REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE M'HAMED BOUGARA-BOUMERDES

Faculté de **T**echnologie
Département **I**ngénierie des **S**ystèmes **E**lectriques

Mémoire de Master

Présenté par :

HAMADOUCHE Dalila

MOKRANI Cylia

Filière : Télécommunications

Spécialité : Réseaux et Télécommunications

Thème

Etude et Analyse par simulation d'un Cryptosystème par flot Rivest Cipher 4 (RC4)

Soutenu le ……../……../2021 devant le jury composé de:

ACHELI	**Dalila**	**Professeur**	**UMBB**	Président
HAICHOUR	***A*mina Selma**	MCB	***E*SI**	Examinateur
HAMADOUCHE	**M'Hamed**	**Professeur**	**UMBB**	Rapporteur

Année Universitaire : 2020/2021

Remerciement.

Nous remercions tout d'abord, le bon dieu tout puissant de nous avoir donné la santé, la volonté, la force, le courage, et la puissance pour pouvoir surmonter les moments difficiles, et atteindre notre objectif.

Nous souhaitons vivement remercier et exprimer notre gratitude à notre promoteur Mr HAMADOUCHE pour nous avoir proposé ce sujet ainsi pour qui est très reconnaissants pour ses remarques, ses orientations, son suivi, ses conseils pendant toute la réalisation de notre projet.

Ainsi que tous nos enseignent qui nous ont enseigné durant nos études au département de Génie Electriques.

Nous tenons à remercier tous nous collègues d'étude, particulièrement notre promotion2021

Nous disons mille fois merci, à notre famille, pour leur soutien et leurs encouragements ainsi qu'à tous ceux qui ont contribué de prés ou de loin à la réalisation de ce modeste mémoire soit par un aide, un encouragement ou même un sourire.

Finalement, nous remercions tout, les membres jury, devant qui nous avons l'honneur d'exposer notre travail, et qui ont pris la peine de lire avec soin ce mémoire.

Dédicaces

A Celle qui m'a donné la vie et l'envie de vivre, à celle qui m'a entouré de sa tendresse, à celle qui a attendu avec patience les fruits de sa Bonne éducation, à ma très chère
Maman GHALIA.

A celui qui a été toujours a mes cotés pour me soutenir et m'encourager, à celui qui m'indique la bonne voie en me rappelant que la volonté fait toujours les grandes choses, à
Mon adorable Papa Ferhat.

A mes sœurs qui ont partagé avec moi tous les moments d'émotion, elles m'ont chaleurement encouragé tout au long de mon parcours :
Nassima, Assia, Linda, wahiba.

A mes très chers frères : Fateh, Rabah, Sofiane, Moh et Yacine.

A mes petites chères : Fulla, Idris, Alaa, Momoh, Fouad, Maria, Adlan, Anais, Sofia, Razan, Iyad, Ilyas.

A mon Binôme : Cylia et toutes sa famille

A tous mes amies : Djahida, Manel, Thilleli, Lisa, Samia, Amel, Nesrine, Imane, Yousra.

Sans oublier tous mes amis d'enfance et du long parcours scolaire et universitaire et a tous ceux qui m'aiment.

DALILA

Dédicaces

A Celle qui m'a donné la vie et l'envie de vivre, à celle qui m'a entouré de sa tendresse, à celle qui a attendu avec patience les fruits de sa Bonne éducation, à ma très chère
Maman FERROUDJA.

A celui qui a été toujours a mes cotés pour me soutenir et m'encourager, à celui qui m'indique la bonne voie en me rappelant que la volonté fait toujours les grandes choses, à
Mon adorable Papa MOHEMMED.

A mes grands parents paternels Amar et Yamina que DIEU ait pitié de leurs âmes en PAIX.

A mes grands parents maternels Mohemmed et Tassadit que DIEU les bénisses.

A mes très chères tantes : Nadia, Dalila, Djamila, Soraya pour leur encouragement permanents tout au long de mon parcours universitaire et leurs soutien moral.

A mes très chères sœurs: Malika, Djamila, Nacira, Hakima, Lydia, Kamilia.

A mes très chers frères : Merzak, Smail, Fares pour leur appui.

A mes très chères cousines : Rym, Ahlem, Yasmine.
A mes petites chères : Fares, Amar, Mahedi, Kamel, Marwa, Merouane, Adel, Yasmine, Oussama.

A ma chère Binôme : DALILA et à toute sa famille.

A tous mes amies : Wissem, Djahida, Amel, Nesrine, Imane, Yousra.

CYLIA

ملخص:

التشفير ، الذي أصبح تخصصًا في حد ذاته ، يهتم بكل من الجيش (مجال الإرسال على سبيل المثال) والمدنيين (البريد والاتصالات السلكية واللاسلكية ، وأي فأي ، إلخ....)

يعد مشروع نهاية الدراسة لهذا المعلم جزءًا من سياق التحليل لنظام تشفير تدفقRC4

حيث تكمن اهتماماته في `` التخليط " وتشفير الرسائل أو الصور، يتغير المفتاح في كل مرة سواء قمت بتشفير رسالة أو صورة. كما أن متانة هذا النظام تعتمد على حجم المفتاح

يتم إجراء التحليل بواسطة Matlabعلى أمثلة ملموسة

النتائج التي تم الحصول عليها مقنعة ونقترح تنفيذها علىFPGA

Résumé :

La cryptographie qui est devenu une spécialité a part entière intéresse aussi bien les militaires (domaine de transmission par exemple) que les civiles (postes & télécommunication, wifi etc.….)

Ce projet de fin d'études master s'inscrit dans le contexte d'analyse le cryptosystème par flot le RC4 (Rivest Cipher 4) ou son intérêt réside dans le 'scrambling' et le chiffrement de messages ou images, la clé changea chaque fois qu'on chiffre un message ou une image. Aussi la robustesse de ce système dépend de la taille de clé.

L'analyse par Matlab est faite sur des exemples concrets avec aussi le déroulement de l'algorithme a la main.

Les résultats obtenus sont probants et nous suggérons sont implémentation sur FPGA.

Summary:

Cryptography, which has become a specialty in its own right, interests both the military (transmission field for example) and civilians (posts & telecommunications, wifi etc.….)

This master's end-of-study project is part of the analysis context of the RC4 (Rivest Cipher 4) flow cryptosystem where its interest lies in the 'scrambling' and the encryption of messages or images, the key changed each time whether you encrypt a message or an image. Also the robustness of this system depends on the key size.

The analysis by Matlab is carried out on concrete examples with also the progress of the algorithm by hand.

The results obtained are convincing and we suggest its implementation on FPGA.

Sommaire

Sommaire

Sommaire

Sommaire

Liste des Figures :

Liste des Figures

Liste des Tableaux :

Liste des abréviations :

1) **DES :** Data Encryption Standard.
2) **AES:** Advenced Encryption Standard.
3) **RC4 :** Rivest Cipher 4 ou Ron's Code 4.
4) **RSA :** Rivest, Shamir et Adelman.
5) **RC5 :** Rivest Cipher 5.
6) **RC6 :** Rivest Cipher 6.
7) **IDEA:** International Data Encryption Algorithm.
8) **Mode ECB:** Electronic code book.
9) **Mode CBC :** Cipher bloc chaining.
10) **Mode CFB:** cipher feedback.
11) **Mode OFB :** output feedback.
12) **GSM :** Global System for Mobile.
13) **ARC4 :** pour Alleged RC4.
14) **KSA :** Key Scheduling Algorithm.
15) **PRGA :** Pseudo Random Generator Algorithm.
16) **http :** HyperText Transfer Protocol.
17) **SQL :** Structured Query Language.
18) **MD5 :** Message Digest développé par Rivest en 1991.
19) **WEP :** Wired Equivalent Privacy.
20) **WPA :** Wifi Protected Access.
21) **TKIP:** Temporal Key Integrity Protocol.
22) **SSL/TLS:** Secure Sockets Layer/Transport Layer Security.
23) **BB84 :** Bennett et Brassard en 1984.
24) **IP :** Internet Protocol.
25) **PGP:** Pretty Good Privacy.

Introduction générale :

Introduction générale :

Depuis l'antiquité l'homme a été toujours préoccupé par la problématique de la sécurité sous toutes ses formes. L'apparition de l'informatique et les télécommunications a accentué la complexité des problèmes et des solutions de sécurité, en introduisant des nouvelles notions telles que les virus informatiques, accès non autorisé, la non identification et la fausse information…

La construction d'un système de sécurité fait appel quasi inévitablement aux notions de la cryptologie qui recouvre la cryptographie et la cryptanalyse. La cryptologie n'a pas cessé de progresser, d'évoluer et de se généraliser dans tous les domaines tels que la téléphonie, le stockage des données et les télécommunications satellitaires…

Pour faire face à ces différentes menaces, il faut mettre en œuvre un système de sécurité capable de qui répondre aux exigences et aux aspirations d'une politique de sécurité**[14].**

Les techniques de sécurisation actuellement les plus répandues sont issus du domaine de la cryptographie. La cryptographie est l'un des domaines les plus importants de notre société moderne. Aujourd'hui, tout est basé sur des systèmes de sécurité informatique, régis par des cryptosystèmes de plus en plus perfectionnés. Si ces derniers sont défaits, tout notre système va s 'effondre. Alors, si la conception de ces cryptosystèmes est très recherchée, c'est parce que les attaques conduites contre les techniques cryptographiques sont de plus en plus efficaces**[13].**

L'avancée de la cryptologie a incontestablement été la publication des algorithmes de cryptographie (RSA, DES, L'AES…)**[14].**

L'objectif de ce travail, est de faire une étude et simulation de RC4 avec une explication détaillée sur le fonctionnement de cet algorithme, comment il chiffre les messages, quelles sont les opérations effectuées pour ce chiffrement. Aussi, des simulations du cryptosystéme RC4 avec des exemples de chiffrement des messages et des images sous MATLAB sont considères.

Après cette introduction générale, la suite de rapport est organisé de la manière suivante :

- **Le premier chapitre** comporte un ensemble de notions et de généralités sur la cryptographie, son historique et l'objectif de sécurité de la cryptographie et ses mécanismes.
- **Le deuxième chapitre** présente l'étude de la cryptographie classique avec quelques exemples des premiers systèmes de chiffrement les plus connus soit manuel, mécanique ou électromécanique.
- **Le troisième chapitre** dans ce chapitre, nous nous intéressons à l'évolution des techniques cryptographiques par l'utilisation de la théorie de nombres de l'information et la physique quantique.
  Nous donnons à ce titre quelques exemples.
- **Le dernier chapitre** est dévolu au cryptosystème RC4, son fonctionnement de chiffrement et déchiffrement avec un exemple de message bien expliqué et détaillé, et on termine avec des simulations de chiffrement et déchiffrement des messages et des images sous MATLAB.

Nous terminons ce manuscrit par une conclusion générale et quelques perspectives.

Chapitre1 :

Généralité sur la Cryptographie

Introduction :

L'information est un élément constitutif et déterminant dans tous les domaines. Tout au long de l'histoire, l'humanité a essayé d'envoyer des informations d'une façon sécurisée. Alors, la question qui se pose est comment assurer la sécurité d'échange d'information ?

La cryptographie a toujours été une nécessité militaire : de tous temps les armées ont dû transmettre des messages confidentiels de telles manière qu'ils soient incompris par un ennemi.

Ce Chapitre donne un aperçu d'ensemble d'historique, des notions élémentaires sur la cryptographie et terminologie employée dans ce domaine. Il décrit aussi certains objectifs de sécurité de la cryptographie.

I .1. Histoire de la cryptographie:

Anciennement considérée comme un art, la cryptographie est désormais reconnue comme une science à part entière **[11].**

Les premières utilisations connues de la cryptographie remontent à l'Antiquité, où la plus ancienne trace de message chiffré a été retrouvée sur une table en argile sur les bords du Tigre en Irak. Au fil des années, les motivations militaires ont conduit les Hommes à développer de nouvelles méthodes de chiffrement plus robustes afin d'éviter que les tactiques ou plans de bataille ne tombent dans les mains de l'ennemi. Les Spartiates ont ainsi inventé le premier dispositif militaire connu : la scytale, ou bâton de Plutarque. La scytale en elle même est un bâton de bois, dont le diamètre est connu uniquement de l'émetteur et du destinataire du message **[11].**

Il a fallu attendre l'époque de Jule César, vers 50 avant J-C, pour voir apparaitre de véritables systèmes cryptographiques. Le plus célèbre d'entre eux est le chiffrement de César, qui consistait simplement à décaler les lettres d'un message de trois positions vers la droite dans l'alphabet latin. Plus tard Blaise de Vigenère (1586) introduit un nouveau chiffrement dans lequel on ne se contente pas d'un seul décalage comme pour César mais de plusieurs. Une section est consacrée à l'étude de ces méthodes un peu plus loin dans ce travail **[11].**

En 1883, Auguste Kerckhoffs énonce un principe fondateur de la cryptographie moderne **[11]:**

« Les mécanismes de chiffrement et de déchiffrement doivent pouvoir être rendus publics, la confidentialité des messages doit être garantie uniquement par le secret d'une clé ».

Le bond technologique suivant survient au XXème siècle lors des deux guerres mondiales. Avec les besoins militaires des différentes armées de protéger leurs communications ont permis de voir l'apparition de machines spécialement conçues pour le chiffrement et le déchiffrement, on peut citer par exemple, Enigma, la C-36, la machine de Lorenz. Dans la deuxième moitié du vingtième siècle, la cryptographie est devenue beaucoup plus mathématique et a été grandement facilité par l'apparition des premiers ordinateurs. Cette cryptographie moderne est initiée par le travail de Claude Shannon en 1948 sur la théorie mathématique de l'information **[11].**

Aujourd'hui, en plus de l'amélioration des méthodes classiques, de nouvelles techniques de chiffrement sont introduites, telles que la cryptographie quantique qui consiste à chiffrer une clé en utilisant des photons envoyés par fibre optique, et toute tentative d'interception de la clé modifie la polarisation des photons. Et la cryptographie chaotique qui se base sur des instabilités de natures inhabituelles des systèmes non linéaires, ce fut alors la découverte des signaux chaotiques, qui ont un comportement déterministe mais qui font penser à des allures pseudo-aléatoires. Le principe de la cryptographie chaotique est alors de noyer le message en clair dans un signal chaotique. Pour le chiffrement et le déchiffrement, on doit alors disposer au niveau de l'émetteur et du récepteur du même signal chaotique pour pouvoir récupérer le message chiffré **[11].**

I .2. Généralités sur la cryptographie :

2.1. Terminologie :

Une certaine confusion règne concernant les différents termes de la cryptographie, à cause en premier lieu de l'utilisation d'anglicismes (termes empruntés à l'anglais), ainsi nous allons définir la terminologie qui va être utilisée tout au long de l'étude afin d'éviter toute ambigüité **[11]:**

Clé: Une clé est un ensemble de paramètres utilisés en entrée d'une opération cryptographique (chiffrement, déchiffrement) **[11].**

Chiffrer ou chiffrement : transformation à l'aide d'une clé de chiffrement d'un message clair en un message chiffré (cryptogramme), incompréhensible par des tiers n'ayant pas la connaissance de la clé (en anglais Encryptions). On utilise aussi le « crypter » **[11].**

Déchiffrer ou déchiffrement : transformation qui consiste à retrouver les informations claires, à partir des informations chiffrées en utilisant la clé de déchiffrement **[11].**

Cryptosystème : Un Cryptosystème est constitué d'un algorithme cryptographique ainsi que toutes les clés possibles et tous les protocoles qui le font fonctionner **[11].**

Ceci dit, nous pouvons à présent donner une définition précise pour :

Texte clair : texte non crypté également appelé texte brut, ou bien le message(en anglais plaintext) **[8].**

Texte chiffré : appelé également cryptogramme, le texte chiffré est le résultat de l'application d'un chiffrement a un texte clair (en anglais ciphertext) **[8].**

Cryptage et décryptage : les données lisibles et compréhensibles sans intervention spécifique sont considérées comme du texte en clair. La méthode permettant de dissimuler du texte en clair en masquant son contenu est appelée le cryptage. Le cryptage consiste à transformer un texte normal en charabia inintelligible appelé texte chiffré.

