = ((A – S [2i]) >>> u) ⊕ t}
- D = D – S [1]
- B = B – S [0]

2.6 Étude comparative entre les algorithmes de chiffrements (DES, 3DES, RC6, RC4) :

Le DES (Data Encryption Standard) est le plus ancien parmi ces algorithmes et utilise une clé de 56 bits, ce qui le rend vulnérable aux attaques par force brute. Pour renforcer la sécurité, le 3DES a été développé en utilisant trois applications successives de DES avec des clés distinctes, mais il demeure relativement lent et accroît la complexité des calculs. En revanche, le RC6 offre à la fois des performances élevées et une sécurité robuste, avec la possibilité d'utiliser des clés allant jusqu'à 256 bits, ce qui en fait un choix moderne et efficace pour le chiffrement. Quant au RC4, c'est un algorithme de flux rapide et facile à implémenter, mais il est désormais considéré comme peu sûr en raison de vulnérabilités connues. En tenant compte de ces critères, le RC6 présente un bon compromis entre sécurité et performances, tandis que le DES et le 3DES ont été largement remplacés par des algorithmes plus récents. Il n'est plus recommandé d'utiliser le RC4 en raison de ses failles de sécurité connues. Lors du choix d'un algorithme de chiffrement, il est essentiel de prendre en compte les exigences spécifiques en matière de sécurité, de performances et de compatibilité avec les systèmes existants. Ce tableau suivant résume les principales caractéristiques de chacun de ces algorithmes de chiffrement symétrique

Algorithme	Taille de la clé	Taille du bloc de message	Nombre de rounds	Sécurité
RC4	40 à 2048 bits	Variable	N/A	Non recommandé pour la sécurité moderne en raison de vulnérabilités connues.
RC6	Jusqu'à 2048 bits	32, 64 ou 128 bits	20	Offre une bonne sécurité avec une flexibilité de taille de bloc et de clé. Moins utilisé que AES.
DES	56 bits (effectif après la suppression de la parité)	64 bits	16	Obsolète et vulnérable aux attaques par force brute en raison de la petite taille de la clé. Ne doit pas être utilisé pour des applications nécessitant une sécurité forte.
3DES	168 bits (trois clés de 56 bits)	64 bits	48 (3 fois 16)	Plus sécurisé que DES en raison de l'utilisation de trois clés, mais plus lent et moins efficace que les algorithmes modernes comme AES. Utilisé principalement dans les systèmes hérités nécessitant une compatibilité avec DES.

Tableau 2.1 Comparaison générale

2.7 Les algorithmes de chiffrement récent basé sur les cartes chaotiques :

-En 2020, Ali Cherif et ses collaborateurs ont proposé une nouvelle méthode pour le chiffrement d'images en utilisant la carte chaotique linéaire par morceaux (PWLCM). Ils ont transformé l'image en un texte clair abstrait et géométrique sous forme de cylindre, créant ainsi une série de cercles superposés. PWLCM a été utilisé comme générateur pseudo-aléatoire pour perturber les angles générés par cette transformation. Les chercheurs ont appliqué des fonctions chaotiques pour transformer l'angle résultant (Alpha) en une autre valeur d'angle aléatoire (BETA). Ensuite, ils ont utilisé les propriétés trigonométriques du cercle pour calculer le sinus et le cosinus de l'angle transformé BETA. Les valeurs obtenues de la première séquence chaotique (indices triés par sinus) ont été utilisées pour déterminer les nouvelles positions des lignes, tandis que les valeurs de la deuxième séquence chaotique (indices triés par cosinus) ont été utilisées pour déterminer les nouvelles positions des colonnes. [37]

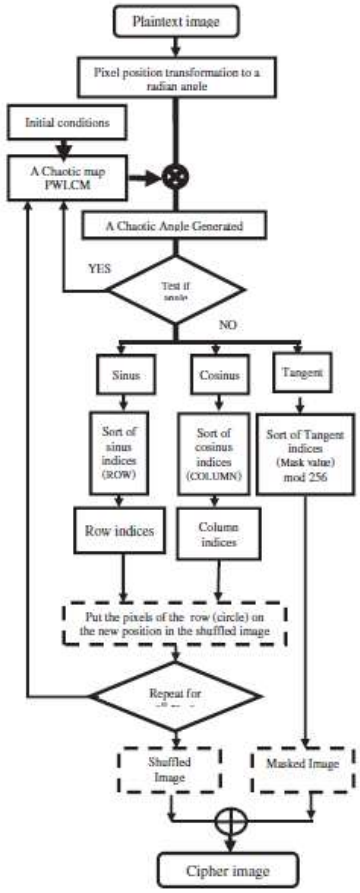


Figure 2.8 Algorithme proposé par Ali Cherif [43]

-En 2019, Seyed Shahab Eddine et al. Ont proposé un nouvel algorithme basé sur des systèmes chaotiques et le système SHA-256 pour protéger les images contre les attaques. Le système est

divisé en deux parties : la première partie consiste en un processus de permutation à grande vitesse où une image simple est prise en entrée. Les vecteurs dans le processus de diffusion adaptative sont obtenus en itérant à travers le système chaotique en utilisant les paramètres de départ et les valeurs de l'image d'entrée. Ensuite, la règle de décalage binaire est appliquée pour quantifier ces vecteurs en tant que clé. À l'aide de la clé générée, le chiffrement et le déchiffrement sont effectués. [19]

Le texte clair sélectionné est testé avec des images spéciales telles que toutes les images noires ou toutes les images blanches. Les résultats obtenus de l'analyse des attaques par bruit et par occlusion montrent que l'algorithme peut résister à ces types d'attaques. Les images chiffrées présentent une qualité satisfaisante et l'algorithme montre également une entropie élevée par rapport à d'autres méthodes. L'algorithme de chiffrement est illustré dans la Figure 2.9.

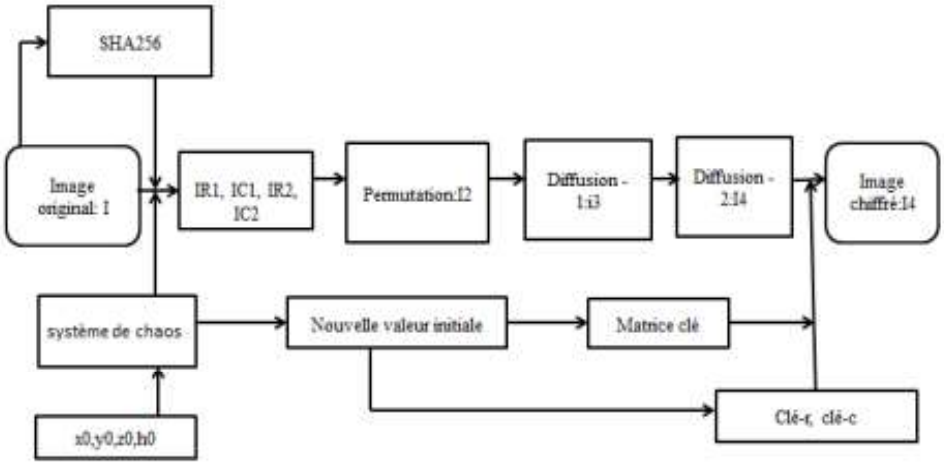


Figure 2.9 Schéma de l'algorithme de chiffrement propose. [19]

-N. K. Pareek et V. Patidar ont proposé une approche de chiffrement d'images basée sur des cartes logistiques. Leur algorithme utilise deux cartes logistiques. La première génère des nombres de 1 à 24 qui servent de condition initiale pour la deuxième carte logistique. En outre, un clé secrète externe de 80 bits ainsi que huit types d'opérations différents sont utilisés pour crypter les pixels d'une image. La décision quant à quelle opération utiliser pour un pixel particulier est déterminée par le résultat de la carte logistique. [19]

2.8 Conclusion :

Ce chapitre a examiné plusieurs algorithmes de chiffrement standard ainsi que des techniques basées sur les cartes chaotiques. Nous avons exploré le DES, le 3DES, RC4 et RC6, en mettant en lumière leurs différences en termes de taille de clé, de taille de bloc et de sécurité. Le DES, bien qu'historiquement significatif, est aujourd'hui considéré comme vulnérable en raison de sa clé relativement petite. Le 3DES a amélioré la sécurité en utilisant trois passes de DES, bien que cela ait entraîné une réduction significative de la vitesse. En revanche, RC4, malgré sa rapidité, présente des vulnérabilités connues qui limitent son utilisation sécurisée dans les applications modernes. RC6, avec sa flexibilité et sa capacité à supporter des clés longues, se distingue comme une option moderne et sécurisée.

En parallèle, les algorithmes basés sur les cartes chaotiques offrent des approches novatrices, exploitant le comportement complexe et imprévisible des systèmes chaotiques pour le chiffrement. Ces méthodes peuvent offrir une sécurité accrue en introduisant une couche supplémentaire de complexité mathématique, rendant plus difficile l'analyse cryptographique traditionnelle.

Chapitre 3 : L’algorithme de chiffrement AES

3.1 Introduction :

Dans le contexte actuel de la sécurité informatique, où les menaces et les attaques évoluent constamment, il est impératif de renforcer les algorithmes de chiffrement pour garantir la confidentialité et l'intégrité des données. L'algorithme AES (Advanced Encryption Standard), adopté comme standard de chiffrement par le gouvernement américain en 2001, est largement utilisé en raison de sa robustesse et de son efficacité. Cependant, comme tout algorithme cryptographique, AES n'est pas exempt de faiblesses potentielles, notamment dans ses composantes structurelles telles que les transformations SubBytes et ShiftRows.

Ce chapitre a pour objectif de détailler les principes fondamentaux de l'algorithme AES, en mettant l'accent sur ses opérations clés : SubBytes, ShiftRows, MixColumns et AddRoundKey. Nous explorerons les mécanismes internes de chacune de ces opérations, en soulignant leurs rôles et leurs implications en termes de sécurité et de performance. Par ailleurs, nous discuterons des vulnérabilités associées à certaines étapes du processus de chiffrement, particulièrement celles qui impliquent des tables de substitution fixes et des schémas de permutation prévisibles. Nous proposerons une amélioration de l'algorithme, désignée sous le nom de AES bis. Cette version modifiée intégrera des transformations dynamiques basées sur des cartes chaotiques, offrant ainsi une résistance accrue contre les attaques cryptographiques avancées.

3.2 Histoire de l'algorithme AES :

L'algorithme Advanced Encryption Standard (AES) est l'un des algorithmes de chiffrement par blocs publiés par le National Institute of Standards and Technology (NIST) en 2000 [38]. Les principaux objectifs de cet algorithme étaient de remplacer l'algorithme DES après l'apparition de certaines vulnérabilités de celui-ci. Le NIST a invité des experts en cryptographie et en sécurité des données du monde entier à proposer un algorithme de chiffrement par blocs innovant pour crypter et décrypter les données avec une structure puissante et complexe. [44]

De nombreux groupes du monde entier ont soumis leurs algorithmes. Le NIST en a retenu cinq pour évaluation. Après avoir appliqué divers critères et paramètres de sécurité, ils ont sélectionné l'un des cinq algorithmes, proposé par les cryptographes belges Joan Daemen et Vincent Rijmen. Le nom original de l'algorithme AES est l'algorithme de Rijndael. Cependant, ce nom n'est pas devenu populaire pour cet algorithme, qui est reconnu dans le monde entier sous le nom d'Advanced Encryption Standard (AES). [39]

3.3 Structure de base de L'algorithme AES :

AES est un chiffre itératif, différent du chiffre Feistel. Il repose sur deux techniques courantes pour crypter et déchiffrer les données, appelées réseau de substitution et de permutation (SPN). SPN consiste en un certain nombre d'opérations mathématiques effectuées dans des algorithmes de chiffrement par blocs. AES a la capacité de traiter des blocs de texte en clair de taille fixe de 128 bits (16 octets). Ces 16 octets sont représentés sous forme de matrice 4x4, et AES fonctionne sur une matrice d'octets. [12]

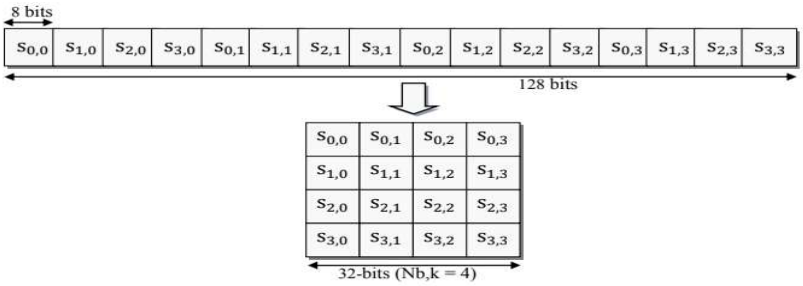


Figure 3.1 Transformation d'un bloc à une matrice 4 x 4 [24]

De plus, une autre caractéristique cruciale d'AES est le nombre de tours. Le nombre de tours dépend de la longueur de la clé. Il existe trois tailles de clé différentes utilisées par l'algorithme AES pour crypter et déchiffrer des données : 128, 192 ou 256 bits. La taille des clés détermine le nombre de tours, par exemple, AES utilise 10 tours pour les clés de 128 bits, 12 tours pour les clés de 192 bits et 14 tours pour les clés de 256 bits. [12]

Le tableau suivant définit les trois versions de l'algorithme AES selon la taille de clé et le nombre de tours.