Cette opération permet de s'assurer que seules les personnes aux quelles les informations sont destinées pourront y accéder. Le processus inverse de transformation du texte chiffré vers le texte d'origine est appelé le décryptage **[4].**

La Figure 1.1 illustre ce processus.

Cryptage **Décryptage**

Texte en clair **texte chiffré** **texte en clair**

Figure 1.1: Cryptage et décryptage [4].

2.2 Cryptologie :

La cryptologie est la science ou l'art des écrits secrets. L'objectif principal est de protéger et défendre le secret et la confidentialité des informations à l'aide d'algorithmes cryptographiques **[1].**

La cryptologie comporte deux branches : la stéganographie (l'écriture couverte), la cryptographie (l'écriture chiffré), cette dernière contient aussi deux grandes branches complémentaires qui sont la cryptographie elle même et la cryptanalyse **[13].**

Le schéma ci-dessous montre les branches de la cryptologie :

:

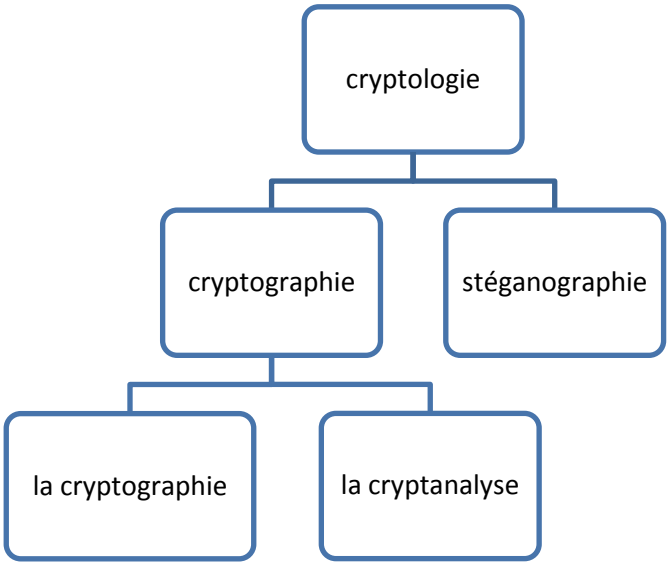


Figure 1.2 : schéma des branches de la cryptologie.

2.3 Stéganographie :

Contrairement à la cryptographie, qui chiffre des messages de manière à les rendre incompréhensibles, la stéganographie (en grec «l'écriture couverte») cache les messages dans un support, par exemple des images ou des textes qui n'a pas d'importance **[13].**

Un peu d'historique [13]:

Les premiers emplois attestés de la stéganographie se lisent chez Hérodote vers le 5ème siècle avant Jésus-Christ: un certain Histiée, voulant prendre contact avec le tyran Aristagoras de Milet, choisit un esclave dévoué, lui rasa la tête, et y inscrivit le message à transmettre.

Il attendit que ses cheveux repoussent pour l'envoyer à Aristagoras avec l'instruction de se faire raser le crâne.

Toujours d'après Hérodote, pour informer les Spartiates de l'attaque imminente des Perses, un certain Démarate utilisa un élégant stratagème: il prit des tablettes, en racla la cire et grava sur le bois le message secret, puis il recouvrit les tablettes de cire. De cette façon, les tablettes, apparemment vierges, n'attirèrent pas l'attention.

Au XVI^{ème}(16) siècle, le scientifique italien Giovanni Porta découvrit comment cacher un message dans un œuf dur: il suffit d'écrire sur la coquille avec une encre contenant une once d'alun pour une pinte de vinaigre; la solution pénètre la coquille et dépose sur la surface du blanc d'œuf, le message que l'on lira aisément après avoir éplucher l'œuf

L'historien de la Grèce Antique Enée le Tacticien imagina d'envoyer un message secret en piquant de minuscules trous sous certaines lettres d'un texte qui n'a pas d'importance. La succession de ces lettres fournit le texte secret.

2.4 Cryptographie :

La cryptographie est un mot grec qui signifie l'écriture chiffré ou caché en lui même par des méthodes qui permettent de transmettre des données de manière confidentielle. Afin de protéger un message, on lui applique une transformation avec une clé qui le rend incompréhensible ; c'est ce qu'on appelle le chiffrement, qui à partir d'un texte en clair, donne un texte chiffré ou cryptogramme. Inversement, le déchiffrement est l'action qui permit de reconstruire le texte en clair à partir de texte chiffré et la clé **[13].**

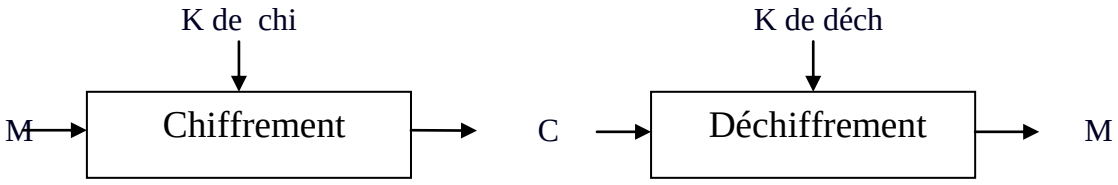


Figure 1.3 : schéma de base de la cryptographie [13].

M: le texte en clair.
C: le texte chiffré ou texte crypté.
K : la clé.
K de chi : la clé de chiffrement.
K de déch : la clé de déchiffrement.

2.4.1 Cryptographie invulnérable :

La vulnérabilité se mesure en termes de temps et de ressources nécessaires pour récupérer le texte en clair.

Une cryptographie invulnérable pourrait être définie comme un texte crypté particulièrement difficile à déchiffrer sans l'aide d'un outil de décodage approprié. Il devrait être impossible de déchiffrer le résultat d'une telle cryptographie avant la fin du monde (même avec un milliard d'ordinateurs effectuant un milliard de vérifications à la seconde).
On pourrait donc penser qu'une cryptographie évoluée résisterait même aux assauts d'un cryptanalyse particulièrement acharné. Personne n'a encore prouvé que le meilleur niveau de cryptage pouvant être obtenu de nos jours tiendra la route avec la puissance informatique de demain. Néanmoins, nous pouvons vous assurer que PGP est actuellement la solution la plus invulnérable à ce jour. La vigilance et la prudence constituent toutefois une meilleure protection que les prétentions d'inviolabilité **[4].**

2.4.2 Cryptographie conventionnelle :

En cryptographie conventionnelle, également appelée cryptage de clé secrète ou de clé symétrique, une seule clé suffit pour le cryptage et le décryptage.
La norme de cryptage de données (DES) est un exemple de système de cryptographie conventionnelle largement utilisé par le gouvernement fédéral des Etats-Unis. La Figure ci-dessus est une illustration du processus de cryptage conventionnel **[4].**

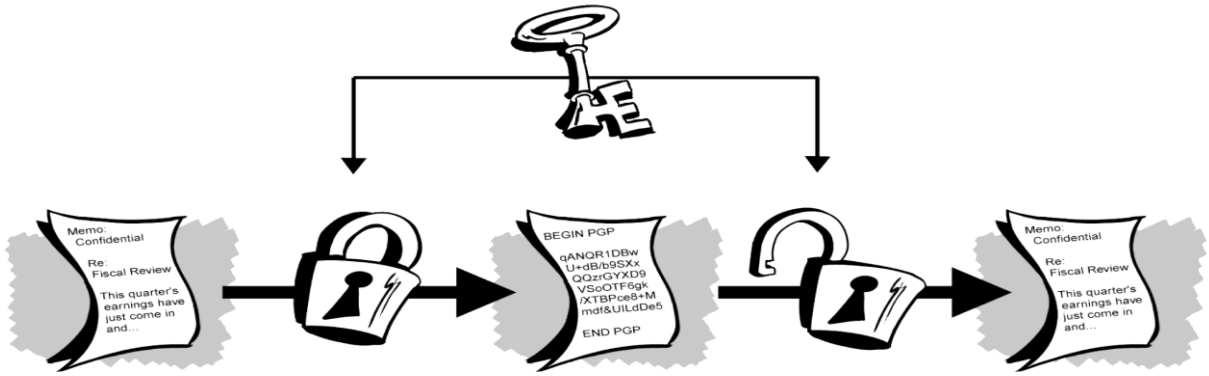


Figure 1.4 : Cryptage conventionnel [4].

I .3. La cryptanalyse :

La cryptanalyse, a l'inverse de la cryptographie, est l'étude des procédés cryptographiques dont le but est de reconstruire les messages en clair correspondant à des messages chiffrés dons on n'est pas destinataire à l'aide de méthodes et techniques mathématiques sans connaître la clef de chiffrement.

Une tentative de cryptanalyse d'un système est appelée une attaque, et elle peut conduire à différentes résultats **[12]:**

- Cassage complet : le cryptanalyste retrouve la clef de déchiffrement.
- Obtention globale : le cryptanalyste trouve un algorithme équivalent à l'algorithme de déchiffrement, mais qui ne nécessite pas la connaissance de la clef de déchiffrement.
- Obtention locale : le cryptanalyste retrouve le texte en clair correspondant à un message chiffré.
- Obtention d'information : le cryptanalyste obtient quelques indications sur le texte en clair ou la clef (certains bits de la clef, un renseignement sur la forme du texte en clair,...).

Une attaque cryptanalytique se caractérise selon des données dont elle dispose, ainsi on peut trouver quatre situations de cryptanalyse :

- Attaque sur texte chiffré seul : disposition seulement d'un nombre fini de textes chiffrés pour retrouver la clef de déchiffrage.
- Attaque à texte clair connu : disposition de couple de message (clairs, chiffrés).
- Attaque à texte clair choisi : l'attaquant a accès à l'algorithme de chiffrement et il l'utilise pour générer le couple (clairs, chiffrés).
- Attaque à texte chiffré choisi : Le cryptanalyste peut choisir les textes à déchiffrer sans connaître la clef **[12].**

3.1 Cryptanalyse différentielle :

La cryptanalyse différentielle est un type d'attaques qui peut-être utilisé contre les algorithmes de chiffrement par blocs itératifs. Elle utilise une attaque à texte en clair choisi et se base sur l'observation de l'évolution des différences entre deux textes lorsqu'ils sont chiffrés avec la même clef. En analysant ces différences entre paires de textes, il est possible d'attribuer des probabilités à chaque clef possible.

A force d'analyser des paires de textes, on finit soit par trouver la clef recherchée, soit par réduire suffisamment le nombre de clefs possibles pour pouvoir mener une attaque exhaustive rapide **[12].**

3.2 Cryptanalyse linéaire :

La cryptanalyse linéaire utilise une attaque à texte en clair connu et consiste à modéliser l'algorithme de chiffrement par blocs par une approximation linéaire. Avec un nombre suffisant de paires (texte en clair, texte chiffré), on peut deviner certains bits de la clef **[12].**

I.4. Mécanismes de la cryptographie :

Un algorithme de cryptographie ou un chiffrement est une fonction mathématique utilisée lors du processus de cryptage et de décryptage. Cet algorithme est associé à une clé (un mot, un nombre ou une phrase), afin de crypter le texte en clair. Avec des clés différentes, le résultat du cryptage variera également.

La sécurité des données cryptées repose entièrement sur deux éléments :

- L'invulnérabilité de l'algorithme de cryptographie**.**
- La confidentialité de la clé.

Un système de cryptographie est constitué d'un algorithme de cryptographie, ainsi que de toutes les clés et tous les protocoles nécessaires à son fonctionnement. PGP est un système de cryptographie **[4].**

I.5. Objectifs de sécurité de la cryptographie :

La cryptographie permet de rendre un certain nombre de services **[11]:**

- **La confidentialité :** Est l'assurance qu'un document ne sera pas lu par une personne qui n'en a pas le droit lors de la transmission de ce document ou lorsqu'il est archivé. La confidentialité représente le service le plus important en cryptographie.
- **L'intégrité :** d'un (message, document, fichier…) est la garantie que cet objet n'a pas été modifié par une autre personne que son auteur. Pour certaines applications, L'intégrité d'une donnée est au moins aussi importante que sa confidentialité. Par exemple, lors d'une transaction bancaire, il est indispensable que la somme d'argent mise en jeu ne soit pas modifiée.
- **L'authentification :** Elle est l'assurance de l'identité d'un objet, généralement une personne, mais cela peut aussi s'appliquer à un serveur, une application,…. Dans la vie courante, la présentation d'une carte d'identité et la signature manuelle assurent un service d'authentification.
- **Non répudiation :** Le but de ce service est que l'émetteur d'un message ne puisse pas nier l'avoir envoyé et le récepteur l'avoir reçu. En cryptographie ce service est assuré par des certificats.

I.6. Système de cryptographie :

Un bon système de cryptographie c'est un système qui permet de rendre un message indéchiffrable par toute personne à laquelle il n'est pas destiné.

Dans la mesure du possible, ce système doit permettre de joindre une signature inviolable au message pour permettre au destinataire de contrôler son authenticité **[13].**

6.1. La structure générale d'un système cryptographique (cryptosystème) ou système de chiffrement :

De façon plus formelle, dans un système cryptographique, on distingue souvent:

- l'espace des messages clairs $\boldsymbol{M}$ (plaintext) sur un alphabet $\boldsymbol{A}$.
- l'espace des messages chiffrés $\boldsymbol{C}$ (ciphertext) sur un alphabet $\boldsymbol{B}$.
- l'espace des clés $\boldsymbol{K}$ (Key).
- un ensemble $\boldsymbol{E}$ de transformations de chiffrement utilisant une clé :

$$E_k : m \in \boldsymbol{M} \rightarrow c \in \boldsymbol{C}$$

- un ensemble $\boldsymbol{D}$ de transformations de déchiffrement utilisant une clé :

$$D_k : c \in \boldsymbol{C} \rightarrow m \in \boldsymbol{M}$$

Evidemment, la condition $D_{k_d}(E_{k_c}(m)) = m$ doit être réalisée et l'ensemble $(\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{E}, \boldsymbol{D})$ va être appelé un cryptosystème **[13].**

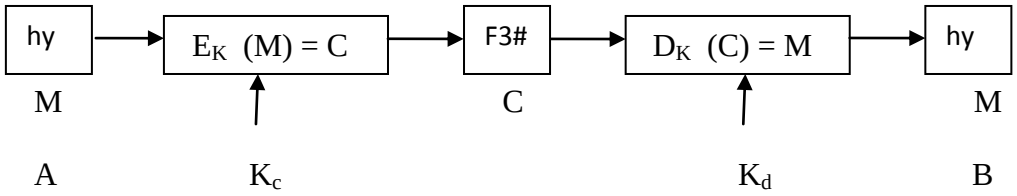


Figure 1.5: schéma d'un cryptosystème simple [13].

Conclusion :

Dans ce chapitre, nous avons présenté une introduction sur la cryptographie en général, son historique ainsi que les différents objectifs y afférents.

Dans le chapitre suivant, nous abordons les techniques classiques de la cryptographie.

Chapitre II

Cryptographie Classique

Introduction :

Depuis des temps très reculés, l'homme avait utilisé diverses méthodes et techniques pour envoyer un message secrètement. Ce sont des méthodes qui transforment le message en clair en message incompréhensible ou qui cachent le message par une image, un texte ou autres choses sans qu'une personne étrangère puisse s'en apercevoir.

II.1.La cryptographie classique :

La cryptographie classique concerne la période de l'antiquité jusqu'à l'apparition des ordinateurs, elle traite les systèmes reposant sur les lettres, symboles et caractères des langues naturelles (allemand, anglais, français, ...) **.**Et inclut tous les mécanismes et algorithmes basés sur des fonctions mathématiques ou logiques (exemple: César, Vigner1) **[17].**

- **Systèmes Manuels :**

Par système manuel, on entend habituellement les méthodes cryptographiques qui requièrent simplement du papier et un crayon. Il n'est pas surprenant que ces systèmes (chiffrement classique) soient les plus anciens. Parmi ces systèmes on trouve celui de César, chiffre affine, Vigenère, Playfair, Hill et bien d'autres **[13].**

II.2.les méthodes de la cryptographie :

On à deux (2) méthodes essentiels de la cryptographie classiques sont la cryptographie par transposition et la cryptographie par substitution.