	Longueur de Clé (NK : Mot 32bits)	**La taille de bloc (NK : Mot 32bits)**	**Nombre de rondes (Tour) (Nr)**
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Tableau 3.1 les trois versions de l'algorithme AES [24]

La figure 3.2 illustre la structure de base de l'algorithme AES.

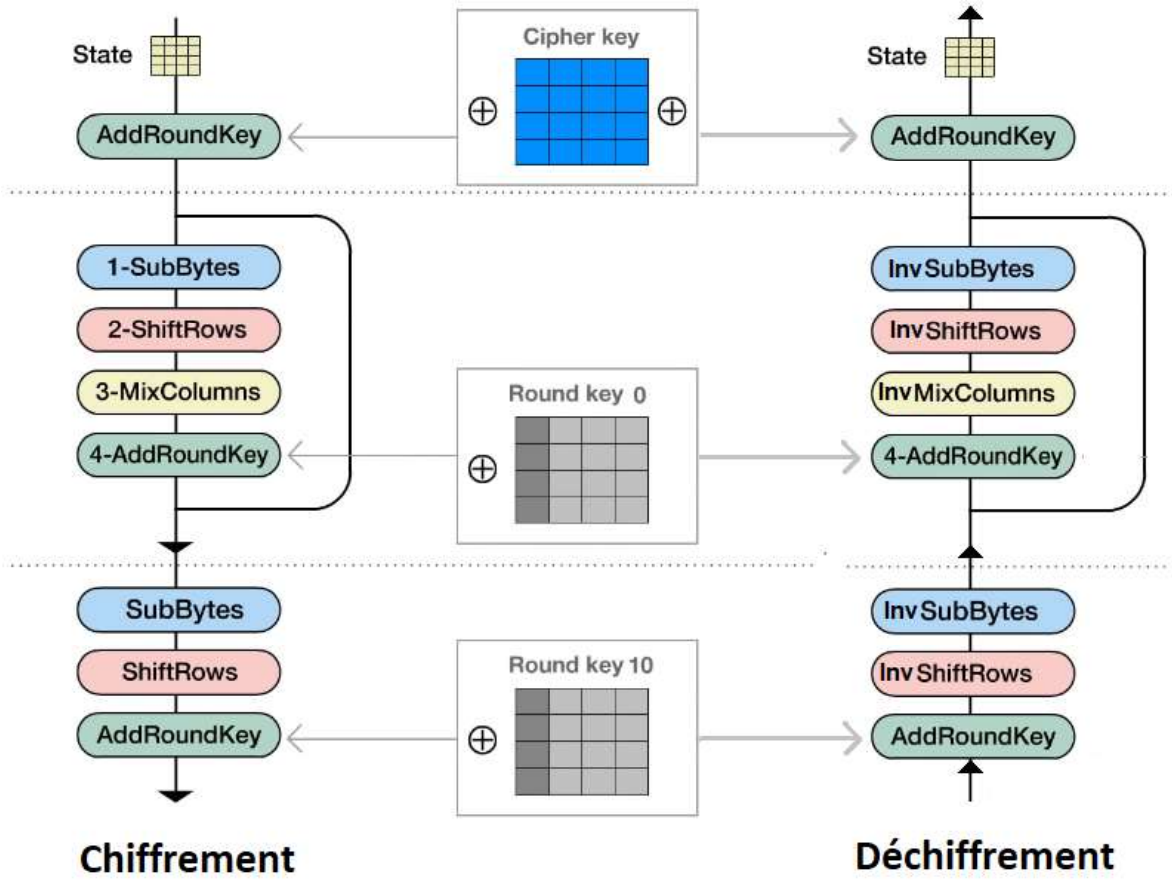


Figure 3.2 Structure de base de L'algorithme AES

3.4 Processus de chiffrement

Le chiffrement est une technique populaire qui joue un rôle majeur dans la protection des données contre les intrus. L'algorithme AES utilise une structure particulière pour chiffrer les données afin d'offrir la meilleure sécurité. Pour ce faire, il s'appuie sur un certain nombre de tours et, à l'intérieur de chaque tour, il comprend quatre sous-processus [2] [44]. Chaque tour inclut les quatre étapes suivantes pour chiffrer un bloc de 128 bits :

- SubBytes (Substitution des octets)
- ShiftRows (Décalage des lignes)
- MixColumns (Mélange des colonnes)
- AddRoundKey (Ajout de la clé de tour)

3.4.1 Transformation SubBytes :

La première étape de chaque tour commence par la transformation des sous-octets. Cette étape dépend de la S-box non linéaire pour remplacer un octet de l'état par un autre octet. Selon les

principes de diffusion et de confusion de Shannon pour la conception d'algorithmes cryptographiques, ces transformations jouent un rôle important pour obtenir une sécurité accrue. Par exemple, si nous avons l'hexadécimal 53 dans l'état, il doit être remplacé par l'hexadécimal ED. ED est obtenu à partir de l'intersection de 5 et 3 dans la S-box. Cette opération doit être effectuée pour chaque octet de l'état [24] [44]. La figure 3.2 présente la table S-box., et la figure 3.3 illustre Principe de fonctionnement de transformation SubBytes.

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

x

Tableau 3.2 Table S-box AES [44]

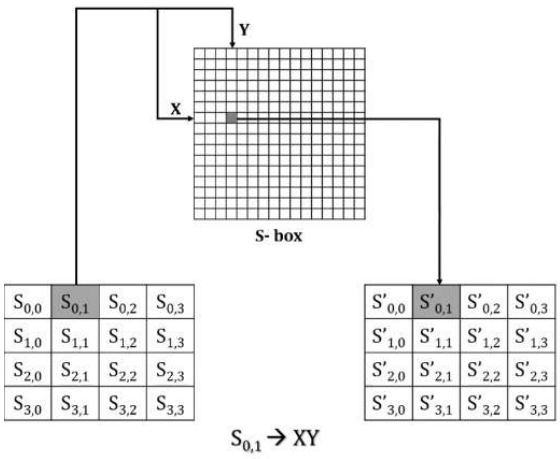


Figure 3.3 Principe de fonctionnement de SubBytes **[44]**

3.4.2 Transformation ShiftRows :

L'étape suivante après SubBytes qui s'exécute sur l'état est ShiftRows. L'idée principale derrière cette étape est de décaler cycliquement les octets de l'état vers la gauche dans chaque ligne, sauf la ligne numéro zéro. Dans ce processus, les octets de la première ligne restent inchangés. Dans la deuxième ligne, chaque octet est décalé circulairement d'une position vers la gauche. Dans la troisième ligne, chaque octet est décalé de deux positions vers la gauche. Dans la quatrième

ligne, chaque octet est décalé de trois positions vers la gauche [44] [24]. La taille du nouvel état n'est pas modifié et reste la même que la taille d'origine de 16 octets, mais la position des octets dans l'état est décalée, comme illustré sur la figure 3.4.

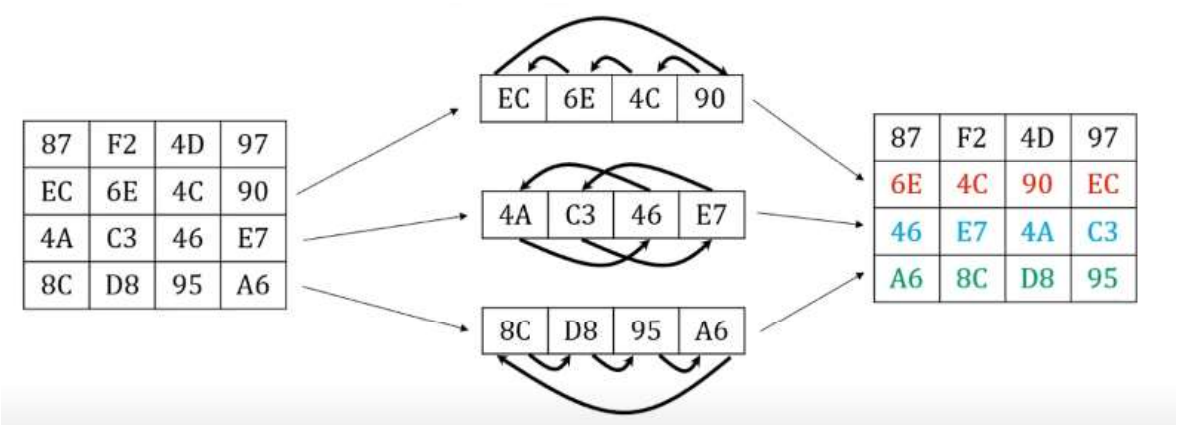


Figure 3.4 Transformation ShiftRows (exemple) [44]

3.4.3 Transformation MixColumns

Une autre étape cruciale de l'état est MixColumns. Dans cette étape, une multiplication matricielle est effectuée sur l'état. Chaque octet d'une ligne de la matrice de transformation est multiplié par chaque octet de la colonne de l'état. En d'autres termes, chaque ligne de la matrice de transformation est multipliée par chaque colonne de l'état. Les résultats de ces multiplications sont combinés à l'aide de l'opération XOR pour produire quatre nouveaux octets pour l'état suivant. La taille de l'état n'est pas modifiée et reste la taille d'origine de 4x4 octets, comme le montre la figure 3.5. [40] [44]

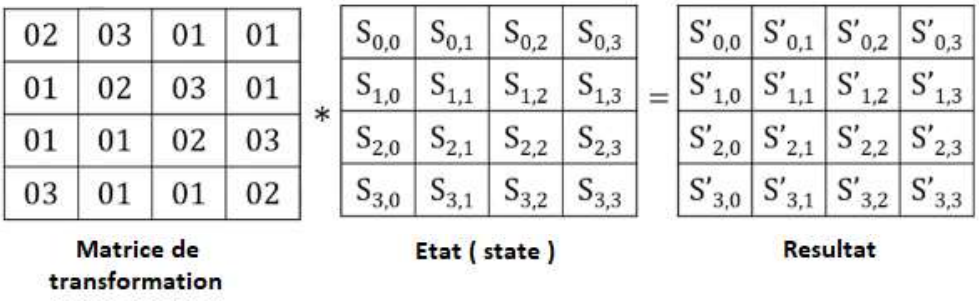


Figure 3.5 Transformation MixColumns [44]

3.4.4 Transformation AddRoundKey :

AddRoundKey est l'étape la plus vitale de l'algorithme AES [24]. La clé et les données d'entrée (également appelées état) sont structurées dans une matrice 4x4 d'octets. AddRoundKey a la capacité de fournir beaucoup plus de sécurité lors du chiffrement des données. Cette opération est basée sur la création de la relation entre la clé et le texte chiffré. Le texte chiffré provient de l'étape précédente. La sortie d'AddRoundKey repose exactement sur la clé indiquée par les utilisateurs. De plus, au cours de cette étape, la sous-clé est également utilisée et combinée avec l'état. La clé principale est utilisée pour dériver la sous-clé à chaque tour en utilisant le programme de clés de Rijndael. La taille de la sous-clé et de l'état est la même. La sous-clé est ajoutée en combinant chaque octet de l'état avec l'octet correspondant de la sous-clé en utilisant l'opération XOR au niveau du bit (voir la figure 3.6). [44]

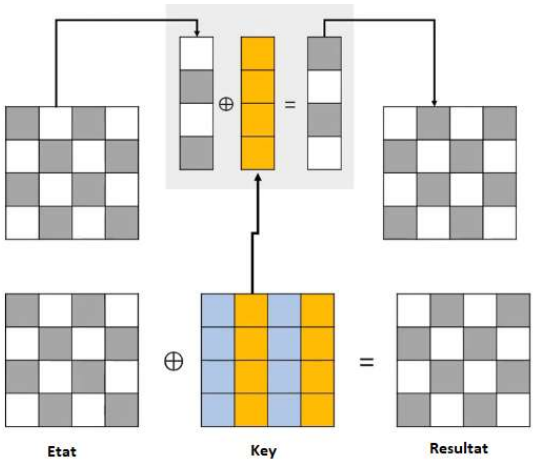


Figure 3.6 Transformation AddRoundKey

Figure 3.7 Transformation AddRoundKey (exemple)

3.5 Key Expansion (Génération des clés) :

L'algorithme AES repose sur l'extension de clé AES pour crypter et déchiffrer les données [12]. Il s'agit d'une étape cruciale dans la structure AES. Chaque tour utilise une nouvelle clé. Cette section se concentre sur la technique d'extension de clé AES. La routine d'expansion de clé crée

des clés de tour mot par mot, où un mot est un tableau de quatre octets. La routine crée 4x(Nr+1) mots, où Nr est le nombre de tours. Le processus est le suivant :

La clé de chiffrement (clé initiale) est utilisée pour créer les quatre premiers mots. La taille de la clé est de 16 octets (k0 à k15), comme le montre la figure 3.8 dans un tableau. Les quatre premiers octets (k0 à k3) représentent w0, les quatre octets suivants (k4 à k7) dans la première colonne représentent w1, et ainsi de suite. Nous pouvons utiliser une équation particulière pour calculer et trouver facilement les clés de chaque tour comme suit :

K [n]: w[i] = k [n-1]: w[i] XOR k[n]: w[i].

Cette équation est utilisée pour trouver une clé pour chaque tour à l'exception de w0. Pour w0, nous devons utiliser une équation particulière différente de celle-ci.

K[n]: w_0 = k [n-1]: w_0 XOR SubByte (k [n-1]: w_3>>8) XOR Rcon [i]. [44]

Avec : Rcon [i] est une constante, L'objectif de cette constante est de modifier la clé produite après le tour. Le tableau 3.3 présente les différentes valeurs possibles de Rcon.

J	1	2	3	4	5	6	7	8	9	10
RC[j]	**01**	**02**	**04**	**08**	**10**	**20**	**40**	**80**	**1B**	**36**

Tableau 3.3 Valeurs de R-con[j] en hexadécimal. [44]

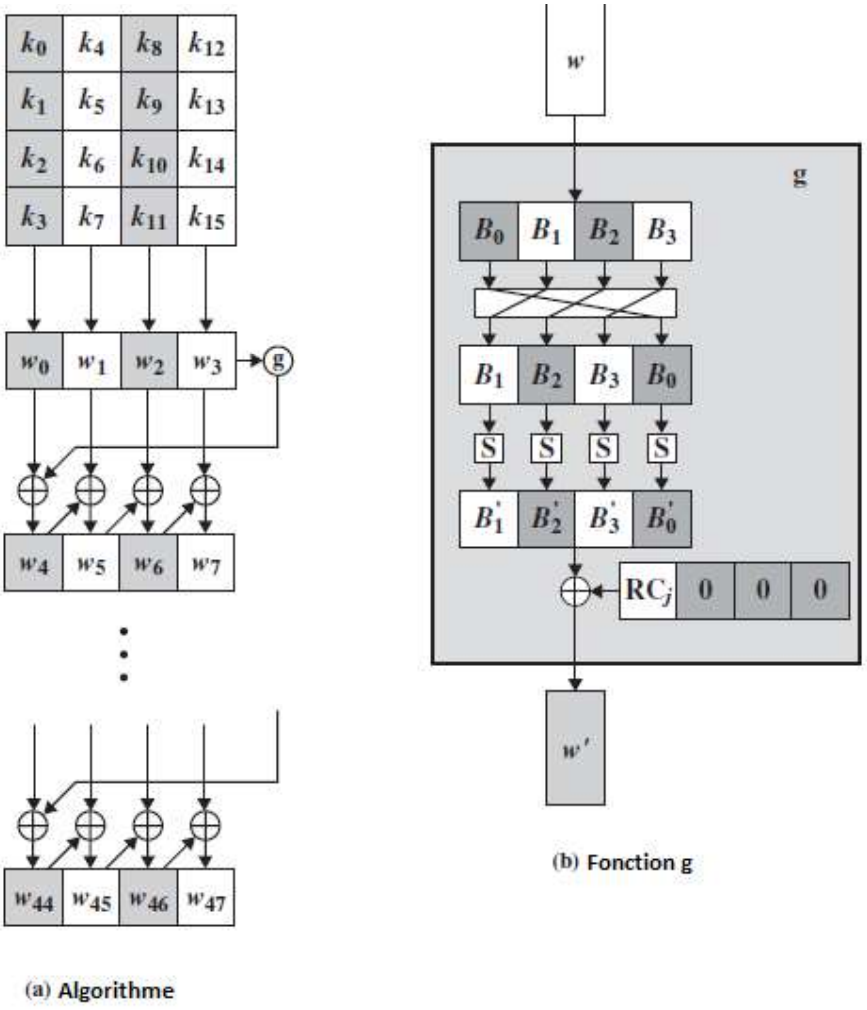


Figure 3.8 Key Expansion (Génération des clés) [44]

3.6 Processus de déchiffrement :

Le décryptage est le processus permettant de récupérer les données originales qui ont été cryptées [38]. Ce processus est basé sur la clé reçue de l'expéditeur des données. Les processus de décryptage dans AES sont similaires au processus de cryptage, mais exécutés dans l'ordre inverse. L'expéditeur et le destinataire utilisent la même clé pour chiffrer et déchiffrer les données. Le dernier cycle d'une étape de décryptage comprend trois étapes : InvShiftRows, InvSubBytes et AddRoundKey, comme illustré sur la figure 3.9. [38]

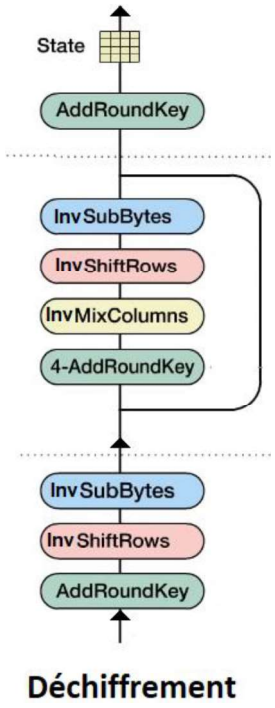


Figure 3.9 Processus de déchiffrement

Les étapes de chaque tour lors du décryptage sont les suivantes.

3.6.1 La Transformation InvSubBytes :

Inv_Sub_Bytes est une opération qui réalise la même fonction que SubBytes, mais en utilisant la table inverse S_box, qui est créée en inversant la table S-box [41]. La table suivante présente la table S-box inverse :

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Tableau 3.4 Table S-box inverse

3.6.2 La Transformation InvShiftRows :

La transformation InvShiftRows implique de déplacer les trois dernières lignes de la matrice state vers la droite [12]. La figure suivante illustre cette transformation.

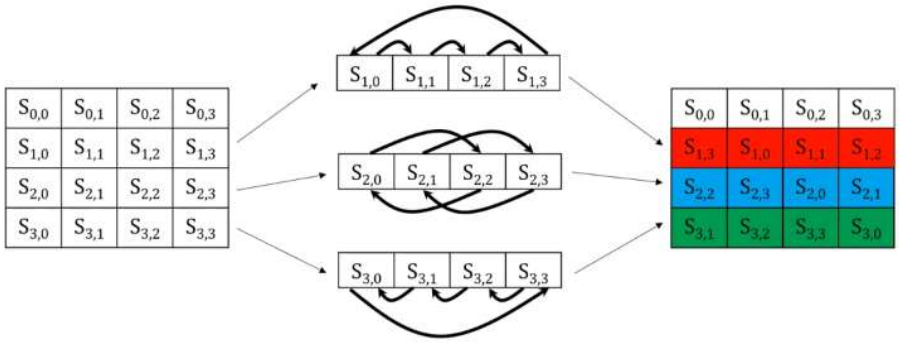


Figure 3.10 Transformation InvShiftRows [44]

3.6.3 La Transformation InvMixColumns:

La transformation InvMixColumns implique la transformation inverse de l'étape MixColumns [12]. Elle est utilisée dans le processus de déchiffrement pour restaurer l'état précédent avant le chiffrement, comme le montre la figure 3.11.

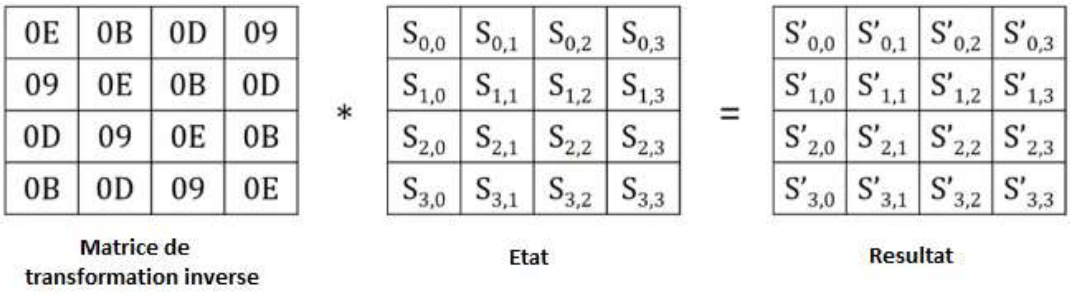


Figure 3.11 Transformation InvMixColumns

3.7 Caractéristiques de l'AES :

L'AES est une méthode de chiffrement qui satisfait parfaitement les exigences en matière de sécurité et de contraintes (temps réel, taille des données et ressources restreintes). Effectivement, l'AES propose un niveau de sécurité adéquat pour toutes les applications commerciales. Il se distingue par sa simplicité de calcul et sa rapidité de traitement, ses faibles

besoins en ressources et en mémoire, sa flexibilité d'implémentation, et la possibilité de l'ajouter à des applications logicielles ou matérielles [2].

A. Flexibilité : L'AES a été développé pour être flexible en ce qui concerne la taille des clés et du bloc de chiffrement. Il est possible de modifier l'AES en utilisant divers nombres de tours, même si ces caractéristiques n'ont pas encore été examinées en ce qui concerne les répercussions sur la sécurité de l'algorithme. Trois tailles de clés sont définies par l'AES : 128, 192 ou 256 bits. Cela représente environ 3,4 x 10^38 clés de 128 bits ; 6,2 x 10^57 clés de 192 bits et 1,1 x 10^77 clés de 256 bits. La construction de machines telles que le « DES Cracker » qui effectuait une recherche exhaustive sur une clé DES de 56 bits (7,2 x 10^16 clés) est rendue impossible. Si l'on suppose qu'une machine peut trouver une clé DES en une seconde (en essayant 2^55 clés par seconde), elle aurait besoin de 149 mille milliards d'années pour casser une clé de 128 bits.

B. Besoins en ressources et mémoire très faibles : L'AES est parfaitement adapté aux systèmes embarqués. En effet, il requiert une quantité limitée de mémoire, c'est pourquoi il est employé avec les cartes sans contact.

C. Sécurité : Il n'y a aucune attaque pratique de sécurité connue contre l'AES. Les boîtes S-Box sont utilisées comme composants non linéaires. Il semble que l'AES dispose d'une marge de sécurité adéquate. Il est en effet résistant aux attaques par cryptanalyse linéaire, cryptanalyse différentielle et attaques par dictionnaire. Il a également montré des résultats satisfaisants pour faire face aux attaques de temps et aux attaques de puissance dans des environnements avec peu de mémoire, tels que les cartes à puce. Ces attaques permettent d'extraire la clé secrète en utilisant des mesures de temps et de puissance fournies par la puce.