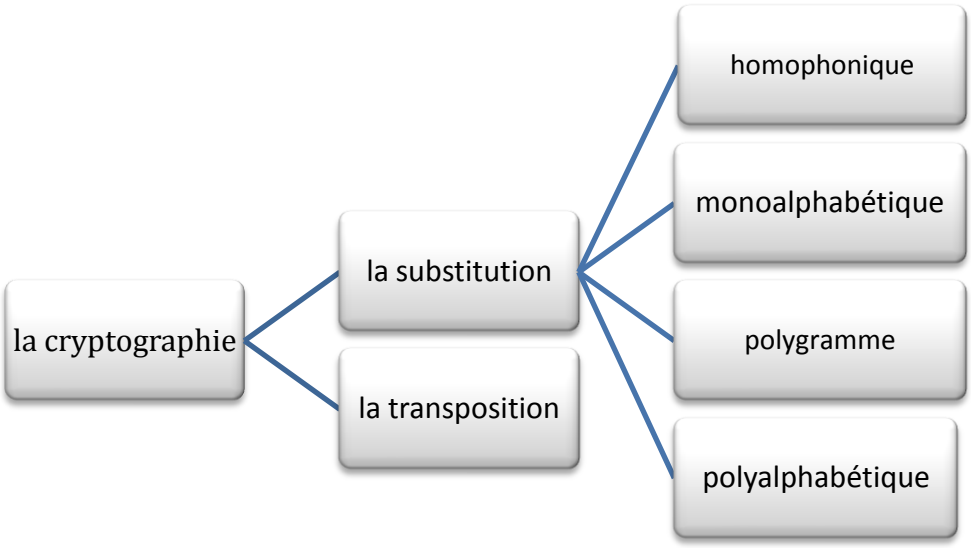


Figure II.1 : les méthodes de la cryptographie classique [13].

2.1. La cryptographie par transposition :

Le Chiffrement par transposition consiste à appliquer une permutation des caractères sur le message clair en entier. De ce fait, le message chiffré est fait du même matériel que le message clair.

Ce type de chiffrement est de nouveau sensible à des attaques par analyse de fréquence de mots. Ces chiffrements sont en général cassables par les moyens de calculs actuels **[9].**

C'est un système simple, mais peu sûre pour de très brefs messages car il y a peu de variantes. Comme un mot de trois lettres ne pourra être transposé que 6 (= 3!) Positions différentes. Par exemple, "nom" ne peut se transformer qu'en "nom", "nmo", "onm", "omn", "mno" et "mon".

Lorsque le nombre de lettres croît, il devient de plus en plus difficile de retrouver le texte original sans connaître la méthode de permutation ou changement. Comme une phrase de 35 lettres peut être disposée de 35! = 1040 manières différentes.

Ce chiffrement utilise le concept mathématique des permutations.

Il y'a plusieurs types de transpositions, parmi les on citera:

- La première méthode et outil utilisé de cryptographie par transposition et très ancien est le scytale de sparte (404 avant J.C) est un objet rond de bois calibré autour duquel il entoura la ceinture (message chiffré) **[13].**

2.1.1. La transposition simple par colonne :

Elle consiste de permuter (changer) les lettres de message clair comme :

Le message suivant =**"" cryptographie "" .**

La clé = **« une permutation des lettres ».**

message	**c**	**R**	**y**	**p**	**t**	**o**	**G**	**R**	**a**	**p**	**h**	**I**	**E**
clé	**4**	**1**	**7**	**5**	**2**	**10**	**12**	**3**	**8**	**11**	**13**	**6**	**9**

Deviant:

message	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
clé	**r**	**T**	**r**	**c**	**p**	**i**	**Y**	**A**	**e**	**o**	**p**	**G**	**H**

Tableau II.1: Transposition simple par colonne

2.1.2 La transposition rectangulaire :

La transposition rectangulaire consiste à écrire le message dans une grille rectangulaire, en utilisant un mot clé secret donné (uniquement avec des caractères différents) et en dérivant une séquence de chiffres à arranger (mélanger) de ce mot clé, on chiffre selon le mot clé secret à arrangé, on écrit le texte par ligne et on le lit par colonne comme indique l'exemple suivant **[2]:**

On a choisi comme mot clé secret**" master"** pour chiffrer le message précédent :

" LE SYSTEME INFORMATIQUE ".

m	a	s	t	e	r
2	5	4	6	1	3
L	E	S	Y	S	T
E	M	E	I	N	F
O	R	M	A	T	I
Q	U	E	a	b	c

⟹

E	m	r	s	a	T
1	2	3	4	5	6
S	L	T	S	E	Y
N	E	F	E	M	I
T	O	I	M	R	A
b	Q	c	E	U	a

Tableau II.2: application d'une transposition rectangulaire

En remplissant la grille, on constate qu'il reste trois cases vides, que l'on peut remplir avec des nulles (ou pas, selon les désirs des correspondants).

Après l'arrangement des colonnes de cette grille, on obtient le message crypté suivant

""SLTSEYNEFEMITOIMRAbQcEUa"".

2.2. La cryptographie par substitution :

Un chiffrement par substitution : est un algorithme par lequel chaque caractère du message clair (écrit dans un alphabet donné) est substitué par un autre caractère dans le message chiffré (qui peut être écrit dans un alphabet différent de celui du message clair) **[9].**

En cryptographie classique il ya quatre types de chiffrement par substitution sont distingués :

- Substitution simple.
- Substitution homophonique.
- Substitution polygramique.
- Substitution poly-alphabétique.

2.2.1. La substitution mono-alphabétique ou simple :

Substitution simple : Un caractère du message clair est substitué par un caractère unique du message chiffré.

Cela correspond le plus souvent à une permutation des caractères de l'alphabet des messages clairs **[9].**

Les faiblesses de cette méthode **[13]:**

- L'analyse de fréquence d'apparition des lettres.
- La longueur de clé est égale au nombre des lettres alphabétiques.

Exemple : Un message codé par la substitution mono-alphabétique:

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

Tableau II.3: le chiffrement aléatoire.

Le message en clair "telecommunication" sera chiffré par <<FMZMLDEENUTLSFTDU >>

❖ **Le chiffrement d'Atbash [13]:**

Le chiffrement d'Atbash (-500) consiste simplement à écrire l'ordre de l'alphabet à l'envers, c'est une méthode facile à casser parce qu'il n'y a pas d'autre choix possible pour la clé.

Le message en clair : « telecommunication ».

La clé : alphabet à l'envers z = a…….a = z .

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

Tableau II.4: le chiffrement d'Atbash.

Le message chiffré : <<GVOVXLNNFMRXZGRLM >>.

- **le carré de Polybe :**

Le carré polybique est une méthode de chiffrement à été créée par Polybe qui est un historien grecque environ de – 205 AJC jusque -125 AJC.

Ce Polybe est placé les lettres alphabétiques dans un tableau 6*6 numéroté en ligne et en colonne, la lettre i et j sont dans la même cage comme indique le tableau suivant :

	1	2	3	4	5
1	A	b	c	d	e
2	F	g	h	I, j	k
3	L	m	n	o	p
4	Q	r	s	t	u
5	V	w	x	y	z

Tableau II.5: le carré de Polybe.

Les coordonnées des lignes et des colonnes correspondant aux lettres du texte à chiffrer sont utilisés pour transcrire le message en chiffres. Avec ce procédé chaque lettre du texte en clair est représentée par deux chiffres écrits verticalement. Ces deux coordonnées sont ensuite transposées en les recombinants par deux sur la ligne ainsi obtenue **[15].**

Exemple :

on a chiffré le message :« telecommunication ».

La clé : le carré polybique.

Le message chiffré : << 44153115133432324533241311442434 33 >>

- **Chiffrement de césar ou par décalage : [13]**

Jules César (100-44 avant J.-C.) : est un générale et homme politique de Rome pendant la guerre des gaules, jules césar envoyait des messages chiffrés à Rome, il employait une substitution simple avec l'alphabet normal, il s'agissait simplement de décaler les lettres de l'alphabet d'une quantité fixe, dans les communications du gouvernement.

Ce chiffre était moins robuste qu'Atbash, mais à une époque où très peu de personnes savaient lire, cela suffisait. César écrivait aussi parfois en remplaçant les lettres latines par les lettres grecques.

Il s'agit d'un des plus simples chiffres classiques les plus populaires. Son principe est un décalage des lettres de l'alphabet. Dans les formules ci-dessous,

p est l'indice de la lettre de l'alphabet, k est le décalage.

Pour le chiffrement, on aura la formule:

$$C = E(p) = (p + k) \bmod 26.$$

Pour le déchiffrement, il viendra:

$$p = D(C) = (C - k) \bmod 26.$$

Si on connait l'algorithme utilisé, la cryptanalyse par force brute est très facile. En effet, dans le chiffre de César, seules 25 (!) clés sont possibles.

Le chiffrement utilisé par César avec un décalage de k= 5.

clair	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
code	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	

Tableau II.6: le chiffrement de césar.

Exemple de chiffrement de césar :

c = p + k mod 26.

On a le message clair : << telecommunication >>.

Clé : décalage de 5.

Le message chiffré : <<YJQJHTRRZSNHFYNTS>>.

Exemple de déchiffrement :

m = c - k mod26.

Le message chiffré : << RFXYJW >>.

Clé : décalage de 5.

Le message clair : << master >>.

Toutes les substitutions simples sont vulnérables à une analyse des fréquences **(**apparition des lettres).

- **Analyse de fréquences [13]:**

Abu Yusuf Al Kindi publie le premier ouvrage connu de la cryptanalyse, Manuscrit sur le déchiffrement des messages cryptographiques, qui présente l'analyse des fréquences au IX^{ème} (9) siècle.

Lorsque la langue de départ et la technique de chiffrement sont connues, on peut exploiter les régularités du langage par le principe d'analyse de la fréquence d'une lettre. Cette technique ne fonctionne bien que si le message chiffré est suffisamment long pour avoir des moyennes significatives.

Dans la langue française chaque lettre n'apparait pas avec la même fréquence.

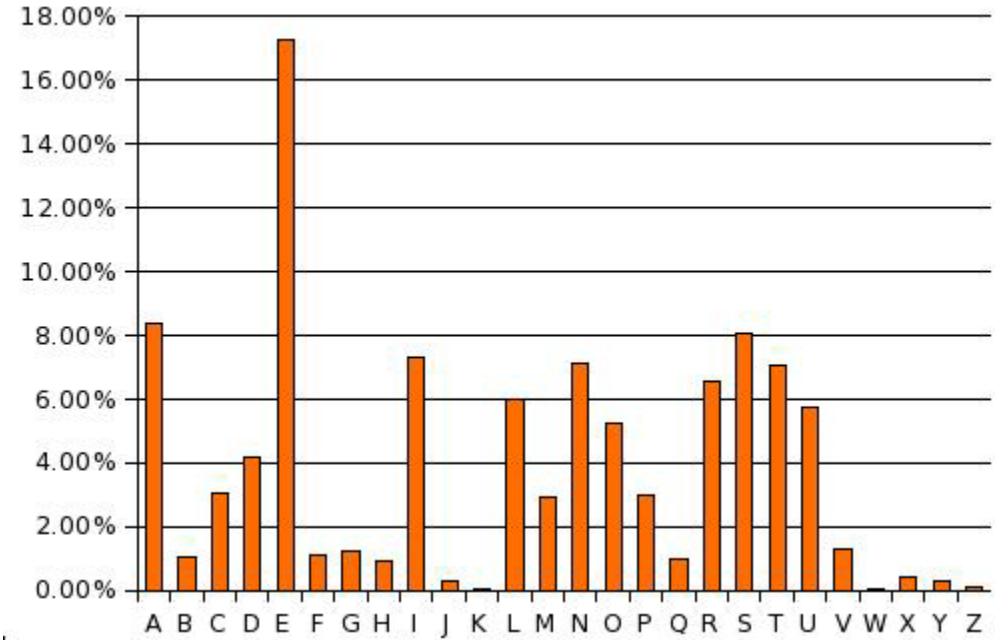


Figure II.2: l'apparition des lettres de français [13].

On peut classifier les lettres en différents groupes :

[17%] → e.

[6% - 8%] → a; i ; l ; n; r ; s; t; u; o.

[3% - 4%] → c; d; m; p.

[0% -1%] → b; f ; g; h; j ; k; q; v; w; x; y; z.

Cependant, il existe également des cas où cette analyse ne fonctionne pas, comme De Zanzibar et la Zambie.

Pour éviter ce type d'attaque sur un texte chiffré, il existe différents moyens:

– On peut par exemple chiffrer le message par digrammes (2 lettres à la fois) comme "au", "en", "ch", "ph", "te" et "ai," ou bien trigrammes (3 lettres à la fois) comme "ant", " eau", "ent", "eur", "pha" et "tion".

Problèmes :

- ✓ L'espace des clefs est trop petit: 26 clefs. Un attaquant peut facilement essayer toutes les clefs.
- ✓ Le chiffrement de César ne cache pas la structure statistique du texte clair.

  Tous les "e" sont transformes en "h".

2.2.2. La substitution poly-alphabétique [15]:

Le principe consiste à remplacer chaque lettre du message en clair par une nouvelle lettre prise dans un ou plusieurs alphabets aléatoires associés. Par exemple, on pourra utiliser n substitutions mono alphabétiques ; celle qui est utilisée dépend de la position du caractère à chiffrer dans le texte en clair. On choisit une clé qui sert d'entrée dans la grille poly alphabétique incluant autant de symboles qu'il y a de lettres différentes à chiffrer. Chaque caractère de la clé désigne une lettre particulière dans la grille de codage. Pour coder un caractère, on doit lire le caractère correspondant du texte en clair en utilisant la grille poly alphabétique et le mot clé associé dans l'ordre séquentiel (on répète la clé si la longueur de celle-ci est inférieure à celle du texte de départ). L'exemple le plus célèbre est l'algorithme de VIGENERE. L'illustration la plus simple qui correspond à ce principe est l'utilisation d'une fonction à base de ou exclusif (XOR).

- ❖ **chiffrement de Vigenère (1568) [13]:**

Blaise de Vigenère, diplomate français publia en 1586 une version simplifiée de l'idée d'Alberti. Le chiffre de Vigenère utilise des substitutions alphabétiques multiples par décalage.

Il choisit un mot comme clé.

Le rang de chaque lettre de la clé définit un décalage à appliquer, Il proposait simplement de n'utiliser que des substitutions par décalage.

Cela avait le mérite de simplifier la méthode d'Alberti (les clefs sont plus simples).

On met le message clair et la clé :

$M = (l_1, l_2, \ldots\ldots\ldots\ldots l_n)$. $K = (k_1, k_2, \ldots\ldots\ldots\ldots k_m)$.

Sous forme numérique avec la correspondance.

(a = 0; b = 1; c = 2;..... z = 25).

On ajoute la clé à le message clair lettre à lettre modulo 26 de façon répétitive.

$$\begin{array}{cccccccccc} & m_1 & m_2 & \cdots & m_\ell & m_{\ell+1} & \cdots & m_{2\ell} & m_{2\ell+1} & \cdots \\ + & k_1 & k_2 & \cdots & k_\ell & k_1 & \cdots & k_\ell & k_1 & \cdots \\ \hline & c_1 & c_2 & \cdots & c_\ell & c_{\ell+1} & \cdots & c_{2\ell} & c_{2\ell+1} & \cdots \end{array}$$

Exemple de chiffrement de Vigenère :

On chiffre le message clair "" cryptage"" avec le mot clé "" master "".

Message Clair :	c	r	y	p	t	a	g	e
	3	18	25	16	20	1	7	5
La clé :	m	a	s	t	e	r	m	a
	13	1	19	20	5	18	13	1
Addition:								
	16	19	44	36	25	19	20	6
addmod26:	16	19	18	10	25	19	20	6
Message chiffré:	p	s	r	j	y	s	t	f

Le message chiffré est: << PSRJYSTF >>.