Cependant, comme tout algorithme cryptographique, il présente certaines faiblesses potentielles qui peuvent être exploitées dans certaines circonstances, notamment en ce qui concerne les S-Box (Substitution Boxes). Étant donné que les S-Box sont fixes et publiques, les attaquants peuvent analyser ces tables en profondeur à l'avance. Cela leur permet de rechercher des schémas ou des vulnérabilités potentielles sans la nécessité d'accéder directement au processus de chiffrement. Au cours des dernières années, des chercheurs ont proposé plusieurs méthodes pour sécuriser les tables S-Box, afin de renforcer la résistance de l'AES contre les attaques potentielles. [2]

3.8 AES bis : une amélioration proposée de l'algorithme AES :

À travers l'architecture de l'algorithme de chiffrement AES, il apparaît que deux étapes sont fixes dans le processus de chiffrement : SubBytes, où les boîtes de substitution (S-Box) sont fixes et publiques, et ShiftRows, où la méthode de déplacement des octets de la matrice est fixe et connue. Cette stabilité rend ces étapes une cible facile pour les voleurs de données.

Pour renforcer la sécurité et la sensibilité de l'AES, nous avons proposé une amélioration de l'algorithme, que nous avons nommé AES bis. Nous avons modifié deux transformations, à savoir ShiftRows et SubBytes, en introduisant des cartes chaotiques (la carte logistique et la carte de Hénon) selon le schéma suivant :

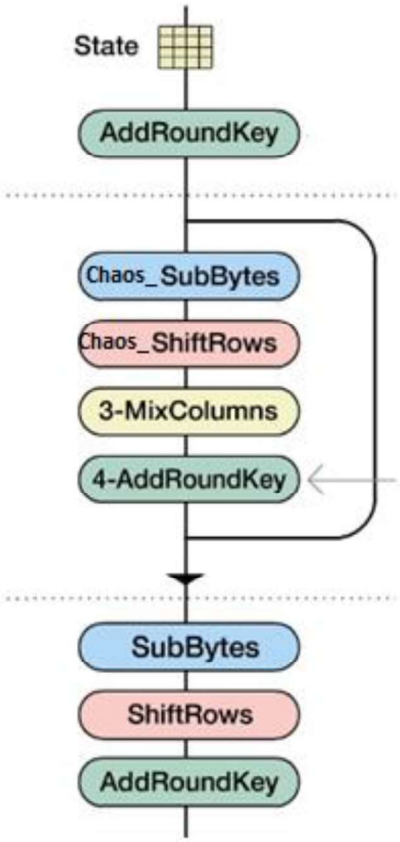


Figure 3.12 le schéma proposé.

A. Chao_SubBytes :

Nous avons adopté une nouvelle boîte de substitution (S-Box) en remplacement de l'ancienne par une boîte dynamique et secrète, tout en conservant les mêmes caractéristiques que la S-Box originale, telles que la non-linéarité. Pour générer de nouvelles S-Box, nous utiliserons de manière dynamique la carte chaotique de Hénon, selon le schéma suivant :

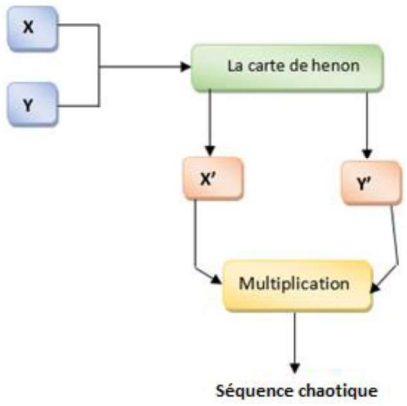


Figure 3.13 Le processus de la carte de Henon

Algorithme 3.1 :

```
Entrée : a ; b ; len = 256; x; y ;
Sortie : chaotic_sequence
for i=1:len
x(1,i+1) = 1-(a*x(1,i)^2) + y(1,i);
y(1,i+1) = b*x(1,i);
end
chaotic_sequence=x.*y;
```

Où :

x' et y' : des constantes en valeurs décimales qui déterminent le comportement du système. Les valeurs initiales x et y, déterminées par l'utilisateur, peuvent être considérées comme des sous-clés. a et b : les paramètres du système de Hénon.

Ces valeurs décimales doivent être converties en nombres entiers avant d'être utilisées dans la S-Box, en utilisant la méthode des indices. La méthode des indices consiste à ordonner la séquence chaotique générée et à utiliser ces indices ordonnés pour créer une S-Box. Cela permet d'introduire de l'aléatoire et de la non-linéarité dans le processus de substitution utilisé dans des algorithmes comme AES, renforçant ainsi la sécurité cryptographique.

D'abord, La séquence chaotique chaotic_sequence est créée en multipliant les séquences x' et y', ensuite, les indices de cette séquence chaotique sont triés (sort) et ajustés pour créer la S-Box. Dans ce cas, les indices sont décalés de -1 (indices - 1) pour correspondre à une S-Box où les indices commencent à zéro.

Exemple (5 valeurs)

- x' = [1.1060 , -0.6825, 0.6796, 0.1486, 1.1730]
- y'= [0.0300 , 0.3318 , -0.2048 , 0.2039 0.0446]
- chaotic_sequence= [0.0332, -0.2265, -0.1392, 0.0303, 0.0523]
- indices= [2 3 4 1 5]
- S-Box=indices-1 = [1, 2, 3, 0, 4]

De cette manière, nous convertissons les nombres décimaux en nombres entiers avec un ordre aléatoire, puis nous utilisons ces nombres pour remplir la S-Box. Notez que toute modification des valeurs de x ou y entraînera une modification complète de la S-Box.

Les Tableaux 3.5 et 3.6 présentent les résultats de la S-Box contenant 256 valeurs.

156	55	127	254	19	123	101	24	83	119	178	189	210	122	15	231
141	249	147	153	118	228	215	52	234	29	92	95	191	32	99	248
54	232	68	37	28	148	61	139	43	96	205	75	70	77	8	151
40	41	4	158	63	137	116	255	174	64	104	79	65	239	131	252
161	172	113	198	46	22	17	154	87	190	241	34	1	102	3	58
26	85	132	143	217	69	167	38	125	209	204	48	6	216	67	212
201	162	9	56	177	0	14	159	130	108	91	181	97	62	126	223
146	194	225	250	103	5	98	199	164	220	242	129	13	117	72	111
112	236	100	82	240	12	7	144	182	245	105	124	115	18	186	145
224	27	214	233	203	114	2	110	49	180	206	229	30	81	88	200
213	45	60	42	90	133	66	222	35	47	93	246	120	197	175	25
59	176	16	173	188	10	71	57	196	94	179	149	134	36	44	160
253	202	166	86	74	135	185	251	80	207	244	221	11	50	235	39
152	89	184	163	78	226	192	150	238	106	219	109	21	168	193	53
157	187	169	195	33	121	170	247	76	218	107	138	136	183	84	140
142	73	51	237	128	31	211	230	20	243	208	23	227	165	171	155

Tableau 3.5 Table S-Box

86	43	209	181	147	140	146	8	79	120	251	27	109	175	9	72
196	100	78	137	195	217	198	36	183	248	4	70	237	149	39	164
105	216	180	220	106	56	103	89	134	7	22	160	23	192	93	167
228	64	138	47	229	128	225	200	227	33	61	171	62	111	207	159
35	143	219	113	34	238	40	99	37	82	135	153	139	245	81	172
87	205	50	253	85	21	96	215	90	187	223	182	218	10	185	152
197	84	117	2	194	60	212	65	203	125	44	136	52	41	127	186
104	191	252	16	107	132	71	144	92	242	101	222	119	97	240	126
226	112	3	54	231	233	163	202	206	13	221	45	249	213	18	241
38	250	19	123	31	29	168	94	83	49	46	108	6	69	57	17
91	5	58	244	76	74	157	208	190	116	110	232	28	158	129	55
204	25	130	11	178	166	174	80	114	255	239	30	73	173	236	124
88	66	235	42	142	161	150	184	254	0	20	75	165	151	24	243
199	145	26	98	210	170	189	133	1	14	59	176	162	188	63	12
102	201	68	214	77	155	121	230	15	51	131	148	169	122	141	48
224	95	154	67	179	177	247	32	53	118	234	193	156	246	211	115

Tableau 3.6 Table S-Box inverse

B. Chao_ShiftRows:

Nous avons utilisé une méthode pour générer des valeurs dynamiques afin de remplacer les valeurs fixes de la transformation ShiftRows dans l'algorithme AES. Cette méthode utilise une fonction logistique pour produire une séquence de nombres pseudo-aléatoires. Le processus commence par l'initialisation d'une valeur de départ et d'un paramètre *R*, qui selon les chercheurs se situe entre 3,57 et 4. La séquence est ensuite générée en itérant cette fonction logistique sur 500 valeurs. Les valeurs sont modulées pour s'assurer qu'elles se situent dans une plage spécifique et sont multipliées par 10^7 pour augmenter la précision. Les zéros sont éliminés de la séquence pour assurer une meilleure randomisation. Finalement, nous avons choisi 40 valeurs parmi les 500 générées, spécifiquement entre les indices 121 et 161, et les avons réorganisées en une matrice de taille appropriée pour être utilisées dans la transformation ShiftRows.

Algorithme 3.2 :

```
Entrer : num=500; x(1); R=3.57
for i=1:num
x(i+1)=mod(R*x(i)*(1-x(i))*10^7,16);
end;
x=round(x);
x = x(x ~= 0);
final=x(122:161);
final=reshape(final,10,4);
```

Figure 3.14 le processus de la carte logistique

Nous avons stocké les valeurs de ShiftRows pour chaque tour dans un tableau, chaque ligne du tableau correspondant à un tour spécifique. Pour chaque élément de ces lignes, nous avons inclus l'indice de permutation des lignes de l'état (state). Le résultat des valeurs de ShiftRows est comme suite :

11	15	12	12	1 ére tour
15	15	15	14	
1	15	15	15	
4	5	7	12	
11	11	12	7	
3	15	9	11	
10	15	4	7	
2	15	10	6	
5	9	4	2	
8	12	10	1	10 éme tours

Tableau 3.7 Les valeurs de ShiftRows pour chaque tour.

3.9 Conclusion :

Dans ce chapitre, nous avons exposé le principe de l'algorithme AES ainsi que ses opérations fondamentales. Nous avons observé que les opérations SubBytes et InvSubBytes peuvent être réalisées en consultant respectivement les tables SBox et son inverse InvSBox. ShiftRows et InvShiftRows reposent sur des décalages simples vers la gauche et la droite. AddRoundKey implique des opérations d'addition exclusive (XOR) sur des opérandes de 8 bits. Nous avons noté que l'opération MixColumns et son inverse InvMixColumns sont les plus coûteuses en termes de complexité calculatoire, étant basées sur la multiplication d'un vecteur par une matrice.

Pour renforcer la sécurité et la résilience de l'AES, nous avons proposé une amélioration de l'algorithme, que nous avons nommée AES bis. Nous avons modifié deux transformations, à savoir ShiftRows et SubBytes, en introduisant des cartes chaotiques.

Chapitre 4 : Implémentation de l'algorithme de chiffrement AES

4.1 Introduction :

Dans ce chapitre, une analyse expérimentale approfondie de l'algorithme AES et de l'AES bis est présentée, en mettant l'accent sur leurs performances et leur sécurité. L'analyse de performance inclut des métriques telles que le temps d'exécution et la vitesse, tandis que l'analyse de sécurité couvre divers aspects comme le chiffrement, le déchiffrement, l'histogramme, l'entropie, la corrélation, le coefficient de corrélation et la sensibilité à la clé. Ces analyses permettent de vérifier non seulement l'efficacité des algorithmes, mais aussi leur capacité à résister aux attaques cryptographiques.

Pour évaluer la robustesse de l'AES et de l'AES bis, une comparaison a été effectuée avec d'autres algorithmes de chiffrement bien connus, notamment RC4 et RC6. Ces algorithmes ont été choisis en raison de leur popularité et de leurs caractéristiques distinctes, offrant un cadre de référence pertinent pour évaluer l'AES et les améliorations apportées par l'AES bis.