Exemple pour le déchiffrement on fait l'inverse (soustraction) :

On a le message chiffré "" NLAJJYFPISKW "" et la clé est ""chiffre "" .

Message chiffré:	N	L	A	J	J	Y	F	P	I	S	K	W
	13	11	26	9	35	24	5	15	8	18	36	22
Cle:	c	h	i	f	r	e	c	h	i	f	r	e
	2	7	8	5	17	4	2	7	8	5	17	4
Soustraction:												
	11	4	18	4	19	20	3	8	0	13	19	18
Message clair:	l	e	s	e	t	u	d	i	a	n	t	s

Le message clair :" les etudiants ".

On a dans le message chiffré la lettre" J" est répétée deux fois, sont" e" et" t" dans le message clair (amélioration de chiffre de césar).

l'interét du chiffrement de Vigenère est que la même lettre sera chiffrée et déchiffrer de différentes manières ce qui rend inutilisable l'analyse de fréquence classique.

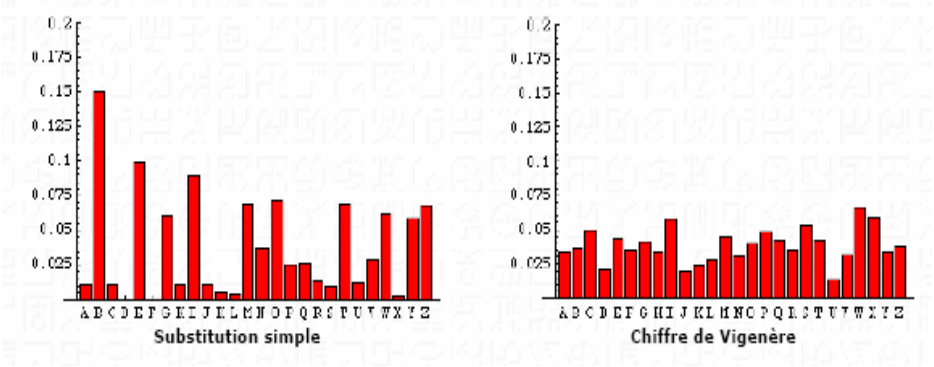


Figure II.3: fréquences des lettres d'une fable de la Fontaine (le chat, la belette et le lapin) chiffrée avec une substitution simple (gauche) et avec le chiffre de Vigenère (droite) [13].

- **Le carré de Vigenère [13]:**

Il présente un tableau du type Trithème, que l'on dénomme aujourd'hui le carré de Vigenère. C'est une amélioration décisive du chiffre de César, Sa force réside dans l'utilisation non pas d'un, mais de 26 alphabets décalés pour chiffrer un message.

On considéra longtemps ce chiffre comme in décryptable, légende si tenace que même en 1917, plus de cinquante après avoir été cassé, le carré de Vigenère était donné pour «impossible à décrypter » par la très sérieuse revue Scientific American.

	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

Tableau II.7: Carré de Vigenère [13].

La lettre de la clé est dans la colonne la plus à gauche du couleur rose et la lettre du message clair est dans la ligne tout en haut du couleur verte. La lettre chiffrée est à l'intersection des deux lettres de ligne et colonne du couleur jeune.

Un autre exemple de chiffrement de vigenère en utilisant le carré de vigenère: chiffrer le texte " cryptage " avec le mot clé " master ".
(Cette clé est éventuellement répétée plusieurs fois pour être aussi longue que le texte clair).

clé	M	A	s	t	e	R	m	a
texte	C	R	y	p	t	a	g	e
chiffré	O	R	Q	I	X	R	S	E

Tableau II.8: application de carré de Vigenère.

Le message chiffré est : << ORQIXRSE >>.

2.2.3 La substitution homophonique [13]:

Son principe consiste *à* remplacer une lettre de message clair non pas par un symbole unique, mais par un symbole choisi au hasard parmi plusieurs, cela empêche la mise en

correspondance des lettres les plus fréquentes avec les symboles les plus représentés dans le texte chiffré.

Exemple :

On chiffre la lettre "b" par 01, 34, 19 ou 89.

"o" par 58, 62,98 ou 37.

"n" par 48, 57,38 ou 10.

Le mot en clair " bon " sera chiffré au hasard comme suit : « 89 62 38».

2.2.4 La substitution polygrammique ou polygraphique [3]:

Dans les substitutions polygrammiques (aussi appelées polygraphiques), les lettres ne sont pas chiffrées séparément, mais par groupes de plusieurs lettres (deux ou trois généralement).

Parmi les substitutions polygrammique, on trouve le chiffre de Playfair (1854) (avec $n = 2$), le chiffre de Hill (1929).

❖ **Le chiffrement de HILL, lester.s.hill (1929) [3]:**

Le chiffre de Hill (Cryptosystème poly-alphabétique.) : C'est une méthode de chiffrement qui utilise des matrices carrées. C'est une manière attrayante pour la familiarisation au calcul matriciel, au calcul modulo *n* et à la notion d'algorithme.

Le chiffre de Hill a été publié par *Lester S. Hill* en 1929 : c'est un chiffre polygraphique, c'est-à-dire qu'on ne (dé)chiffre pas les lettres les unes après les autres, mais par paquets.

Les lettres sont d'abord remplacées par leur rang dans l'alphabet :

a	b	c	d	e	f	g	h	I	J	k	l	M
1	2	3	4	5	6	7	8	9	10	11	12	13
n	o	p	q	r	s	t	u	V	W	x	y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Tableau II.9: les rangs de l'alphabet.

Les lettres P_k et P_{k+1} du texte clair seront chiffrées C_k et C_{k+1} avec la formule ci-dessous :

$$\begin{pmatrix} Ck \\ Ck+1 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} Pk \\ Pk+1 \end{pmatrix} \bmod 26$$

Ce qui signifie, pour fixer les idées, que les deux premières lettres du message clair (P_1 et P_2) seront chiffrées (C_1 et C_2) selon les deux équations suivantes :

$$C_1 = a \cdot P_1 + b \cdot P_2 \pmod{26}.$$
$$C_2 = c \cdot P_1 + d \cdot P_2 \pmod{26}.$$

Exemple de chiffrement :

On prend comme clef de chiffrement la matrice $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$ pour chiffrer le message.

«Crypte».

D'après le tableau précédant : P_1 = « c » = 3 et P_2 = «r» = 18. Les deux premières lettres du message seront donc cryptées ainsi :

$$C_1 = 3 \cdot 9 + 18 \cdot 4 \pmod{26} = 99 \pmod{26} = 21. \quad C_2 = 3 \cdot 5 + 18 \cdot 7 \pmod{26} = 141 \pmod{26} = 11.$$

P_3= «y» = 25 et P_4 = « p » = 16 seront :

$$C_3 = 9 \cdot 25 + 4 \cdot 16 \pmod{26} = 289 \pmod{26} = 3.$$
$$C_4 = 5 \cdot 25 + 7 \cdot 16 \pmod{26} = 237 \pmod{26} = 3.$$

P_5= «t » = 20 et P_6 = « e » = 5 seront :

$$C_5 = 9 \cdot 20 + 4 \cdot 5 \pmod{26} = 200 \pmod{26} = 18.$$
$$C_6 = 5 \cdot 20 + 7 \cdot 5 \pmod{26} = 135 \pmod{26} = 5.$$

En procédant de même avec les paires de lettres suivantes, elle obtiendra finalement :

Lettres	c	r	Y	p	T	E
Rangs (P_k)	3	18	25	16	20	5
Rang chiffré(C_k)	21	11	3	3	18	5
Lettres chiffrées	u	k	C	c	R	E

Tableau II.10: le chiffrement de HILL.

Le message chiffré sera donc ««ukccr e»» (on a l'habitude d'écrire le message chiffré en lettres majuscules groupées par cinq).

- **Remarque :** si le nombre de lettres du message clair avait été impair, on aurait simplement ajouté une lettre arbitraire à la fin du message original.
- **Remarques sur la matrice de chiffrement :**

On ne peut pas prendre n'importe quoi comme matrice de chiffrement, Ses composantes doivent tout d'abord être des nombres entiers positifs. Il faut aussi qu'elle ait une matrice inverse dans Z26 **[3].**

Cette matrice inverse existe si $(ad–bc)^{-1}$ (mod 26) existe, ce qui est le cas quand $(ad–bc)$ et 26 sont premiers entre eux.

Il faut donc contrôler que $(ad–bc)$ est impair et n'est pas multiple de 13.

Déchiffrement :

Pour déchiffrer, le principe est le même que pour le chiffrement : on prend les lettres deux par deux, puis on les multiplie par la matrice de déchiffrement.

$$\begin{pmatrix} P_k \\ P_{k+1} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} \begin{pmatrix} C_k \\ C_{k+1} \end{pmatrix} \pmod{26}$$

Ordinairement, l'inverse de la matrice:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ est : } \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad-bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Exemple de déchiffrement :

Pour déchiffrer le message «ukccr e», on doit calculer :

$$\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}^{-1} \pmod{26} = \frac{1}{9\cdot 7 - 4\cdot 5} \begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} \pmod{26} = 43^{-1} \begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} \pmod{26} = \ldots ?$$

Le problème est maintenant de calculer l'inverse de 43 modulos 26.

Il existe des algorithmes efficaces pour déterminer l'inverse de k (mod n), par exemple l'algorithme d'Euclide étendu.

Mais quand n = 26, la méthode force brute est sans doute la manière la plus simple :

- Algorithme pour trouver k^{-1} modulo 26 (force brute).

1. Multiplier successivement k par les entiers m de l'ensemble {1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23,25}.
2. Stopper quand le produit $\boldsymbol{k.m}$ est égal à 1 (mod 26) ; $\boldsymbol{k^{-1}}$ modulo 26 = m.

L'utilisation de cet algorithme nous donne que 43^{-1} (mod 26) = 23.

On peut maintenant terminer de calculer la matrice de déchiffrement :

$$\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}^{-1} (\text{mod } 26) = 23 \begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} (\text{mod } 26) = \begin{pmatrix} 161 & -92 \\ -115 & 207 \end{pmatrix} (\text{mod } 26) = \begin{pmatrix} 5 & 12 \\ 15 & 25 \end{pmatrix}$$

On va donc utiliser la matrice $\begin{pmatrix} 5 & 12 \\ 15 & 25 \end{pmatrix}$ pour déchiffrer le message.

«ukccr e». Après avoir remplacé les lettres par leur rang, il calculera :

$$P_1 = 5 \cdot 21 + 12 \cdot 11 \text{ (mod 26)} = 237 \text{ (mod 26)} = 3$$

$$P_2 = 15 \cdot 21 + 25 \cdot 11 \text{ (mod 26)} = 590 \text{ (mod 26)} = 18$$

En procédant de même avec les paires de lettres suivantes, on obtiendra finalement :

Lettres chiffrées	u	k	c	c	r	e
Rangs chiffré(C_k)	21	11	3	3	18	5
Rangs (P_k)	3	18	25	16	20	5
Lettres	c	r	y	p	t	e

Tableau II.11: le déchiffrement de HILL.

Le message déchiffré sera donc bien « crypte ».

II.3.Systèmes Mécaniques :

Les systèmes manuels sont souvent lents et laborieux pour l'usager puisque fondés sur l'emploi de papier et crayons. De plus, ils ne permettent pas l'utilisation d'algorithmes compliqués. Des méthodes de chiffrement plus rigoureuses et complexes utilisant des appareils mécaniques ont donc été mises au point. La section qui suit offre un bref aperçu des plus célèbres de ces machines à chiffrer mécaniques **[13].**

- **le masque jetable (Chiffrement de Vernam) :**

en 1917, Gilbert S. Vernam, a inventé une machine de chiffre poly alphabétique pratique capable d'employer une clé qui est totalement aléatoire et ne se répète jamais - un masque jetable.

C'est seul le chiffre, dans nos connaissances actuelles, dont on a prouvé qu'il était in décryptable en pratique et en théorie. Ce procédé ne fut cependant jamais utilisé par l'armée car il exigeait de devoir produire des millions de clés différentes (une par message), ce qui est impraticable. Par contre, il fut utilisé par les diplomates allemands dès 1921. Le chiffrement de Vernam (masque jetable) est un algorithme de cryptographie en décalage circulaire. Bien que simple facile et rapide tant pour le codage que pour le décodage, il est inconditionnellement sûr. Cependant, il présente d'importantes difficultés de mise en œuvre pratique. Le chiffrement par la méthode du masque jetable consiste à combiner le message en clair avec une clé présentant les caractéristiques très particulières suivantes :

- La clé doit être une suite de caractères au moins aussi longue que le message à chiffrer.
- Les caractères composant la clé doivent être choisis de façon totalement aléatoire.
- Chaque clé, ou masque, ne doit être utilisée qu'une seule fois.

Dans la pratique, la première difficulté que présente ce système est la longueur et le nombre des clés nécessaires. On a de plus, le problème de leur transmission au correspondant, de leur stockage. On travaille en effet souvent avec plusieurs correspondants, ayant chacun plusieurs jeux de clés en commun.

Ensuite générer des clés réellement aléatoires nécessite des moyens complexes, et enfin, garantir l'utilisation unique de chaque clé, pose des problèmes d'organisation importants. A défaut, la sécurité du système peut être compromise **[13]:**

Un exemple de chiffrement de Vernam ou masque jetable, utilise addition mod 26:

Le message en clair :" infotronique".

La clé : génération des lettres aléatoirement " n k t e a b p x d z m y".

Le message chiffré est : << V Y Z T U T E L M Q H D>>.

- **L'Appareils à disques codeurs :**

Premier appareil à disque codeur fut inventé par Léon Battista Alberti au 15^{e} siècle. Il était formé de deux disques concentriques en cuivre, l'un de ces disques étant grand et fixe, et l'autre plus petit et mobile. Ces disques étaient divisés en 24 parties radiales égales. Le disque extérieur contenait un nombre suffisant de lettres de l'alphabet

pour former la majorité des mots latins. De plus, le disque intérieur contenait une permutation de l'alphabet latin. Ce mécanisme, quoique illustre l'ingéniosité d'Alberti, qui combine pour la première fois une substitution dite poly alphabétique et l'usage d'un code **[13].**

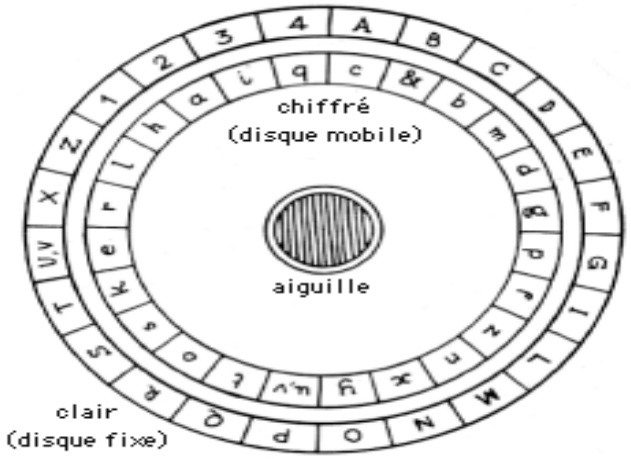


Figure II.4 : Cadran d'Alberti [13].

- **Le cylindre à roues codeuses M-94/CSP-488 :**

En 1922, l'armée américaine fit fabriquer le M-94, un appareil cylindrique comportant 25 anneaux (de diamètre d'environ 4 cm) en aluminium sur un pivot d'environ 10,5 cm de long. Ce mécanisme demeura en service jusqu'au début de la Deuxième Guerre mondiale. Il fut aussi utilisé par la garde côtière et la « Federal Communications Commission » des États-Unis. La marine américaine avait une version semblable dans la CSP-488 M-94 **[13].**

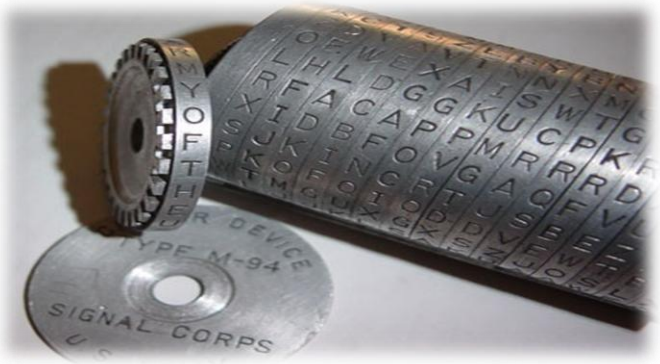


Figure II.5: Roues codeuses M94 [13].

- **Le convertisseur M-209 de Hagelin:**

Fabriqué par Boris Hagelin au début des années 1940 pour l'armée américaine, le M-209 était un appareil mécanique simple mesurant 18 cm de large par 14 cm de profond et 9 cm de haut avec un poids d'environ 4 kg. Ses composantes principales incluaient six roues codeuses à 26, 25, 23, 21, 19 et 17 positions respectivement, assurant une longueur de cycle (c'est-à-dire le nombre d'étapes avant qu'une clé donnée se répète) de 101 405 850 pas. L'appareil imprime la lettre chiffrée sur une bande-papier. Pendant la Deuxième Guerre mondiale, plus de 140 000 unités de cet appareil furent fabriquées aux États-Unis **[13].**

Figure II.6: M-209 de Hagelin [13].

- **Systèmes électromécaniques:**

Des machines à chiffrement assez complexes et efficaces fonctionnant notamment avec des piles comme source d'énergie furent mises au point au 20^{e} siècle. Plusieurs furent utilisées pendant la Deuxième Guerre mondiale, dont la plus connue fut l'Enigma.

Ayant pour origine des brevets de 1919 d'Alexandre Koch et des brevets d'Arthur Scherbius durant les années 1920, l'Enigma fut produite en version commerciale en 3 modèles : A, B et C. Le modèle C était un appareil qui n'avait pas d'imprimante, mais plutôt un mécanisme qui illuminait les lettres chiffrées. Tous les modèles possédaient un clavier. Pendant la Deuxième Guerre mondiale, l'Enigma servit l'armée (Armee), la marine (Kriegsmarine), et les forces aériennes (Luftwaffe) allemandes **[13].**

- **L'Enigma :**

Différait par deux aspects importants des machines à roues codeuses précédentes. La dernière roue codeuse entrait en contact avec un tambour final quiavait 26 boutons de contact sur une seule face, ces boutons étant reliés l'un à l'autre par 13 fils internes. Ceci rendait le chiffrement réciproque, c'est-à-dire que $E(E(x)) = x$ avec la condition que $E(x) \neq x$ pour tout caractère x de l'alphabet. Ce chiffrement réciproque permettait le déchiffrement avec la même configuration initiale que le chiffrement. L'autre aspect important était la présence d'engrenages qui assuraient un mouvement irrégulier des roues codeuses.

Un désavantage de l'Enigma était qu'elle n'imprimait pas les résultats, et donc que son utilisation nécessitait trois personnes : une pour lire le texte reçu et manipuler le clavier, une pour énoncer d'une voix forte chaque lettre illuminée et une autre pour rédiger le texte **[13].**

Figure II.7: la machine Enigma [13].

Conclusion :

Ce chapitre présente les différentes techniques classiques telles que celles de César, Vigenère, Hill, Enigma etc.….

Dans le chapitre suivant nous procédons à l'étude des techniques de chiffrement moderne symétrique et asymétrique.

Chapitre III

Cryptographie Moderne

Introduction :

Avec la prolifération des ordinateurs et l'essor de leur connectivité, plusieurs méthodes cryptographiques utilisent maintenant des logiciels plutôt que des machines électromécanique. Bien plus, bon nombre de logiciels commerciaux intègrent maintenant des méthodes de chiffrement pour offrir plus de sécurité. La majorité de ces méthodes requièrent des connaissances mathématiques trop complexes qui s'appellent la cryptographie moderne**.**

III.1.La cryptographie moderne :

La cryptographie moderne est un grand nombre de mathématiques fascinantes dont certaines ont été développées pour des applications cryptographiques, mais beaucoup dont est tiré du canon mathématique classique. La cryptographie mathématique moderne s'appuie sur de nombreux domaines des mathématiques, incluant notamment la théorie des nombres, l'algèbre abstraite (groupes, anneaux, corps), probabilités, statistiques et théorie de l'information, de sorte que les conditions préalables à l'étude le sujet peut paraître redoutable**[5].**

Le principe du chiffrement et du déchiffrement est illustré sur la figure 3.1

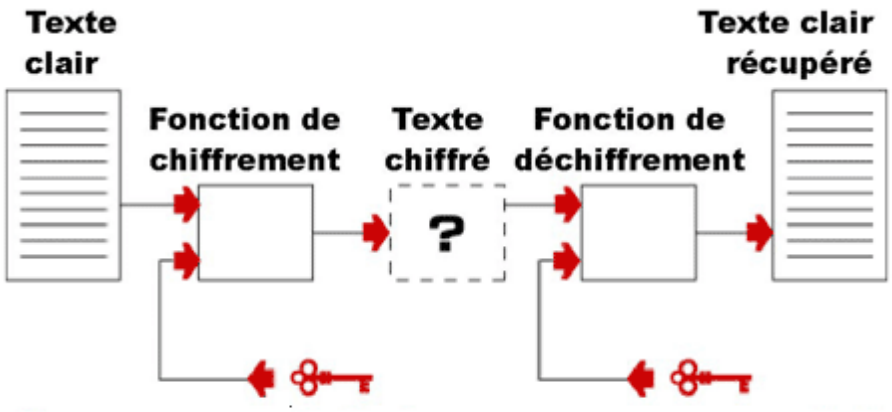


Figure III.1 : Principe de base d'un système cryptographique [19].

III.2.les principes de base de cryptographie moderne

Les principes fondamentaux d'un algorithme de cryptographie sont basés sur deux notions essentielles, énoncés par Shannon:

- **La confusion :** elle vise à rendre le texte aussi peu lisible que possible. Ceci peut se faire par une substitution systématique de symboles, ou par un algorithme de codage aussi complexe que l'on veut.

- **La diffusion :** elle vise à rendre chaque élément d'information du texte chiffré dépendant d'un nombre aussi grand que possible d'éléments d'information du texte clair. Ceci rend la découverte de l'algorithme, ou de la clé de cet algorithme, en principe plus difficile.

En plus des algorithmes de chiffrement et de déchiffrement, la cryptographie nécessite également un protocole, qu'il est nécessaire de suivre scrupuleusement afin de garantir la sécurité de la transaction. La transaction désigne ici l'ensemble des opérations menant du document en clair (texte clair) de départ au document (en principe identique) en clair lu par le destinataire. Le protocole désigne l'ensemble des opérations qui vont permettre le déchiffrement sûr par le destinataire, à commencer par l'entente sur l'algorithme de chiffrement et l'échange de clés de chiffrement **[13].**

III.3.les méthodes de la cryptographie moderne :

Parmi les méthodes de la cryptographie moderne, il existe deux grandes familles d'algorithmes à base de clés : les algorithmes à clé secrète ou algorithmes symétriques, et les algorithmes à clé publique ou algorithmes asymétriques comme figure le schéma suivant **[13] :**

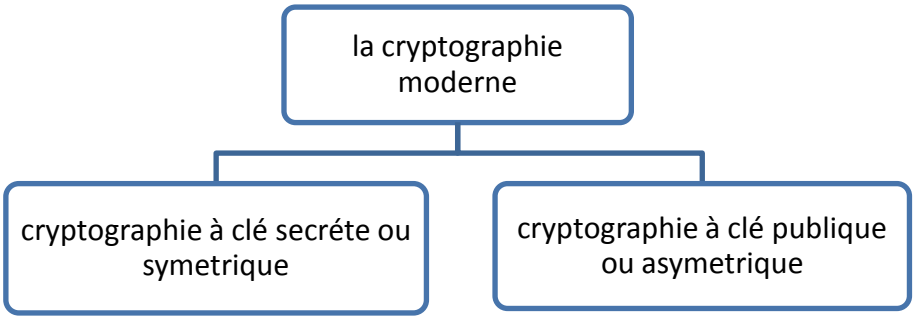


Figure III.2: les méthodes de la cryptographie moderne

Ils ont tous les deux leurs avantages et leur inconvénients. La différence qui existe entre ces deux types se situe au niveau de la clé **[17].**

3.1. La cryptographie à clé secrète (clé privé) ou symétriques :

Un système de chiffrement symétrique (aussi appelé chiffrement à clé privée ou chiffrement à clé secrète) consiste à utiliser une clé qui sert à chiffrer et déchiffrer les messages échangés entre deux personnes en communication. Cette clé est connue seulement par les deux personnes impliquées dans la communication, on aurait $k_C = k_D$, comme indiqué dans la figure suivante **[13] :**

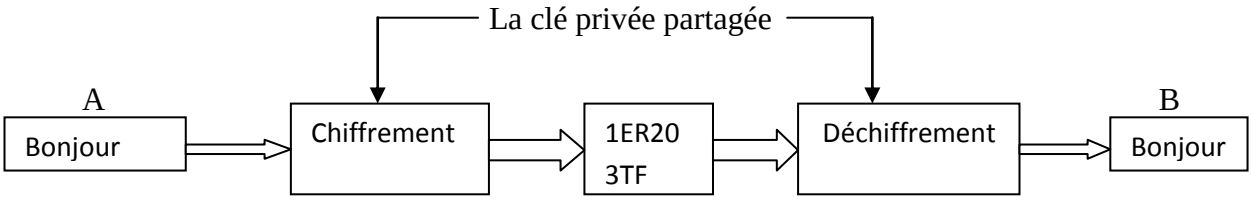


Figure III.3 : Principe de la cryptographie symétrique.

Ceci implique que les deux parties se sont au préalable mis d'accord sur la clé à utiliser pour chiffrement et déchiffrement dans leur échange d'information.

L'échange de clés dans un tel système est un problème non trivial. On parle de « canal sûr » pour désigner le média qui servira à cet échange **[13].**

- **Avantage et inconvénient de la cryptographie symétrique :**

Le grand avantage des algorithmes symétriques (à clé secrète) est la rapidité. Ce qui explique le fait que malgré ses limites, les algorithmes symétriques sont beaucoup plus utilisés dans les applications réelles. Le cryptage à clé symétrique possède des inconvénients dont le principal provient de l'échange des clés. En effet l'emploi d'un algorithme à clé secrète lors d'une communication nécessite un échange préalable d'un secret entre les deux communicants à travers un canal sécurisé ou au moyen d'autres techniques cryptographiques ; car si la clé est interceptée par une troisième personne, Celle-ci n'aura donc aucune difficulté pour découvrir la conversation.

Deuxième problème important, dans le cas d'échange dans un groupe de N personnes utilisant un cryptosystème à clé secrète susceptibles de communiquer, il faut distribuer $N*(N-1)/2$ clés **[13].**

➢ **Les systèmes de chiffrement de la cryptographie symétrique :**

On distingue volontiers deux types d'algorithme de chiffrement de la cryptographie symétrique ou à clé privée (secrète) :

- les algorithmes de chiffrement par bloc
- les algorithmes de chiffrement par flot (flux)

La différence est que les algorithmes de chiffrement par blocs traitent l'information sur des blocs de longueur fixe, alors que les algorithmes de chiffrement par flot traitent l'information de manière continue et rapide **[13].**

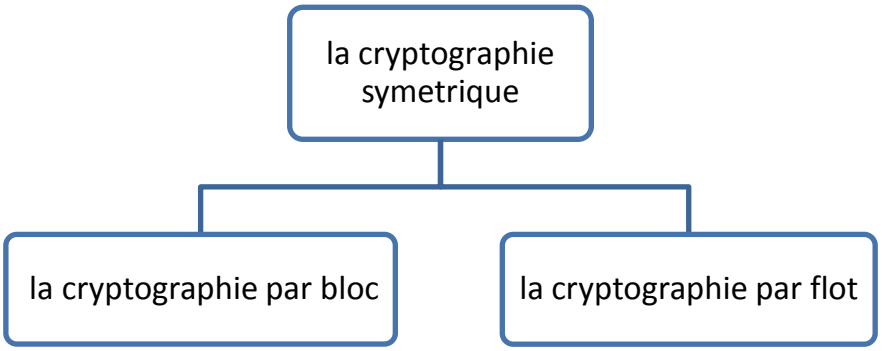


Figure III.4: les types de cryptographie symétrique.

3.1.1. Les algorithmes de chiffrement par bloc :

Dans un algorithme de chiffrement par bloc, chaque clair est découpé en bloc de taille fixe de la même longueur et chiffré bloc par bloc. Chaque bloc subit les mêmes transformations.

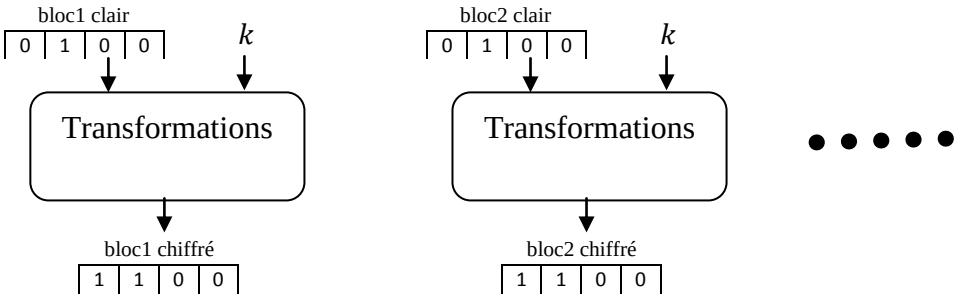


Figure III.5 : Principe du chiffrement par bloc [13].

Parmi les algorithmes ou système de chiffrement par bloc les systèmes de chiffrement RC5, RC6, IDEA, les plus connus sont le système de chiffrement DES et le système de chiffrement AES. **[13]**

3.1.1.a. Les modes de chiffrement : [15]

Le message est décomposé en blocs de longueur fixe avant de leur appliquer l'algorithme bloc par bloc. Pour cela, quatre modes sont possibles : ECB, CBC, OFB et CFB.

- **Mode ECB (Electronic code book) : [16]**

Ce mode est le plus simple : un même bloc est toujours codé de la même manière. Il n'y a pas de rétroaction de l'entrée ou de la sortie sur la fonction de chiffrement.

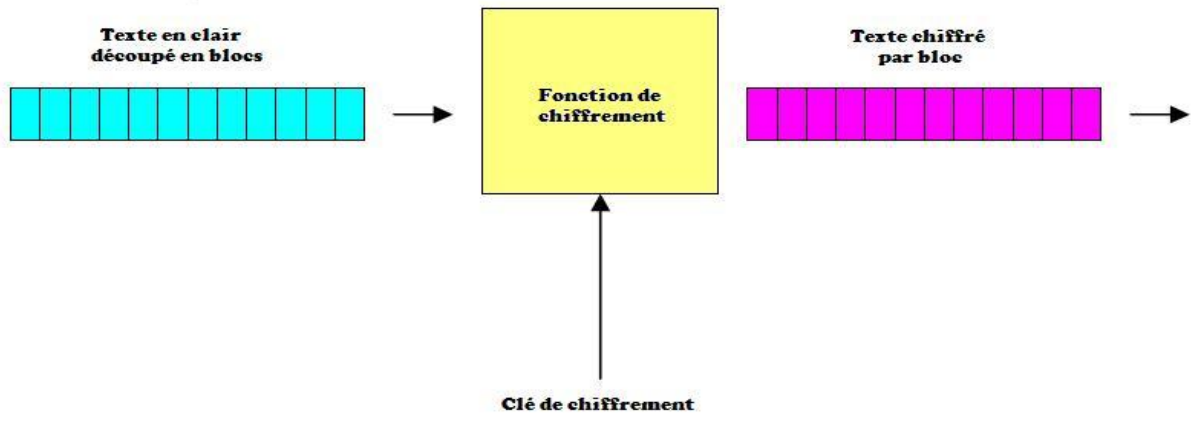


Figure III.6 : Le mode ECB [16].