4.2 Analyse Expérimentale :

Cette analyse présente plusieurs tests effectués pour déterminer les performances et démontrer l'efficacité et la robustesse de l'algorithme de chiffrement AES et AES bis. Les performances ont été évaluées à l'aide d'une suite d'analyses incluant l'histogramme, le temps d'exécution, la vitesse, la corrélation des images, le coefficient de corrélation, l'entropie et la sensibilité des messages. Les résultats sont exécutés pour différentes tailles d'images telles que les images en niveaux de gris : « Cameraman » avec une résolution de 256 x 256 et une taille de 64 Ko , « Lena » avec une résolution de 256 x 256 et une taille de 84,3 Ko, ainsi que l'image "Département" avec une résolution de 128 x 128 et une taille de 25 Ko.

4.2.1 Présentation de la Plateforme Utilisée :

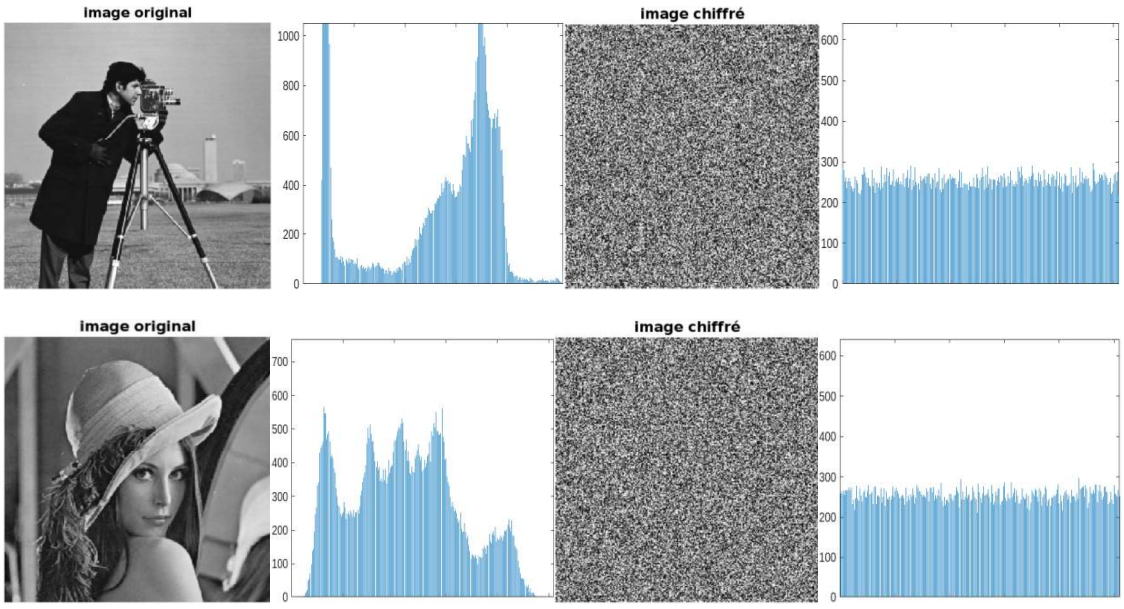
Pour l'analyse de performance et de sécurité de l'algorithme de chiffrement AES et AES bis, nous avons utilisé la configuration matérielle et logicielle suivante :

Processeur	Intel(R) Celerons (R) CPU N2840 @ 216GHz 2.16GHz
Fabricant	Intel
Vitesse	2GHZ
Nombre de cœurs	2
Mémoire	2GB
Système d'exploitation	Microsoft Windows 10 professionnel x64
Taille	64 bits
Environnement	Matlab R2013a

Tableau 4.1 Les paramètres du système.

4.2.2 Analyse d'histogramme :

L'histogramme d'une image représente la distribution des niveaux d'intensité de ses pixels. Un histogramme montre les valeurs de distribution des pixels. Par conséquent, l'histogramme de l'image chiffrée doit être uniformément distribué pour empêcher tout type d'attaques statistiques. Les figures 4.1 et 4.2 montrent que l'image chiffrée masque toutes les caractéristiques de l'image en clair et que les images chiffrées ont des histogrammes uniformes et nettement différents de l'histogramme des images en clair. Par conséquent, elles ne fournissent aucune information utile aux attaquants pour effectuer tout type d'attaque statistique sur l'image chiffrée. [41]



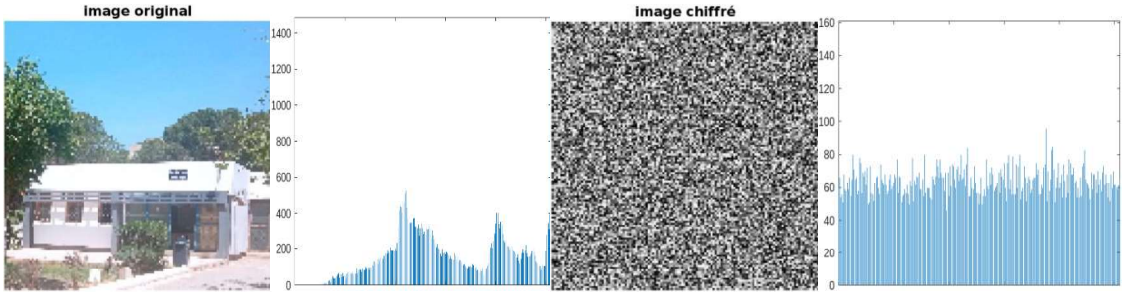


Image original son histogramme image chiffré son histogramme

Figure 4.1 : les images clairs et chiffré avec AES et leurs histogrammes.

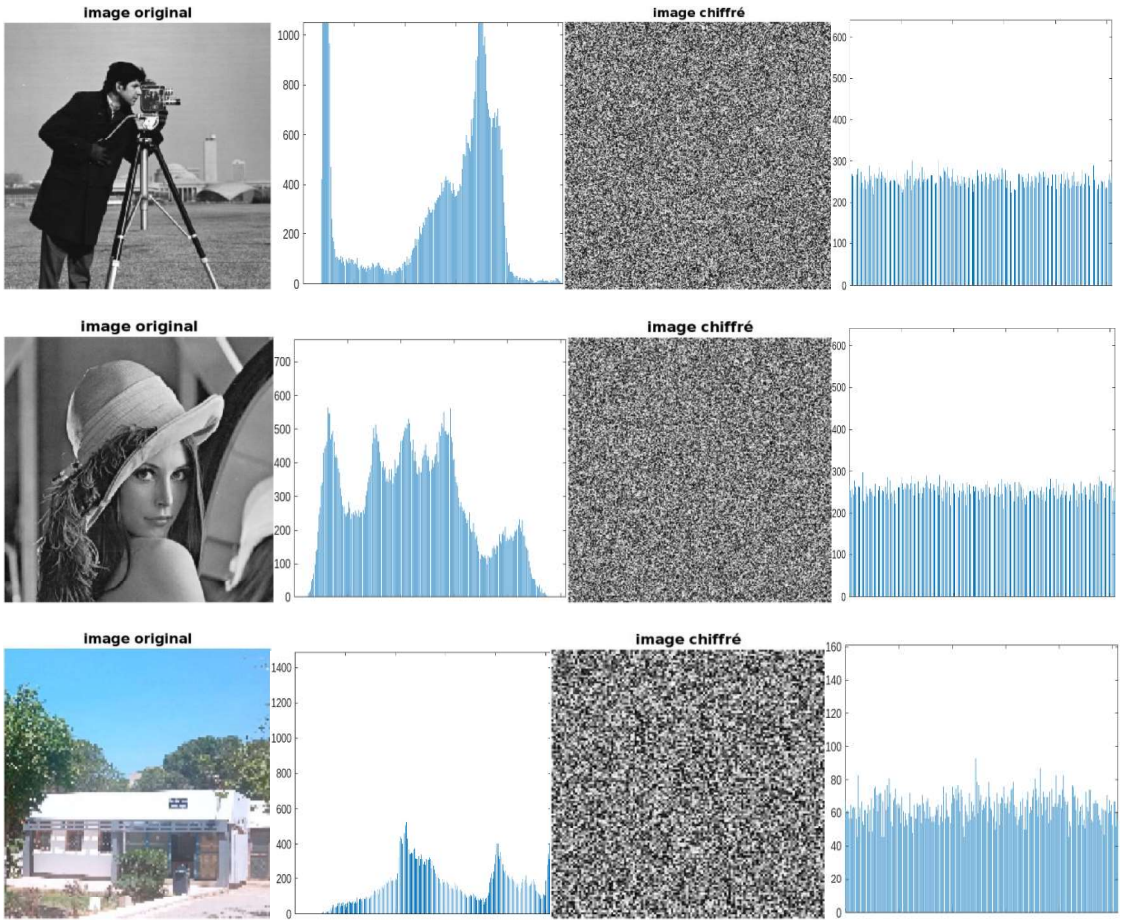


Image original son histogramme image chiffré son histogramme

Figure 4.2 Les images clairs et chiffré avec AES bis et leurs histogrammes.

4.2.3 Analyse de Corrélation :

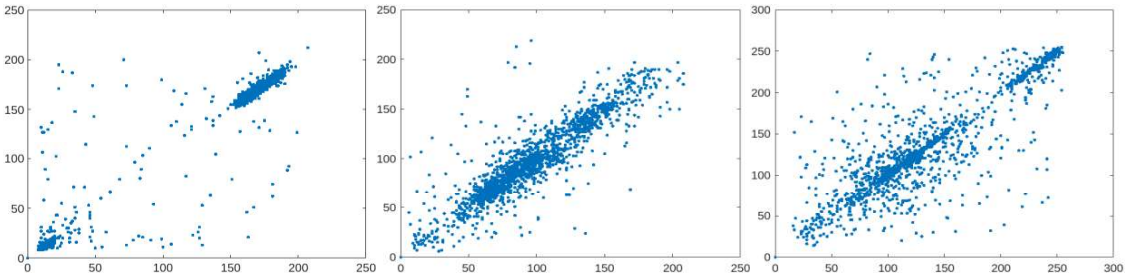
La méthode de corrélation est utilisée pour comparer deux images et analyser le mouvement des pixels dans l'une par rapport à l'autre, l'autre étant considérée comme l'image de référence.

La technique d'analyse de similarité entre les images numériques, connue sous le nom de corrélation d'images numériques (Digital Image Correlation ou DIC en anglais), permet de mesurer les déplacements entre les deux images en utilisant un style visuel bidimensionnel ou tridimensionnel. [42]

Dans les images non cryptées (originales), les valeurs des pixels adjacents sont fortement corrélées, ce qui signifie que la valeur d'un pixel peut prédire celle du pixel voisin. Pour vérifier la corrélation entre deux images, des paires aléatoires de pixels adjacents sont sélectionnées dans trois directions : horizontale, verticale et diagonale.

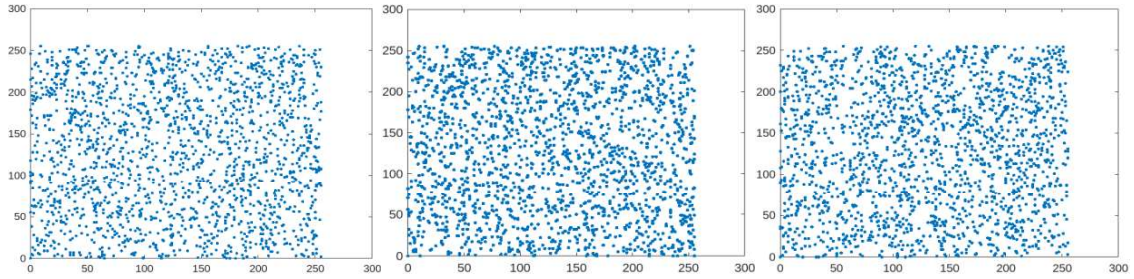
Dans le domaine du cryptage des images, le cryptage vise à briser cette relation entre les valeurs des pixels adjacents pour les rendre aléatoires et non corrélées, rendant ainsi difficile la reconstruction de l'image originale sans la clé de décryptage. Les distributions de deux pixels adjacents horizontalement montrent les différences entre l'image originale et l'image cryptée, où la corrélation dans l'image cryptée est très faible ou inexistante, ce qui est un indicateur de l'efficacité du cryptage.