Les avantages de mode ECB sont: [16]

- Le travail de chiffrement ou de déchiffrement peut être parallélisé. Plusieurs machines ou CPU peuvent travailler simultanément sur des parties différentes du message.
- Il permet un accès aléatoire dans le texte chiffré.
- Une erreur de transmission d'un bit affecte uniquement le décodage du bloc courant.

Les inconvénients de ce mode sont: [16]

- Les répétitions du texte en clair ne sont pas masquées et se retrouvent sous la forme de répétitions de textes chiffrés.
- Des portions complètes du message peuvent être modifiées, répétées ou remplacées sans difficulté.
- La perte de synchronisation (perte ou ajout d'un bit) est irrécupérable.

❖ **Mode CBC (Cipher bloc chaining) : [16]**

Dans ce mode de chiffrement, chaque bloc de texte en clair est d'abord combiné par un ou exclusif avec le dernier bloc du texte chiffré. La sortie de ce ou exclusif est ensuite appliquée à la fonction de chiffrement.

Ce mode de chiffrement dispose en plus d'un vecteur d'initialisation appelée IV (pour Initialisation Vector) qui permet d'initialiser le processus quand aucun bloc n'a encore été chiffré.

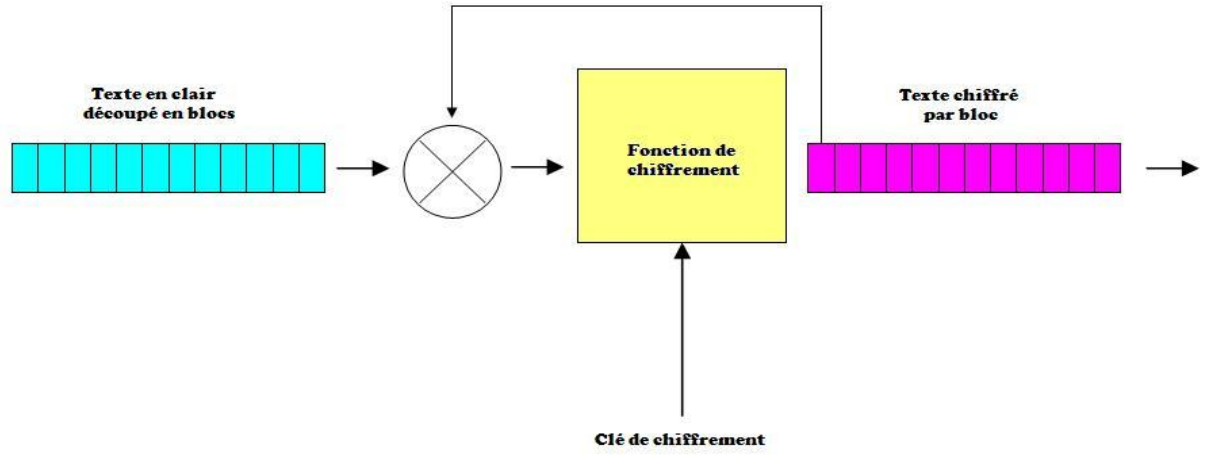


Figure III.7 : Le mode CBC [16].

Les avantages de mode CBC sont : [16]

- les répétitions de texte en clair sont masquées dans le texte chiffré ;
- la valeur du vecteur d'initialisation IV n'a pas besoin d'être secrète.

Les inconvénients de ce mode sont : [16]

- deux textes en clair commençant pareil auront le même début de texte chiffré ;
- une erreur de transmission d'un bit affecte uniquement le décodage du bloc courant ainsi que le décodage du même bit dans le bloc suivant ;
- la perte de synchronisation (perte ou ajout d'un bit) est irrécupérable.

❖ **Mode CFB (cipher feedback): [16]**

Les modes ECB et CBC travaillent avec sur des blocs de texte en clair (64 bits par exemple). Ces modes ne sont pas utilisables lorsque le chiffrement ne peut débuter que lorsqu'un bloc est complet. Sur des applications réseau, cela peut poser des problèmes, car les valeurs à

chiffrer arrivent de manière asynchrone sous forme d'octets et doivent être transmises immédiatement (cas du protocole Telnet par exemple).

Le registre à décalage est initialisé avec un vecteur d'initialisation. Le bloc complet est alors chiffré. L'octet de poids faible du texte chiffré est combiné par un ou exclusif avec l'octet de texte en clair. Le résultat de cette opération est alors transmis en même temps qu'il est injecté dans le registre à décalage.

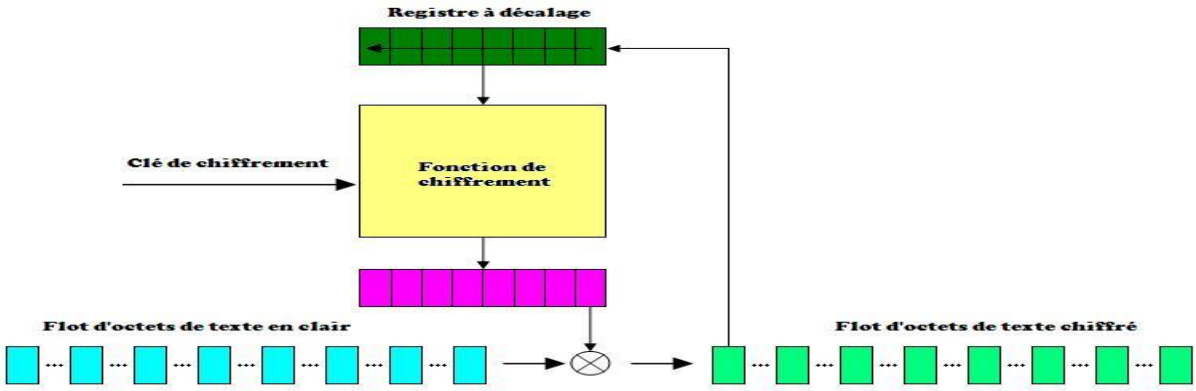


Figure III.8 : Le mode CFB [16].

Les avantages de mode CBF sont : [16]

- il est possible de chiffrer un flot de valeurs plus petites que la taille standard du bloc géré par l'algorithme.
- les répétitions de texte en clair sont masquées dans le texte chiffré.
- la valeur du vecteur d'initialisation IV n'a pas besoin d'être secrète.
- la perte de synchronisation (perte ou ajout d'un bit) est récupérable.

Les inconvénients de ce mode sont: [16]

- une erreur de transmission d'un bit affecte uniquement le décodage du bloc courant ainsi que le décodage du même bit dans le bloc suivant.

❖ Mode OFB (output feedback): [16]

Le mode OFB ressemble au mode CFB. La seule différence est que l'octet injecté dans le registre à décalage est l'octet de poids faible du texte chiffré.

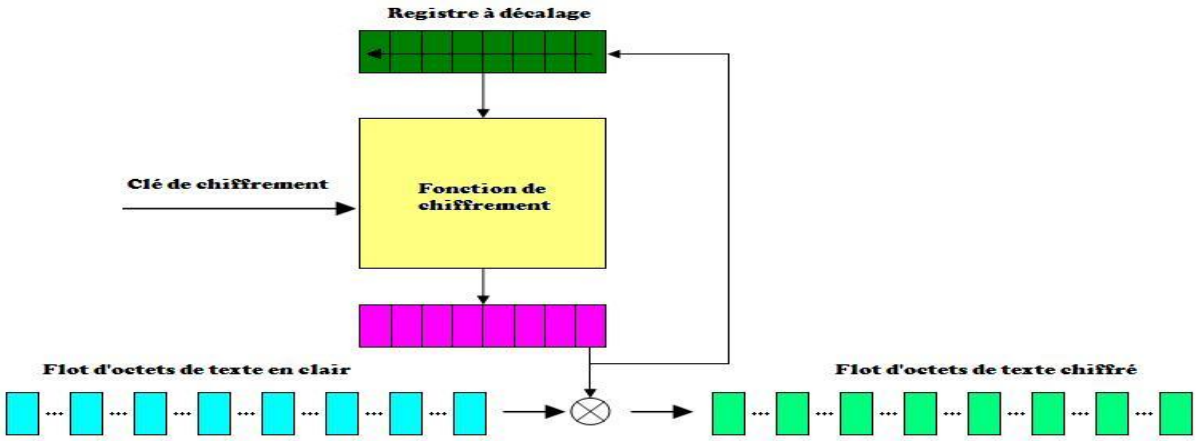


Figure III.9 : Le mode OFB [16].

Les avantages de mode OFB sont : [16]

- les répétitions de texte en clair sont masquées dans le texte chiffré.
- la valeur du vecteur d'initialisation IV n'a pas besoin d'être secrète.
- ce mode n'amplifie pas les erreurs. Une erreur de transmission d'un bit affecte uniquement ce bit lors du décodage.

Les inconvénients de ce mode sont: [16]

- la perte de synchronisation (perte ou ajout d'un bit) est irrécupérable.

Quel mode choisir :

Les quatre modes présentés ainsi que tous les autres ont chacun leurs avantages et leurs faiblesses. Il conviendra de choisir le mode en fonction de ce que l'on veut faire.

Pour finir, voici trois photos du mont Cervin dans les Alpes. La première photo est le **texte en clair**, la seconde est le **texte chiffré** de la photo utilisant un algorithme en mode ECB et enfin la dernière photo utilise le même algorithme en mode CBC.

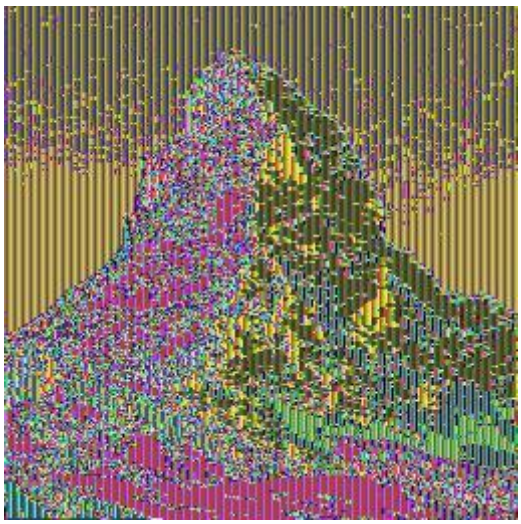
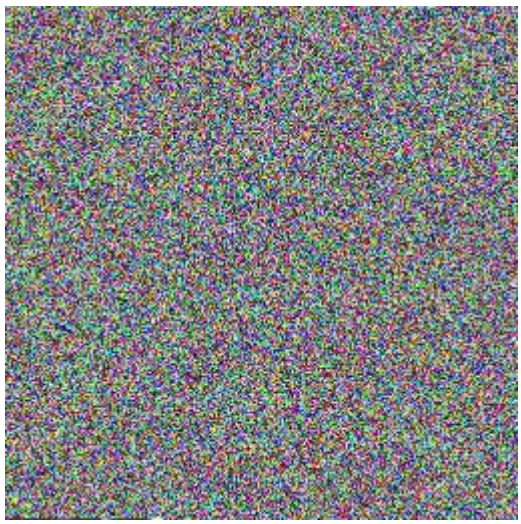
Non chiffré	Chiffré en mode ECB	Chiffré en mode CBC
		

Figure III.10 : Chiffrement mode ECB et mode CBC [16].

Ces photos montrent clairement que le mode ECB n'est pas adapté au chiffrement de photographies. Ceci ne veut toutefois pas dire que le mode ECB ne doit pas être utilisé, il a d'autres domaines d'utilisation.

Après le parcours des différents modes de chiffrement utilisables, nous allons maintenant nous intéresser aux différents algorithmes de chiffrement. Il existe en tout quatre grandes familles d'algorithmes utilisées par la cryptographie. Ces familles sont : **[16]**

- les algorithmes de calcul d'empreinte ;
- les algorithmes symétriques ;
- les algorithmes asymétriques ;
- les méthodes de génération de nombres aléatoires.

Chacune de ces familles est présentée dans les paragraphes suivants.

3.1.2 Les algorithmes de chiffrement par flot (flux) :

Les systèmes de chiffrement par flot, traitent l'information bit à bit. Leur principe est d'effectuer un chiffrement de Vernam en utilisant une clé pseudo-aléatoire. Cette clé (qu'on appellera suite pseudo aléatoire ou suite chiffrant) est générée par différents procédés à partir d'une clé courte aléatoire d'une longueur juste suffisante pour résister aux attaques exhaustive Les systèmes de chiffrement à flot sont utilisés dans les contextes où il est primordial de pouvoir chiffrer et déchiffrer très rapidement et où les ressources matérielles, comme par

Exemple la taille du circuit ou la capacité de stockage, sont très restreintes. Ceci est la raison pour laquelle les chiffrements à flot sont implantés dans les téléphones mobiles et dans d'autre d'autres dispositifs embarqués **[13].**

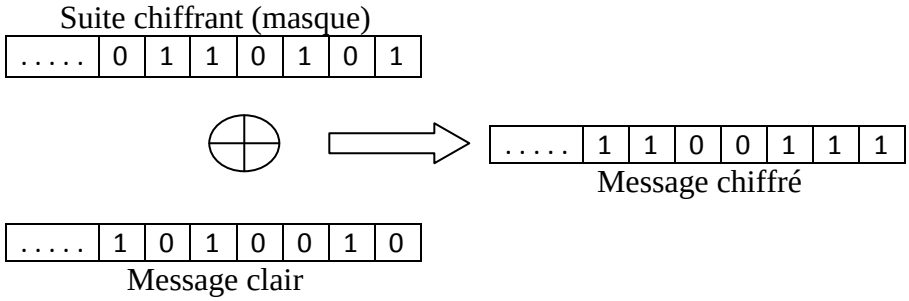


Figure III.11 : Principe du chiffrement par flot [13].

On peut donner comme un exemple de chiffrement par flot, les systèmes de chiffrement :

A5 : utilisé dans les téléphones mobiles de type GSM, pour chiffrer la communication par radio entre le mobile et l'antenne-relais la plus proche,

E0 : utilisé par le protocole Bluetooth

Py : un algorithme récent d'Eli Biham

3.1.2.A. Rivest Cipher 4 "RC4" :

RC4 (pour Rivest Cipher 4) a été conçu par Ron Rivest en 1987, pour RSA Security, Il fut d'abord vendu avec des accords de confidentialité et resta donc longtemps secret. Mais en septembre 1994, un abonné anonyme de la mailing List Cypherpunk publie un code source, vraisemblablement issu de la rétro-ingénierie d'un programme compilé, qu'il affirme être RC4.

Plusieurs utilisateurs confirmeront par la suite la compatibilité de ce code avec le code propriétaire de RSA Security.

RC4 n'étant protégé par aucun brevet, son utilisation est libre de droits. En revanche, le nom RC4 étant une marque déposée, des implémentations alternatives se sont vues baptisées ARC4 (pour Alleged RC4). **[13]**

3.1.2.B. Principe de fonctionnement de RC4 :

RC4 est un algorithme de chiffrement par flux. C'est un générateur pseudo-aléatoire qui génère une suite d'octets (appelée keystream). Ces octets sont ensuite combinés au texte à chiffrer par un OU exclusif (XOR) **[13].**

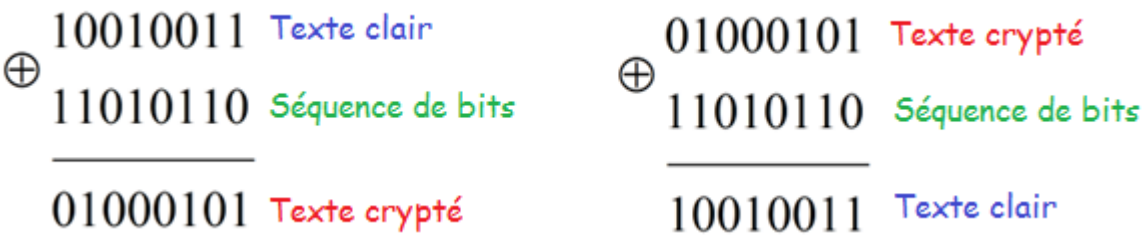


Figure III.12: Opération XOR logique lors d'un cryptage et décryptage [13].