Nous avons analysé la corrélation des pixels adjacents horizontaux dans les images originales et cryptées, et obtenu les figures (4.3), (4.4), (4.5) suivantes :



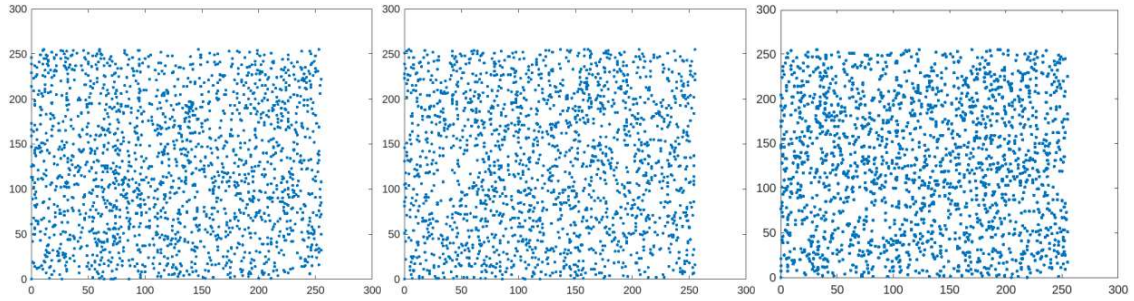
Cameraman Lena Département

Figure 4.3 Corrélation des images clair



Cameraman Lena Département

Figure 4.4 Corrélation des images chiffré avec AES



Cameraman Lena Département

Figure 4.5 Corrélation des images chiffré avec AES bis

En analysant les corrélations entre les pixels adjacents horizontaux dans les images en clair et celles chiffrées par les algorithmes AES et AES bis, il apparaît que dans le cas des images originales, il existe une forte corrélation entre les pixels adjacents horizontaux, où l'on peut voir que la plupart des pixels de l'image non chiffrée sont alignés de manière régulière. En revanche, dans le cas des images chiffrées par les deux algorithmes, les pixels adjacents horizontaux sont fortement disséminés. Cette dissémination des pixels indique clairement que les deux algorithmes sont robustes contre toute attaque statistique.

4.2.4 Analyse de coefficient de corrélation :

Le coefficient de corrélation est une mesure qui évalue la force de la relation linéaire entre deux variables lors de l'analyse de corrélation [36]. Il est représenté par "r" dans le rapport de corrélation et varie de -1 à 1. Plus le coefficient de corrélation se rapproche de 1 (ou -1), plus la relation entre les variables est forte, Un coefficient de corrélation de 0 indique l'absence de corrélation linéaire entre les variables. Pour ce calcul, nous avons utilisé la formule suivante :

$$r_{xy} = cov\,(x,y)/(\sqrt{var(x)}\,\sqrt{var(y)})$$

$$cov\,(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mathrm{E(x)})\,(y_i - \mathrm{E(x)})$$

$$\mathrm{E(x)}=\frac{1}{N}\sum_{i=1}^{N}(x_i) \quad \text{et} \quad \mathrm{E(y)}=\frac{1}{N}\sum_{i=1}^{N}(y_i)$$

$$var(x)=\frac{1}{N}\sum_{i=1}^{N}(x_i - \mathrm{E(x)})^2 \text{ et } var(y)=\frac{1}{N}\sum_{i=1}^{N}(y_i - \mathrm{E(y)})^2$$

Où x et y sont les valeurs de deux pixels adjacents dans l'image, cov(x,y) est la covariance, E(x) et E(y) sont les valeurs moyennes, var(x) et var(y) sont les variances et N est le nombre total de pixels.

Nous avons calculé le coefficient de corrélation entre deux pixels adjacents dans chacune des images originales et chiffrées par l'AES et l'AES bis. Les résultats ont été classés dans le tableau (4.2) suivant :

Sens	Image original	Image chiffré (AES)	Image chiffré (AES bis)
Horizontal (cameraman)	0,9334	0,0020	-5,4956e -04
Horizontal (Lena)	0 ,9213	0,0039	4 ,4300e -04
Horizontal (Département)	0,9719	-0,0080	-0,0021

Tableau 4.2 Analyse des coefficients de corrélation des images originales et chiffrées.

On observe que le coefficient de corrélation pour les images originales est très proche de 1, ce qui indique une forte corrélation entre les pixels adjacents. En revanche, dans les deux cas de chiffrement par AES et AES bis, le coefficient de corrélation tend vers zéro, ce qui témoigne d'une faible corrélation entre les pixels adjacents. Cela démontre l'efficacité des algorithmes dans la dispersion des pixels, rendant les images chiffrées fortement aléatoires et non susceptibles à une analyse statistique.

4.2.5 Analyse de l'entropie :

L'entropie quantifie la quantité minimale d'informations nécessaires pour communiquer un message de manière optimale [43]. L'estimateur de la probabilité d'apparition du niveau de gris $[i]$ dans l'image (variable aléatoire de dimension 2) est donnée par :

$$p_i = \frac{H[i]}{taille\ de\ l'image} = \frac{H[i]}{M \times N}$$

Ou p_i est la probabilité d'occurrence du niveau de gris, M est le nombre de pixels par ligne, N est le nombre de pixels par colonne et H est le nombre d'occurrence du niveau de gris.

Alors, l'entropie E de l'image est donnée par :

$$E = -\sum_i p_i \times log_2(p_i)$$

L'entropie d'une image est un indicateur de sa complexité.

Après l'évaluation de l'entropie, on obtient le résultat indiqué dans le tableau (4.3) suivant :

/	**Image original**	**Image chiffré (AES)**	**Image chiffré (AES bis)**
Entropie (cameraman)	7,0074	7,9972	7,9974
Entropie (Lena)	7,5954	7,9974	7,9972
Entropie (Département)	7,1878	7,9893	7,9883

Tableau 4.3 : Valeurs d'entropie des images chiffrées

On observe que les valeurs d'entropie des images chiffrées par l'AES et l'AES bis se rapprochent considérablement de la valeur théorique idéale, qui est de 8. Cela indique que les pixels dans les images chiffrées ont presque la même probabilité, ce qui permet à l'opération de chiffrement de réduire significativement toute fuite d'informations et renforce la sécurité du système de chiffrement.

4.2.6 Analyse de sensibilité :

NPCR et UACI sont généralement considérés comme la norme de mesure de la façon dont un léger changement du texte en clair ou de la clé affecte les images chiffrées. NPCR (Nombre De Pixels Change de Rage) et UACI (Intensité Variable Moyenne Uniforme) définis par les formules suivantes [43]

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\%$$

$$UACI = \frac{1}{M \times N}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\%$$

$$D(i,j) = \begin{cases} 0 \; si C_1(i,j) = C_2(i,j) \\ 1 \; si C_1(i,j) \neq C_2(i,j) \end{cases}$$

Où C1 et C2 sont deux images de même taille (M × N). Si C_1 (i, j) = C_2 (i, j) alors D (i, j)=1, Sinon D(i, j) = 0.

Le NPCR mesure le pourcentage du nombre de pixels différents par rapport au nombre total de pixels entre deux images. Tandis que l'UACI mesure la différence moyenne d'intensité entre les deux images.

Nous avons analysé le NPCR et UACI des images par l'AES et l'AES bis proposée, les résultats ont été classés dans le tableau (4.4) suivant :

Image	NPCR (AES)	UACI (AES)	NPCR (AES bis)	UACI (AES bis)
Cameraman	99,6246	31,1268	99,5712	31,2315
Lena	99,5926	30,5226	99,6109	30,5479
Département	99,6643	30,5579	99,6765	30,4726

Tableau 4.4 Valeurs NPCR et UACI entre L'image originale C1 et l'image chiffrée C2

Un score NPCR/UACI élevé indique généralement une résistance élevée aux attaques différentielles. Pour une image en niveaux de gris de 8 bits, les estimations attendues sont NPCR = 99,60% et UACI = 31,23%. Les résultats expérimentaux présentés dans le Tableau 4.4 indiquent que l'algorithme de chiffrement AES et l'algorithme AES bis proposé peuvent résister aux attaques différentielles.

4.2.7 Analyse de performance :

A. Temps d'exécution : Nous avons calculé les temps d'exécution pour le chiffrement et le déchiffrement pour chacun des deux algorithmes, et les résultats ont été répertoriés dans le tableau (4.5) suivant :

Etat	Temps d'exécution AES	Temps d'exécution AES bis
Chiffrement (Cameraman)	16 ,756006 seconds	23 ,363882 seconds
Déchiffrement (Cameraman)	32,107749 seconds	38,283774 seconds
Chiffrement (Lena)	17,073383 seconds	24,192833 seconds
Déchiffrement (Lena)	31,990605 seconds	38,631676 seconds
Chiffrement (Département)	4,236478 seconds	6,290251 seconds
Déchiffrement (Département)	7,928782 seconds	9,987989 seconds

Tableau 4.5 : Analyse de temps d'exécution.

B. La vitesse : Nous avons calculé la vitesse de chiffrement et de déchiffrement pour chacun des deux algorithmes, et les résultats ont été répertoriés dans le tableau (4.6) suivant :

Algorithme	**AES**		**AES bis**	
Image	**Vitesse de Chiffrement (bit/sec)**	**Vitesse de déchiffrement (bit/sec)**	**Vitesse de chiffrement (bit/sec)**	**Vitesse de déchiffrement (bit/sec)**
Cameraman	3,81952597	1 ,99328829	2,739270	1 ,67172651
Lena	4,93751004	2 ,63514866	3,48450303	2,18214711
Département	5,90112825	3,15306941	3,9744042	2 ,50300636

Tableau 4.6 Analyse de vitesse de chiffrement et de déchiffrement.

Nous remarquons que le temps de chiffrement et de déchiffrement de l'algorithme AES est légèrement plus court que celui de l'algorithme AES bis. Ainsi, la vitesse de chiffrement et de déchiffrement de l'algorithme AES est légèrement supérieure à celle de l'algorithme AES bis.

Discussion des résultats :

D'après les résultats de l'analyse des performances et de la sécurité des algorithmes AES et AES bis, nous pouvons affirmer que les deux offrent un niveau élevé de sécurité pour les opérations de chiffrement. En termes de performances, il y a quelques différences mineures entre les deux algorithmes, qui peuvent être négligées. En ce qui concerne la sécurité, la distribution des histogrammes et l'entropie des images chiffrées se rapprochent des valeurs théoriques idéales, ce qui indique l'efficacité des deux algorithmes à atteindre un haut degré de diffusion. De plus, les résultats expérimentaux de NPCR et UACI indiquent que l'algorithme de chiffrement AES

et l'algorithme AES bis proposé peuvent résister aux attaques différentielles. Cela signifie que lorsque nous avons modifié l'algorithme AES pour obtenir AES bis afin d'améliorer la sécurité, les performances et la sécurité de l'algorithme n'ont pas été affectées. Ainsi, on peut généralement compter sur les algorithmes AES et AES bis comme des algorithmes de chiffrement sûrs et efficaces.

4.3 Comparaison :

Dans cette section, nous avons comparé les algorithmes de chiffrement AES et AES bis proposés avec d'autres algorithmes courants, à savoir RC6 et RC4, en utilisant l'image Cameraman et en prenant en compte l'analyse des performances et de la sécurité.

4.3.1 Evaluation de la corrélation

La figure (4.6) suivante illustre les résultats de corrélation d'image avec les algorithmes AES, AES bis, RC6 et RC4. Nous observons que, dans le cas de l'image originale, il existe une forte corrélation entre les pixels adjacents horizontaux. En revanche, dans le cas de l'image chiffrée par l'algorithme RC4, les pixels adjacents horizontaux sont moins dispersés par rapport aux images chiffrées par les algorithmes AES, AES bis et RC6, qui sont fortement dispersés.