3.1.2.C. Le fonctionnement de RC4 :

RC4 réalise quatre (4) étapes essentielles : **[13]**

- **la première étape :** initialisation d'un tableau S qui s'appelle le vecteur d'initialisation (initialization vector IV en anglais)
- **la deuxième étape** : consiste à faire une permutation entre le vecteur d'initialisation S de première étape avec une clé de chiffrement donnée
- **la troisième étape :** consiste à faire une génération d'une suite des octets pseudo aléatoires deuxième étape
- **la dernière étape** : consiste à faire un XOR entre le résultat de troisième étape et le message en clair octet par octet (chaque octet généré de troisième étape avec un octet de message clair respectivement)

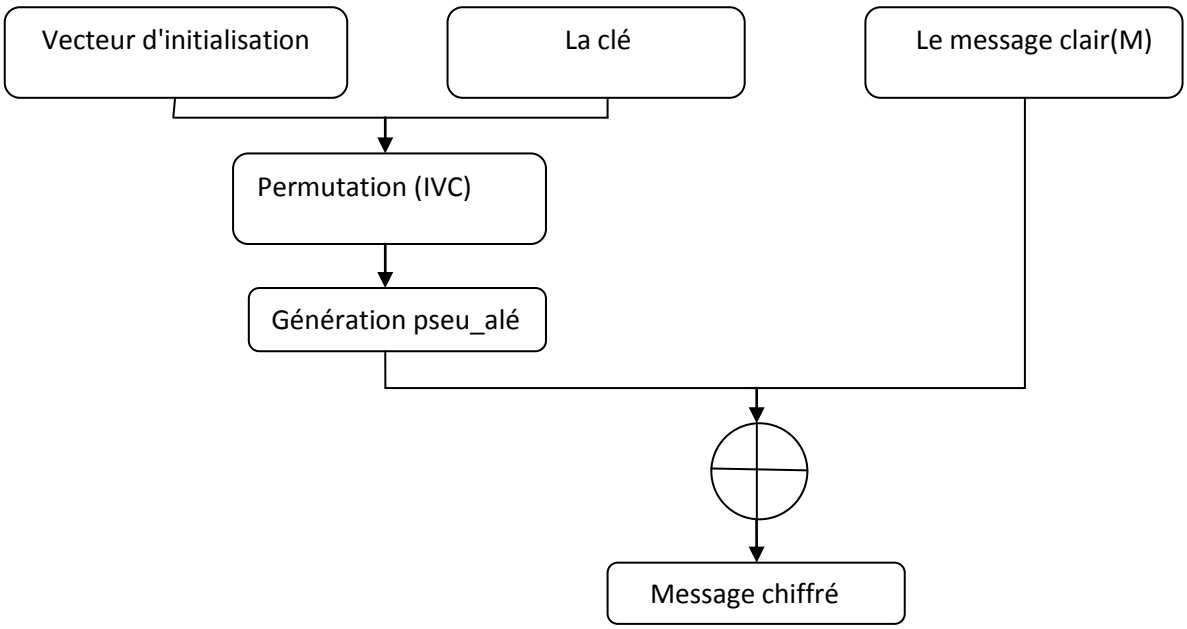


Figure III.13.: schéma des étapes de fonctionnement de RC4

- Algorithme de permutation de la clé (Key Scheduling Algorithm) (KSA) :L'état interne de RC4 est constitué de deux entiers i et j, et d'une permutation *S* de taille N = 256. Pour constituer *S*, on part de la permutation identité. On parcourt cette permutation en échangeant chaque élément avec un autre élément choisi en fonction de la clé. Cet algorithme est appelé *Key Scheduling Algorithm*, ou KSA. Algorithm 1 Algorithme d'initialisation de RC4.

K est un clé de longueur L.

S est une permutation des entiers de 0 `à 255.

- Initialisation de la permutation identité

```
For i in 0 to 255 do
    S[i]:= i
End for
```
} étape1

- Mélange de S dépendant de K

Étape 2 {
```
J: = 0
For i in 0 to 255 do
    j: = (j + S[i] + K[i mod L]) mod 256
    Swap (S[i], S[j])
End for
```

Algorithme de générateur pseudo aléatoire Pseudo Random Generator Algorithme (PRGA)

Une fois son état interne initialisé, RC4 peut générer un par un des octets pseudo aléatoires, comme l'indiqué la figure ci-dessous :**[12]**

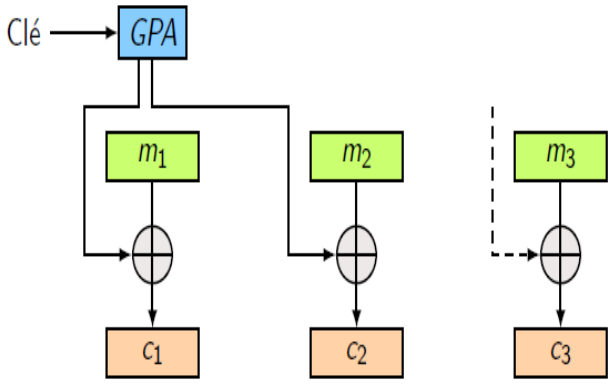


Figure III.14: générateur pseudo aléatoire

Algorithme 2 Générateur pseudo aléatoire de RC4.

i := i + 1
j := j + S[i]
Swap (S[i], S[j])
Octet : = S [(S[i] + S[j]) mod 256]
} étape 3

Retourner octet

Octet_ crypté := octet_ de_ sortie XOR octet_ donné) étape 4

3.1.2.D. Utilisation de RC4 : [13]

Le RC4 est très utilisé dans le monde, on le trouve dans :

- le grand moteur de recherche "Google" dans " http "
- l'oracle SQL,
- SSL/TLS (Secure Sockets Layer/Transport Layer Security)
- le WEP(wired equivalent privacy) et WPA(wifi protected access)

Exemple d'utilisation de RC4 : [13]

Le WPA : Le WPA (Wifi Protected Access) est une évolution du Wep. En effet, il a essayé de corriger au maximum toutes les failles du Wep. Le WPA possède deux modes, le mode « Enterprise » et le mode « personal ».Le mode « Enterprise » nécessite une authentification 802.1x ainsi qu'une clé WPA. Le plus souvent, le serveur d'authentification est un serveur radius.

Le mode « personal » est un mode dans lequel l'authentifié ne se fera pas avec un serveur d'authentification, mais seulement avec la clé WPA (ce mode est appelé WPA-PSK, car on utilise une clé). Le plus souvent c'est ce qui est utilisé chez tous le monde alors que le mode « Enterprise » est plus utilisé dans un environnement en entreprise, ou université par exemple

La clé WPA utilisée n'est plus comme en WEP une clé en hexadécimal, mais c'est un mot de passe à rentrer.

Le WPA repose aussi sur le cryptage RC4, comme avec WEP, mais il utilise en plus le protocole TKIP (Temporal key Integrity Protocol). TKIP utilise un IV de 48 bits. La clé RC4 (IV + mot de passe) utilisée pour crypter les paquets, est distribuée de façon bien intelligente qu'avec WEP.

Pour le cryptage des messages, l'IV n'est plus de 24 mais de 48 bits. De plus un mécanisme à été mis en place avec TKIP afin d'éviter les clés RC4 faibles. La clé temporaire mise en place avec TKIP changera à chaque session ou alors changera plusieurs fois au cours de la session. Ce qui permet d'avoir des paquets qui seront cryptés différemment. Ce qui est une sécurité en plus.

3.1.2.E. Performances de cryptosystème RC4 : [13]

Le chiffrement RC4 est extrêmement rapide, sûrement le plus rapide des chiffrements utilisés à l'heure actuelle même, simple à comprendre et à implémenter.

Cependant, il comporte quelques failles de sécurité qui sont exploitables de façon plus efficace qu'une recherche exhaustive de clé.

Fluhrer, Mantin et Shamir ont en effet explicité 2 faiblesses dans la spécification de l'algorithme RC4**.**

La première repose sur le fait qu'il existe de larges ensembles de clés dites faibles, c'est-à-dire des clés dont quelques bits seulement suffisent à déterminer de nombreux bits dans la table d'état *S* (avec une forte probabilité), ce qui affecte directement les données produites en sortie c'est l'attaque nommée <invariance weakness>.

La deuxième attaque connue est la <known IV attack >. Elle nécessite, comme son nom l'indique, la connaissance de l'*IV* (Initiale Value), ce qui peut être le cas lorsqu'il circule en clair sur le réseau, ainsi que la connaissance du premier octet du message *M* (à retrouver). Dans un certain nombre de cas (les cas résolus, suivant l'expression de Fluhrer, Mantin et Shamir), la connaissance de ces 2 éléments permet de déduire des informations sur la clé *K*.

Selon les trois chercheurs, ces 2 attaques sont applicables et peuvent permettre une récupération complète de la clé avec une efficacité bien supérieure à l'attaque par recherche exhaustive**.**

Cependant, il faut savoir que ces attaques ne sont pas réalisables dans tous les cas. Ainsi l'utilisation du chiffrement RC4 dans le protocole SSL, par exemple, est faite de façon à éviter ses deux types d'attaques**.**

3.2. La cryptographies à clé public ou asymétrique : [13]

Jusqu'au milieu des années 70, les seuls cryptosystèmes connus étaient symétriques : la clé de chiffrement k_c était la même que la clé de déchiffrement k_d, ce qui obligeait à garder secrète la clé k_c. Cela pose alors le problème crucial de l'échange de clé, difficile à résoudre dans le cas d'un système développé à grande échelle**.**

En 1976, W. Diffie et M. Hellman introduisirent le concept de cryptographie à clé publique : dans ce type de système, la clé de chiffrement est publique, c'est à dire connue de tous. Seule la clé de déchiffrement reste secrète (clé privée). La source peut utiliser la clé publique pour chiffrer le message, alors que le destinataire utilisera la clé privée pour déchiffrer le message, on aura $k_C \neq k_D$.

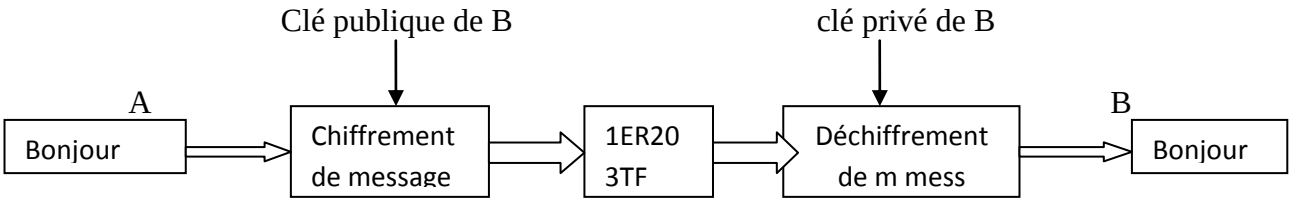


Figure III.15: la cryptographie asymétrique

- **Avantages et inconvénients de la cryptographie asymétrique : [13]**

L'avantage majeur de chiffrement à clés publiques est :

Permettre d'échanger destinataire n'ont plus besoin de partager des clés secrètes via une voie de transmission sécurisée .les communication impliquent uniquement l'utilisation de clés publiques et plus aucune clé privée n'est partagée.

Les inconvénients des systèmes à clés publiques sont :

La lenteur à cause de la complexité des fonctions utilisées pour le chiffrement et la nécessite d'une certification.

3.2.1. Les systèmes de chiffrement à clés publiques : [6]

Parmi les cryptosystèmes à clé publique, on trouve Elgamel (d'après le nom de son inventeur, taher elgamal), RSA (d'après le nom de ses inventeurs, Ron Rivest, Adi Shamir et Leonard Adleman), sont des exemples de systèmes de cryptographie à clé publique.

3.2.1.A. Elgamel : [6]

C'est en 1985 qu'un système de chiffrement praticable utilisant le logarithme discret a été proposé par Tahar El Gamal**.**

Ce système repose sur le principe suivant :

Supposons que A souhaite envoyer un message chiffré à B. Pour cela :

- A et B s'entendent au préalable sur un groupe G dans lequel travailler, sur un générateur g de G , et sur un grand nombre premier p .
- A doit mettre en place une paire de clés, l'une secrète (privée) a ; et l'autre publique $\beta_A = g^a$.
- B doit mettre en place une paire de clés, l'une secrète (privée) b ; et l'autre publique $\beta_B = g^b$
- Pour chiffrer le message m qu'il souhaite envoyer, A choisit d'abord un entier k aléatoire et premier avec p-1. Il calcule les deux éléments suivants de G :

$$\delta = m.\beta_B{}^k \bmod p$$

$$\gamma = g^k \bmod p$$

Le texte chiffré est alors la paire (δ ; γ).

Pour déchiffrer le message m, Bob calcule $\delta.(\gamma^b)^{-1} mod\ p$.

Le message chiffré, sous la forme du couple (δ ; γ), est de taille deux fois supérieure à l'information transmise (le message m). C'est un inconvénient du système.

3.2.1.B. Algorithme RSA : [6]

L'algorithme RSA publié en 1978 par Rivest, Shamir et Adleman dans leur article « A Method for Obtaining Digital Signatures and Public-Key Crypto-Systems ». Ce système cryptographique asymétrique est le plus utilisé de nos jours dans l'industrie grâce à sa simplicité d'implémentation. Le crypto système à clé publique RSA est basé sur le calcul de l'exponentiation modulaire ; sa sécurité réside dans la difficulté à factoriser le produit de deux grands nombres premiers.

Le principe de RSA est alors le suivant :

- la clé publique d'un utilisateur est formée d'un nombre n produit de deux nombres premiers p et q n= p*q, et d'un entier e premier avec $\varphi(n) = (p-1)(q-1)$.
- Les valeurs de n et e sont publiées dans un annuaire tel que PGCD (e, $\varphi(n)$)= 1.
- La clé secrète correspondant est un entier d qui vérifié $e * d = 1\ mod\ \varphi(n)$.
- m est le message à chiffrer dont la taille en bits est comprise entre 0 et n-1.
- Pour envoyer un message m à B, A va donc chercher la clé publique de B et il calcule le message chiffré c correspondant par : $c = m^e \bmod n$.
- Lorsqu'il reçoit le message chiffré c, B retrouve le texte clair en calculant

$$c^d \bmod n = m.$$

Afin de garantir un niveau de sécurité élevé lors de l'implémentation du cryptosystème RSA, la taille de la clé de déchiffrement, qui dépend de la taille du modulo, doit être au moins égale à 1024 bits et plus, ce qui induit beaucoup de calcul.

3.2.1.C. Systèmes de signature : [13]

Un codage asymétrique, quel que soit, peut être très avantageux du fait que la clé privée ne soit détenue que par le receveur du message. Cela assure notamment que la clé privée ne puisse être divulguée, car elle n'est connue de personne d'autre que son utilisateur. Cependant, un problème apparaît rapidement : étant donné que la clé de chiffrement est publique, comment authentifier l'expéditeur. C'est là qu'intervient le système de signature. En effet, si la personne qui envoie le message y ajoute une signature qui lui est propre, il n'y a plus aucun doute sur son identité.

Un procédé de signature numérique consiste à joindre au texte clair un petit nombre de bits qui dépendent simultanément du message et de son auteur.

Un schéma de signature est composé d'une fonction de signature et d'une fonction de vérification.

- La fonction de signature est paramétrée par une clé secrète propre au signataire; elle associe à tout message clair une signature.
- La fonction de vérification, elle, ne nécessite la connaissance d'aucun secret. Elle permet à partir du message clair et de la signature de vérifier l'authenticité de cette dernière.