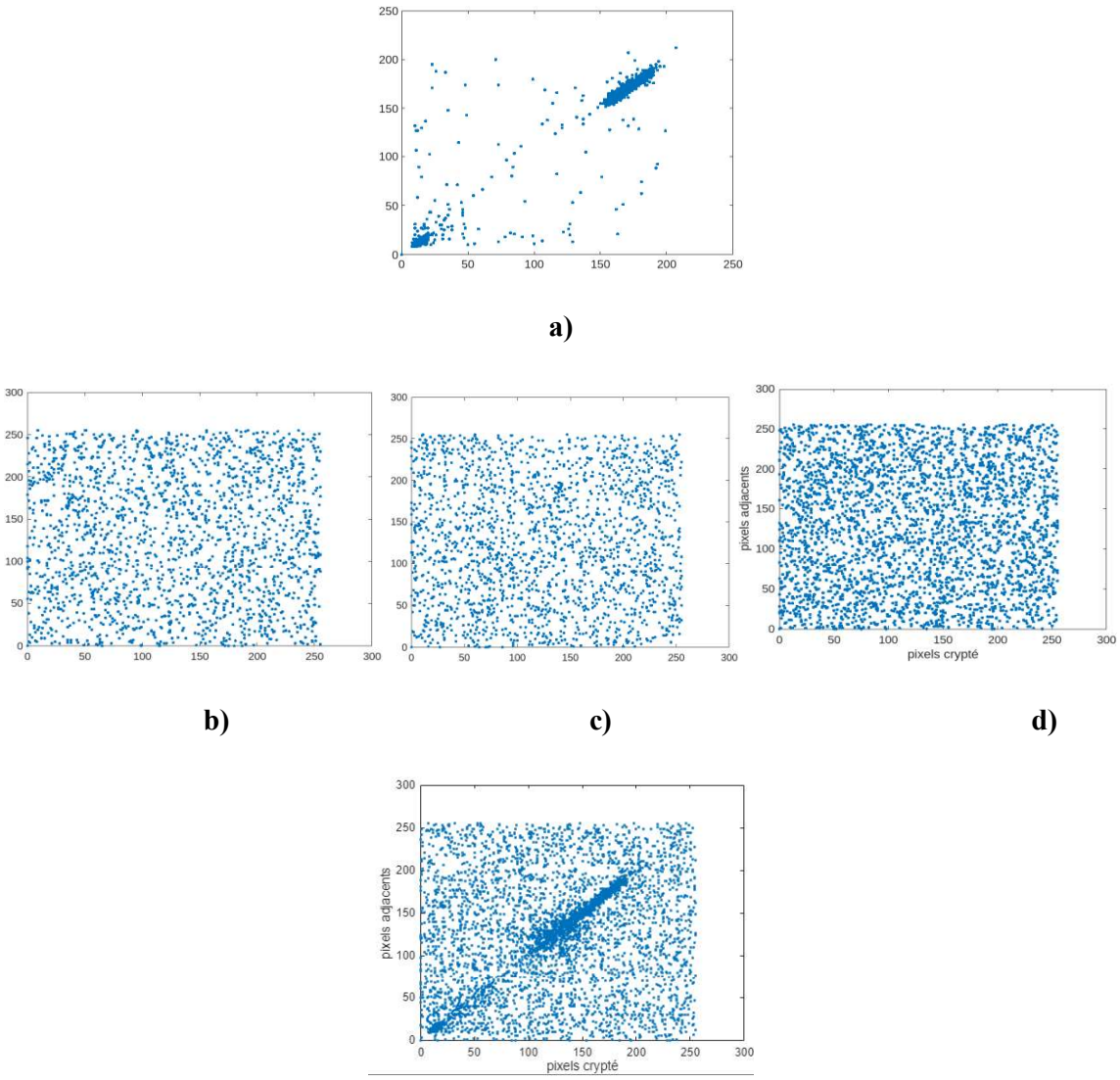


Figure 4.6 Résultats de corrélation d'image : a) image originale, b) avec AES, c) avec AES bis d) avec RC6, e) avec RC4

4.3.2 Evaluation de Coefficient de corrélation :

Nous avons calculé le coefficient de corrélation entre deux pixels adjacents dans chacune des images originales et chiffrées. Les résultats ont été classés dans le tableau (4.7) suivant :

Algorithme	AES	AES bis	RC4	RC6
Image original (Cameraman)	0,9334	0,9334	0.9334	0.9334
Image chiffré (Cameraman)	0,0020	-0,00054956	0,0017	0.0022
Image original (Lena)	0 ,9213	0,9213	0,8942	0,8942
Image chiffré (Lena)	0,0039	0,00044300	-0,0091	0,0011
Image original (Département)	0,9719	0,9719	0,0412	0 ,0319
Image chiffré (Département)	-0,0080	-0,0021	-0,0091	0,0214

Tableau 4.7 Coefficients de corrélation des images originales et chiffrées du cameraman, Lena et Département.

À partir du tableau 4.7, nous constatons que le coefficient de corrélation pour les images originales approche de 1, indiquant une forte corrélation entre les pixels, tandis qu'il approche de 0 pour les images chiffrées, indiquant une corrélation faible. Sur la base de ces résultats, nous concluons que les algorithmes sont sécurisés contre les attaques par corrélation.

4.3.3 Evaluation d'entropie :

Après l'évaluation de l'entropie, on obtient le résultat qui indiqué dans le tableau 4.8.

Les résultats du tableau indiquent que tous les algorithmes de chiffrement étudiés dans cette comparaison sont sécurisés, car l'entropie des images chiffrées est très proche de la valeur théorique de 8. Cette valeur signifie que les fuites d'information dans le processus de chiffrement sont presque négligeables.

Algorithme	AES	AES bis	RC4	RC6
Image original (Cameraman)	7,0074	7,0074	7.0097	7.0097
Image chiffré (Cameraman)	7,9972	7,9974	7.9971	7.9969
Image original (Lena)	7,5954	7,5954	7.5673	7.5673
Image chiffré (Lena)	7,9974	7,9972	7.9865	7.9873
Image original (Département)	7,1878	7,1878	**2,1810**	**7.9883**
Image chiffré (Département)	7,9893	7,9883	**3,9503**	**7,9891**

Tableau 4.8: Analyse de l'entropie des algorithmes AES, AES bis, RC4 et RC6.

4.3.4 Evaluation de la sensibilité :

A. Test NPCR : Nous avons analysé le NPCR et les résultats sont présentés dans le tableau (4.9) suivant :

Image	Cameraman	Lena	Département
NPCR (AES)	99.6246	99.5926	99,6643
NPCR (AES bis)	99,5712	99,6109	99,6765
NPCR (RC4)	99,6201	99,6231	99,5483
NPCR (RC6)	99,6292	99,6231	99,5972

Tableau 4.9 Analyse de NPCR.

Les valeurs NPCR élevées (proches de 100 %) pour tous les algorithmes et les images indiquent que le chiffrement modifie la majorité des pixels, ce qui est un bon indicateur de l'efficacité du chiffrement. [19]

B. Test UACI : Nous avons analysé le UACI et les résultats sont présentés dans le tableau (4.10)

Les valeurs UACI varient entre 30,07 et 31,2315, ce qui indique une variation moyenne de l'intensité des pixels entre les images originales et chiffrées. Tous les algorithmes ont montré des performances similaires avec des différences mineures. [19]

Image	Cameraman	Lena	Département
UACI (AES)	31,1268	30,5226	30,5579
UACI(AES bis)	31,2315	30,5479	30,4726
UACI(RC4)	31,2256	30,5976	30,0777
UACI (RC6)	31,1231	30,6650	30,4683

Tableau 4.10 Analyse de UACI.

4.3.5 Évaluation du temps de calcul :

Cette section indique les résultats obtenus à partir de l'exécution du programme d'évaluation en utilisant différentes tailles d'image. Les résultats montrent l'impact de la variation de la charge de données sur chaque Algorithme :

A. Temps d'exécution : Nous avons calculé le temps de chiffrement et de déchiffrement pour chaque algorithme, et les résultats sont présentés dans le tableau (4.11) suivant :

Algorithme / Temps (sec)	AES	AES bis	RC4	RC6
Temps de chiffrement (cameraman)	16 ,756006	23,363882	0.31725	71.758260
Temps de déchiffrement (cameraman)	32,107749	38,283774	0.20050	84.430235
Temps de chiffrement (Lena)	17,073383	24,192833	0.320304	20.413495
Temps de déchiffrement (Lena)	31,990605	38,631676	0.153339	23.859016
Temps de chiffrement (Département)	4,236478	6,290251	0,515996	1 ,712591
Temps de déchiffrement (Département)	7,928782	9,987989	0,747784	0,629514

Tableau 4.11 Temps de chiffrement et déchiffrement des algorithmes AES, AES bis, RC4, RC6.

B. La vitesse : Nous avons calculé la vitesse de chiffrement et de déchiffrement pour chaque algorithme, et les résultats sont présentés dans le tableau (4.12) suivant :

Algorithme / Temps (sec)	AES	AES bis	RC4	RC6
Vitesse de chiffrement (cameraman)	3,81	2,73	201,73	0,89
Vitesse de déchiffrement (cameraman)	1 ,99	1 ,67	319,20	0,75
Vitesse de chiffrement (Lena)	4,93	3,48	263,18	4,12
Vitesse de déchiffrement (Lena)	2 ,63	2,18	549,76	3,53
Vitesse de chiffrement (Département)	5,90	3,97	48,44	14,59
Vitesse de déchiffrement (Département)	3,15	2 ,50	33,43	39,71

Tableau 4.12 Vitesse de chiffrement et déchiffrement des algorithmes AES, AES bis, RC4, RC6.

Les résultats montrent que l'algorithme RC4 est de loin le plus rapide en termes de temps et de vitesse de chiffrement et de déchiffrement par rapport aux autres algorithmes. Il est suivi par l'algorithme AES, qui est plus rapide que l'AES bis et le RC6, mais nettement plus lent que le RC4. L'algorithme AES bis est légèrement plus lent que l'AES, mais il dépasse le RC6 en termes de vitesse. Quant à l'algorithme RC6, il est le plus lent en général, bien que ses performances s'améliorent avec certaines images comme "Lena" et "Département". Ces résultats montrent que le choix de l'algorithme de chiffrement dépend fortement de l'équilibre entre le besoin de vitesse et de sécurité.

Discussion et résultats :

D'après la comparaison des résultats d'analyse de performance et de sécurité des algorithmes de chiffrement AES, AES bis, RC4 et RC6, nous constatons que, du point de vue des performances, l'algorithme RC4 surpasse les autres algorithmes. En ce qui concerne la sécurité, les algorithmes AES, AES bis et RC6 montrent une supériorité.

Notre algorithme proposé AES bis utilise deux cartes chaotiques, Henon et logistique, ce qui le rend plus sensible aux conditions initiales. De plus, un petit changement dans les valeurs initiales peut complètement modifier le résultat du chiffrement. Ces valeurs initiales peuvent être considérées comme des clés, renforçant ainsi la sécurité de l'algorithme en augmentant la taille de la clé.

4.3.6 Comparaison générale

Critère	AES bis	AES	RC6	RC4	DES
Nom complet	Advanced Encryption Standard bis	Advanced Encryption Standard	Rivest Cipher 6	Rivest Cipher 4	Data Encryption Standard
Historique	2024	2001 (adopté come norme)	1998	1987	1977
Type de chiffrement	Chiffrement par bloc	Chiffrement par bloc	Chiffrement par bloc	Chiffrement par flux	Chiffrement par bloc
Taille du bloc	128 bits	128 bits	128 bits	/	64 bits
Longueur de clé		128,192,256 bits	128,192, 256 bits	De 40 à 2048 Bits	56 bits
Rond	10 ,12,14(selon la longueur de clé)	10 ,12,14(selon la longueur de clé)	20	/	16
Sécurité	Très élevée	Très élevée	Très élevée	Très Faible	Faible (cassé par la force brute)
Vitesse	Rapide	Rapide	Moyenne	Très rapide	Lent par rapport aux techniques modernes
Structure	Réseau de substitution, permutation	Réseau de substitution, permutation	Réseau de feistel modifié	Basé sur le flux (stream-based)	Réseau de feistel
Application	Chiffrement des données standard, des réseaux sans fil, des disques dures	Chiffrement Des données standard, des réseaux sans fil, des disques dures	Chiffrement des données dans les applications sensibles à la sécurité	Chiffrement des données dans les applications simples comme SSL et WEP	Remplacé par AES dans de nombreuse application

Tableau 4.13 Comparaison générale entre AES, AES bis, RC4 et RC6

4.4 Conclusion :

Dans ce chapitre, nous avons implémenté l'algorithme AES ainsi que notre proposition d'amélioration, AES bis, en termes de performances et de sécurité, en évaluant des métriques telles que l'histogramme, l'entropie, la corrélation, le coefficient de corrélation, ainsi que l'analyse de sensibilité. Nous avons ensuite comparé ces résultats avec ceux obtenus pour les algorithmes RC6 et RC4, utilisant les mêmes critères de performance et de sécurité.

Nos analyses ont révélé que l'algorithme RC4 se distingue par ses performances supérieures en termes de vitesse de chiffrement et de déchiffrement. Cependant, du point de vue de la sécurité, les algorithmes AES, notamment avec l'ajout de la composante chaotique dans AES bis, ainsi que RC6, démontrent une robustesse supérieure, notamment en termes de dispersion des histogrammes, de faible corrélation entre les pixels chiffrés, et d'entropie approchant les valeurs théoriques idéales. En particulier, l'utilisation de cartes chaotiques dans AES bis rend l'algorithme plus sensible aux conditions initiales, ce qui renforce la sécurité en augmentant la complexité des clés.

En conclusion, le choix de l'algorithme de chiffrement optimal dépend des exigences spécifiques en termes de vitesse, de sécurité et de robustesse contre les attaques. Tandis que RC4 offre des performances rapides, les algorithmes AES et AES bis, ainsi que RC6, se révèlent être des choix plus sûrs pour des applications nécessitant une sécurité renforcée et une résistance à diverses attaques cryptographiques.

Conclusion générale

Conclusion générale :

Les avancées dans l'utilisation d'Internet et des réseaux constituent un pilier essentiel du progrès technologique moderne. Dans ce contexte, le besoin urgent de garantir la sécurité et la protection dans le domaine des communications électroniques est devenu manifeste. Comprendre les failles et les défis auxquels est confrontée l'infrastructure réseau est essentiel pour renforcer la sécurité et protéger les données. Dans cette optique, les chercheurs et les ingénieurs adoptent de nombreuses technologies et outils pour atteindre cet objectif, où le cryptage et les algorithmes de sécurité jouent un rôle crucial dans la protection des informations contre les intrusions et les attaques cybernétiques. En se concentrant sur des applications telles que l'algorithme de cryptage AES, la compréhension est renforcée et des solutions efficaces sont développées pour garantir la sécurité des réseaux et la préservation intégrale des données.

Le principal objectif qui nous est confié est d'appliquer l'algorithme de cryptage avancé AES en utilisant le logiciel MATLAB, dans le but de sécuriser efficacement et rapidement le processus de cryptage et de décryptage des messages.

L'implémentation et l'évaluation comparative avec d'autres algorithmes comme RC4 et RC6 ont permis de mettre en lumière les forces et les faiblesses de chaque méthode de chiffrement, offrant ainsi des insights précieux pour le choix et l'application sécurisée des techniques de cryptage dans des environnements réels.

En intégrant des transformations dynamiques basées sur des cartes chaotiques dans AES bis, nous avons démontré une approche innovante pour renforcer la sécurité tout en maintenant des performances acceptables. Cette démarche illustre l'évolution continue de la cryptographie pour répondre aux défis croissants de sécurité dans un monde numérique en constante évolution.

Dans les travaux futurs, Nous cherchons à développer un système basé sur une clé secrète qui répond à nos exigences et aspirations en matière de protection des données et des informations. Grâce à ce travail, nous acquerrons une expertise étendue dans l'utilisation des technologies modernes de cryptage et de programmation, renforçant ainsi nos capacités et nous permettant d'atteindre avec succès nos objectifs dans le domaine de la sécurité des données et des réseaux.

Bibliographie

[1] BENDJEDDOU Saad, BENMAKHLOUF Zakaria, « Cryptage et compression conjoints des images numériques à base de fonctions chaotiques », mémoire de master, Université de Mohamed El-Bachir El-Ibrahimi - Bordj Bou Arreridj,2021-2022.

[2] Allou Said, Allouane Kahina, « Cryptographie et sécurité des Réseaux Implémentation de l'AES sous MATLAB », mémoire master, UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU,2008.

[3] ABACHE HOUSSAM, DEBBAH MOHAMMED ABDESLAM, « Etude des performances de quelques fonctions chaotiques dédiées au Cryptage de la parole », mémoire master, Université de Mohamed El-Bachir El-Ibrahim - Bordj Bou Arreridj, 2021-2022.

[4] T. BEKKOUCHE, « Développement et implémentation des techniques de cryptage des données basées sur les transformées discrètes », Thèse doctorat, Université FERHAT ABBAS SETIF-1.

[5] MOUSSAOUI Lina, CHOUAICHIA Safa, « Etude et simulation d'un crypto-système basé sur l'algorithme AES-GCM : Application au cryptage des images médicales », mémoire master, Université 8Mai 1945 – Guelma, Juin 2023.

[6] Alipacha hana, « implémentation de systèmes cryptographie et conception des générateurs d'aléas », diplôme doctorat, université Oran Mohamed Boudiaf, 2020-2021

[7] Guermat. N, « Implémentation d'un algorithme de cryptage sur un circuit FPGA », Mémoire de master, Université Mohamed Boudiaf - M'sila,2017.

[8] DAHMANE Zouhir, ABDELLI Lyamine, « Implémentation d'un algorithme de cryptage sur un circuit FPGA », mémoire master, Université Mohamed Boudiaf - M'silla, Mai 2017.

[9] A. Lan et B. Vandervelde, « Panorama des algorithmes de Cryptographie », 13 mars 2011.

[10] cryptographie & sécurité réseaux 2020/2021.

[11] les fichies du club alkindi, juillet 2021.

[12] BOUSALAH Malika,TIFOUR Yamina, « Contribution à la conception d'un crypto système symétrique flexible sur circuit FPGA », mémoire master, UNIVERSITE M'HAMED BOUGARA DE BOUMERDES,2015-2016.

[13] Hassan Noura : « Conception et simulation des générateurs, crypto-systèmes et fonctions de hachage basés chaos performants », thèse de doctorat, Université de Nantes 19 Janvier 2015.

[14] F. Arnault, « Théorie des nombres & cryptographie », cour2 D.E.A, Université de Limoges, 12 mai 2003.

[15] J. PHILIPPE Aumasson « SERIOUS CRYPTOGRAPHY A Practical Introduction to Modern Encryption »,2018.

[16] T. BEKKOUCH, « Développement et implémentation des techniques de cryptage des données basées sur les transformées discrètes » Thèse de Doctorat, Université FERHAT ABBAS SETIF-1, Faculté de Technologie.

[17] R. RIMANI « Sécurisation des images par une combinaison des techniques de chiffrement et de recalage d'image », Thèse de Doctorat, Université des Science et de la Technologie d'Oran MOHAMED BOUDIAF, 2021.

[18] A. BELOUCIF, « Contribution à l'étude des mécanismes cryptographes », Thèse de Doctorat en informatique, Université de BATNA-2, 22. Septembre.2016.

[19] Frikha Houria, Tellouche Ines, « Cryptage des images médicales à la base des cartes chaotiques », Université Mohammed Seddik Ben Yahia – Jijel, 2021 – 2022.

[20] Agaguena Houdjatoulah, Tifouti Miloud, « Etude et simulation d'un système de cryptage d'images à base de chaos », mémoire master, Université 8Mai 1945 – Guelma, Juin 2023.

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

[22] AZZOUZI Oussama, HADDADI Ferhat : « Plateforme de chiffrement/déchiffrement pour la sécurisation du stockage et de la transmission de l'information », thèse d'ingéniorat, école nationale supérieur de l'informatique, 2012.

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

[24] KHIDER Ali, YAALA Mohamed, « Implémentation de l'Algorithme de Cryptage AES sur un circuit FPGA », mémoire master, Université Dr. Yahia Farès de Médéa, 2012-2013.

[25] Ziani Rezki, Bouizeri Brahim, « Cryptographie : Approche Quantique », mémoire master, Université Mouloud Mammeri de Tizi-Ouzou,2016/2017.

[26] ZELLAGUI AMINE, « Synthèse et implémentation des fonctions de hachage appliquées à la signature électronique, Mohamed Boudiaf », mémoire doctorat, Université des Sciences et Technologies d'Oran,2022-2023,

[27] E. Yavuz, A novel chaotic image encryption algorithm based on content sensitive dynamic functions witching scheme, Optics and Laser Technology, Optics & Laser Technology, 114, 224-239, 2019.

[28] M. Alawida, A. Samsudin, J.S. Teh, R.S. Alkhawaldeh, A new hybrid digital chaotic system with applications in image encryption, Signal Process, 160, 45ñ58, 2019.

[29] Louzzani Noura, « Contribution à l'amélioration de la transmission sécurisée des images a base du chaos », (Doctorat en –sciences), Université Badji Mokhtar_Annaba, 2022.

[30] Beloucif Assia Informatique Contribution à l'étude des mécanismes cryptographiques L'obtention du diplôme de Doctorat en Soutenue 22 / 09 / 2016.

[31] Fekhr El Islam Khelil, « Les systèmes chaotiques pour le chiffrement Mémoire de fin d'études » Pour l'obtention du diplôme de Master Université Larbi Ben M'hidi - Oum El Bouaghi.

[32] Z.Tang,Y, Yang, S. XuandX.Z, C.Yu, Image Encryption with Double Spiral Scans and Chaotic Maps,Security and Communication Networks, 19 Article ID 8694678, 15 pages, 2019.

[33] internatinal gournal of computer application (0975-8887), « performance evaluation of RC6, blowfish,DES,IDEA , cast-128block ciphers, april 2013

[34] « cryptographie et sécurité informatique », Université de liège, note de cours provisoires 2009-2010 Renaud Dumont.

[35] MEHENNI AMIRA, « Etude et implémentation sur SoC-FPGA d'une méthode de cryptage symétrique », Université de Mohame d El-Bachir El-Ibrahimi - Bordj Bou Arreridj, 2021/2022.

[36] ZELLAGUI AMINE, NOURI RIAD, « Etude, implémentation et synthèse des algorithmes RC4, RC5 et RC6 », mémoire master, Université des sciences et de technologie Mohamed Boudiaf, 2015-2016.

[37] lakam bouchra, brahimi amina, « Préparation d'un programme basé sur un nouvel algorithme de chiffrement ». Université Mostapha istambouli, mascara.

[38]National Institute of Standards and Technology, Federal Information Processing Standard,197, The Advanced Encryption Standard (AES), P5-34, http://csrc.nist.gov/publications/fips/ fips197/fips-197.pdf, November 26, 2001. Télécharger le 7 Février 2013,

[39] RENAUD Dumont. Introduction à la Cryptographie et à la Sécurité informatique, Faculté des Sciences Appliquées, Université de Liège, 2006-2007. 46,47, 66-71.

[40] Zine El Abidine ALAOUI ISMAILI, Ahmed MOUSSA: « Self-Partial and Dynamic Reconfiguration Implementation for AES using FPGA» , IJCSI International Journal of Computer Science Issues, Vol. 2, National School of Applied Sciences, Morocco, 2009.

[41] Sambasiva Reddy, Y. Amar Babu : «Evaluation of Microblaze and Implementation of AES Algorithm using Spartan-3E», Babu2 International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering ,Vol. 2, Issue 7, Juillet 2013.

[42] Hadj Brahim Abderrahmane, « intitule cryptogtaphy and compressed sensing », diplôme doctorat, université Oran Mohamed Boudiaf, 2020-2021.

[43] Mr ALI CHERIF Khalfallah, « Intitulé Synthèse et implémentation des algorithmes de chiffrement des images », diplôme doctorat, université Oran Mohamed Boudiaf, 2020-2021.

[44] Ako Muhamad Abdullah, MSc Computer Science –UK, « Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data », June 16, 2017, ako.abdullah@univsul.edu.iq.

Site web

[45] https://www.apprendre-en-ligne.net/crypto/blocs/feistel.html

[46] https://koor.fr/Java/Tutorial/java_operateurs_binaires.wp

[47] https://cermics.enpc.fr/polys/info1/main/node24.html

[48] https://web.maths.unsw.edu.au/~lafaye/CCM/crypto/des.htm

[49] https://www.precisely.com/blog/data-security/aes-vs-des-encryption-standard-3des-tdea

[50] https://fr.wikipedia.org/wiki/Cryptographie_sym%C3%A9trique