3.2.1.D. Fonction de hachage : [18]

Une fonction de hachage est une fonction qui prend en entrée une chaîne de bits arbitrairement longue (ou octets) et produit un résultat de taille fixe. Les fonctions de hachage sont parfois appelées fonctions de résumé de message, et le hachage le résultat est également connu sous le nom de condensé ou d'empreinte digitale. Les fonctions de hachage peuvent être utilisées en pseudo-aléatoire cryptographique générateurs de nombres pour générer plusieurs clés à partir d'un même secret partagé. Et ils ont une propriété unidirectionnelle qui isole les différentes parties d'un système, assurant que même si un attaquant apprend une valeur, il n'apprend pas les autres.

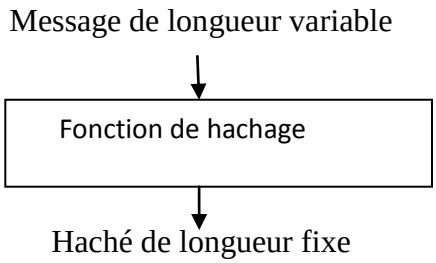


Figure III.16: principe de fonction de hachage

Fonctions de hachage usuelles :

– **MD4 et MD5** (Message Digest) furent développées par Ron Rivest. MD5 produit des hachés de 128 bits en travaillant les données originales par blocs de 512 bits.

– SHA-1 (Secure Hash Algorithm 1), comme MD5, est basé sur MD4. Il fonctionne également à partir de blocs de 512 bits de données et produit par contre des condensés de 160 bits en sortie. Il nécessite donc plus de ressources que MD5.

– **SHA-2** (Secure Hash Algorithm 2) a été publié récemment et est destiné à remplacer SHA-1. Les différences principales résident dans les tailles de hachés possibles : 256, 384 ou 512 bits. Il sera bientôt la nouvelle référence en termes de fonction de hachage.

– **RIPEMD-160** (Ripe Message Digest) est la dernière version de l'algorithme RIPEMD. La version précédente produisait des condensés de 128 bits mais présentait des failles de sécurité importantes. La version actuelle reste pour l'instant sûre; elle produit comme son nom l'indique des condensés de 160 bits.

Un dernier point la concernant est sa relative gourmandise en termes de ressources et en comparaison avec SHA-1 qui est son principal concurrent.

– Tiger : Tiger est une fonction de hachage cryptographique conçue par Ross Anderson et Eli Biham en 1996. Tiger fournit une empreinte sur 192 bits mais des versions sur 128 et 160 bits existent aussi. Ces versions raccourcies prennent simplement les premiers bits de la signature de 192 bits.

3.2.1.E. Certificats numériques : [13]

Un certificat est un document numérique qui garantit la relation entre un sujet (personne physique ou machine) et sa clé publique. La réalisation du certificat repose fortement sur l’existence du mécanisme de signature.

La forme générale du certificat telle que décrite dans la norme X.509, comprend les informations suivantes :

- Subject : nom distingué, clé publique ;
- Issuer : nom distingué, Signature ;
- Period of Validity : date de début, date de fin ;

Administrative Information : version, numéro de série;

III.4. la cryptographie mixte (symétrique et asymétrique) : [13]

Le problème non trivial qui se pose lors de l'établissement des communications sécurisées entre plusieurs entités est bien la distribution des clés. Si n personnes veulent communiquer secrètement 2 à 2, il leurs faut un certain nombre de clés.

Pour un cryptosystème asymétrique, la clé publique par définition peut être connue de tous, et la clé de déchiffrement n'a pas besoin d'être transmise à son correspondant, donc le nombre de clés nécessaires est $2n$: chaque personne détient une clé secrète et diffuse une clé publique. Alors que si un chiffrement symétrique est utilisé, il faut avoir u avoir une clé secrète pour chaque paire de correspondant, c'est à dire en tout $\frac{n(n-1)}{2}$ clés.

Mais c'est connu que les cryptosystèmes symétriques sont beaucoup plus rapides que les cryptosystèmes asymétriques.

De ce fait, on utilise souvent un cryptosystème mixte : une clé symétrique est échangée de manière transparente en utilisant un algorithme asymétrique, et ensuite le reste de la transaction utilise un algorithme symétrique pour chiffrer les données. Ceci implique néanmoins une entente (éventuellement implicite) entre client et serveur quant à l'échange nécessaire de la clé symétrique, c'est ici qu'intervient le protocole de la transaction.

4.1. Chiffrement du futur :

Les systèmes de cryptographie symétrique ou clé secrète sont plus rapides que les systèmes de cryptographie asymétrique ou clé publique mais un problème se pose : comment se fait la distribution des clés secrète entre l'émetteur et le récepteur sans espionnage. Les physiciens pensent à la mécanique quantique pour la transmission des clés secrètes : c'est la cryptographie quantique **[13].**

III.5. La cryptographie quantique :

La cryptographie quantique n'est pas une parte de la cryptographie moderne, elle a été créé pour résoudre le problème de distribution des clés secrètes

Elle utilise les photons pour la transmission des clés, chaque bit de clé transmit est un photon lumineux polarisé (0°,90°,45°et 135°) transmit sur un canal quantique qui est la fibre optique, le premier protocole de la cryptographie quantique est le BB84 **[13].**

5.1. Fonctionnement :

L'apport majeur des mécanismes quantiques pour le traitement de l'information réside dans le fait que les données peuvent être codées sur des photons de lumières (notion de bit quantique ou qubit), polarisés selon différentes orientations. Six types de polarisation sont possibles : horizontale (0°), verticale (90°), diagonale droite (45°), diagonale gauche (135°), circulaire droite et circulaire gauche.

Le concept d'un protocole de communication utilisant une transmission de bits quantiques comme illustré ci-dessous :

Deux interlocuteurs communiquent à travers deux canaux, un canal optique ; où ils peuvent s'échanger des photons polarisés et un canal public non protégé ; où ils peuvent discuter.

Les deux parties se mettent d'accord sur la signification des polarisations des photons. Verticale et diagonale droite pour le « 1 » et horizontale et diagonale gauche pour le « 0 », où chaque bit est polarisé par un seul photon.

Le récepteur choisit une suite de bases (filtres) qui serviront à détecter les photons et mesure leurs polarisations. Il existe deux types de filtre «+ » pour le vertical et l'horizontal et « ×» pour les diagonales.

L'émetteur envoie une suite de photons polarisés au récepteur sur le canal optique, à la réception, si la polarisation ne correspond pas au filtre choisi, le photon est détruit, donc le récepteur perd l'information portée par le photon détruit. En effet, il existe une chance sur deux que le filtre choisi par le récepteur corresponde à la polarisation des photons envoyés. Puis le récepteur transmet à l'émetteur les filtres choisis via le canal publique .ce dernier sélectionnera certains ou tout dont l'orientation laisse passer les photons polarisés et dont l'information peut être extraite, ce processus s'appelle la distillation.

A l'issue de cette réception, les deux parties communicantes appliquent la même opération arithmétique et logique sur l'information envoyée et sur le résultat de l'échange quantique pour obtenir une clé secrète dite, clé quantique.

Par ce mécanisme d'échange d'informations basé sur la polarisation des photons et le choix de filtres, on peut assurer la confidentialité de l'information secrète générée et partagée par uniquement deux interlocuteurs sans que cette information soit stockée ou transmise. Cette information secrète peut être une clé de chiffrement à utiliser dans un système de chiffrement classique (AES par exemple).

De plus, la physique quantique permet de générer des nombres vraiment aléatoires (non prédictibles) qui peuvent être utilisés pour construire les clés de chiffrement des systèmes de cryptographie classique qui verraient de ce fait, leur niveau de sécurité augmenté **[13].**

5.2. Principaux algorithmes et techniques :

Plusieurs algorithmes de cryptographie utilisant une transmission quantique de clés de chiffrement ont été publiés, ces algorithmes, différent selon le nombre d'états polarisés des photons, la sécurité apportée et la facilité d'implémentation. Parmi eux : BB84 à quatre états, protocole à deux états et protocole à six états.

Ainsi différentes techniques de génération de photons ont été inventées pour mettre en pratique la cryptographie quantique **[13].**

5.3. Performances :

La cryptographie quantique permet d'obtenir un niveau de sécurité qui peut être qualifié d'inconditionnel pour ce qui concerne la génération de clés. L'échange de clés quantique est prouvé inviolable et peut être mis en œuvre pour servir un algorithme de chiffrement classique ou être intégré dans des protocoles de cryptographie existant comme IP sec par exemple **[13].**

5.4. Limites de la cryptographie quantique :

L'application de la physique quantique à la cryptographie est actuellement limitée à l'échange de clés cryptographiques. Comme il est encore techniquement difficile de générer et d'isoler un photon. Il existe toujours une probabilité d'avoir deux photons ou plus envoyé à la destination, ce qui pose un problème puisque la fiabilité d'une transmission quantique est basée sur la propriété du non clonage d'un photon. De plus, du point de vue des performances, le débit d'échange demeure inférieur au mégabit par seconde **[13].**

Conclusion :

Dans ce chapitre, nous nous sommes intéressés à la cryptographie moderne et son rôle principale dans la sécurisation des donnés de communication, En effet, nous avons présenté les bases de la cryptographie moderne qui remplit deux partie :

La première partie est le cryptosystème symétrique où l'émetteur et le récepteur possèdent la même clé secrète de chiffrement et déchiffrement tel que : DES, AES, RC4 etc…

La deuxième partie est relative au cryptosystème asymétrique où le seule le récepteur possède la clé secrète de déchiffrement et la clé de chiffrement est public, mise à la disposition de tout émetteur.

Le chapitre suivant est dérolu à l'étude et simulation de cryptosystème RC4.

Chapitre IV

Etude et simulation RC4

Introduction :

Après faire une étude théorique sur le cryptosystème symétrique par flot RC4 dans le chapitre précédent, on va le faire une étude pratique dans ce chapitre avec un exemple de chiffrement d'un message expliqué et détaillé sur son fonctionnement, ensuite on a fait plusieurs simulations sur le chiffrement(cryptage) et le déchiffrement des messages, des simulations sur le cryptage et décryptage des images sous logiciel MATLAB et une implémentation sur le cryptage d'une vidéo qui est plus compliqué qu'un message ou image sous DEV C++

IV.1. Le cryptosystème RC4 :

Le cryptosystème symétrique RC4 est un système de chiffrement par flot ,il chiffre les données lors d'une communication ou stockage des données pour les protéger de manière continu et très rapide, son fonctionnement sert à générer une suite pseudo aléatoire pour la combiner avec la donnée clair avec l'opération XOR logique pour le cryptage et utilise l'opération inverse c'est-à-dire qu'il combine le texte crypté et la séquence de bits pseudo aléatoire avec l'opération XOR logique pour le décryptage.

IV.2. Le fonctionnement de RC4 :

L'algorithme de cryptage RC4 est constitué de plusieurs parties, chacune importante dans la capacité du cryptage et du décryptage.

2.1 Le cryptage de RC4 :

On va décrire ci-dessous toutes ces parties pour mieux comprendre étape par étape comment se déroule le cryptage d'un texte en clair avec le RC4, le fonctionnement de cryptosystème RC4 se fait en quatre (4) étapes essentielles :

- la première étape : l'initialement d'un tableau S ou vecteur d'initialisation
- la deuxième étape : permutation entre le tableau S et la clé de chiffrement
- la troisième étape : génération d'une suite pseudo aléatoire
- la quatrième étape : la suite pseudo aléatoire XOR le message clair

2.1.1 L'initialisation de tableau S :

L'initialement d'un tableau S ou s'appelle aussi le vecteur d'initialement (initialized vector IV en anglais), cette étape consiste de générer un tableau S de taille N =256 selon l'algorithme suivant :

Pour i de 0 à 255

S [i]:= i

Fin pour.

- l'exécution de cette étape nous a donné le tableau d'état S suivant :

i =	**0**	**1**	**. . . . . .**	**255**
00000000		**00000001**	**.....**	**11111111**

2.1.2 .La permutation de vecteur d'initialisation ou tableau d'état S :

Cette étape consiste à faire une permutation ou brouillage de la clé secrète donné avec le vecteur d'initialisation (IV) généré par l'étape 1 :

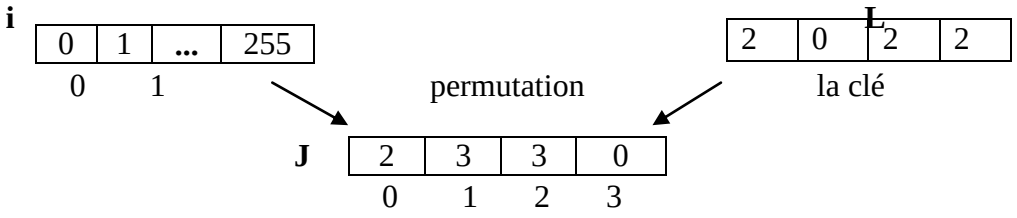


Tableau IV.1: la permutation de Tableau S avec la clé

Cette permutation se fait selon l'algorithme 2 suivant :

J := 0

Pour i de 0 à 255

j := (j + S[i] + clé [i mod longueur clé]) mod 256

Permuter (S[i], S[j])

Fin pour.

2.1.3. Génération d'une suite pseudo aléatoire :

Cette étape consiste à générer pour chaque octet (8bits) de message en clair un octet (8bits) de la suite généré par l'algorithme 2, c'est à dire que si notre message en clair contient 10 caractères (8bits pour chaque caractère), cet algorithme fait 10 itérations pour chiffrer chaque caractère de message en clair, il fait une autre permutation pour générer le plus possible une suite pseudo aléatoire, cette génération des octets de cryptage se faite selon l'algorithme3 suivant :

i := 0
j := 0
Tant que générer un octet
i := (i + 1) mod 256
j := (j + S [i]) mod 256
permuter (S [i] , S [j])
octet_de_sortie := S [(S [i] + S [j]) mod 256]
Fin tant que.

2.1.4. Le cryptage :

Cette dernière étape consiste à chiffrer notre donnée, après la génération des octets de suite pseudo aléatoire, octet par octet en utilisant l'opération ou-exclusif (XOR), c'est à dire que c'est notre message en clair à 10 octets (caractères), l'étape 3consiste à générer 10 octets ou chaque octet de message en clair sera combiné avec un octet de suite selon la ligne algorithmique suivante :

Octet crypté = octet_de_sortie XOR octet clair

Pour comprendre les quatre étapes vues précédemment, illustrons un exemple concret où chaque étape sera présentée et calculée.

2.1.5. Le décryptage de RC4 :

On refaire toutes les quatre étapes précédentes de cryptage, le même principe, il suffit juste de mettre à la place de texte clair le texte chiffré.

IV.3. Le cryptage d'un message avec le RC4 :

Cryptons par exemple le mot **« Master2021 »** avec la clé **« 2016 ».**

Avant toute, il faut mentionner que pour l'échange d'information, les caractères sont normalisés et codés en ASCII. Donc, le mot « Master2021 » et la clé « 2016» correspondent en **ASCII** à :

M = 01001101
a = 01100001
s = 01110011
t = 00110100
e = 01100101
r = 01110010
2 = 00110010
0 = 00110000
2 = 00110010
1 = 00110001

2 = 00110010
0 = 00110000
1 = 00110001
6 = 00110110

❖ ***Etape 1 : Initialisation du tableau d'état S :***

Dans notre exemple, nous supposons que la taille du tableau d'état est de 4 en raison de l'impossibilité d'illustrer manuellement les calculs dans un tableau d'état de taille 256.
Pour bien comprendre réellement le principe et le déroulement du fonctionnement de l'algorithme de cryptage RC4, on représente les valeurs, ainsi que les opérations en binaire.
Initialisation du tableau d'état selon l'algorithme 1**:**

```
Pour i de 0 à 3
    S [i] := i
Fin pour
```

Le déroulement de cette boucle se déroulera comme suit :

- **1ere iteration**

Pour i = 0

S [0] = 0

Le tableau d'état S devient :

00000000			
0	1	2	3

- **2eme iteration**

Pour i = 1

S [1] = 1

Le tableau d'état S devient :

00000000	00000001		
0	1	2	3

On va remplir le tableau d'état avec le même algorithme jusqu'à la **3eme itération**

Le tableau d'état S devient comme suit :

00000000	00000001	00000010	00000011
0	1	2	3

- ***Etape 2 : la permutation de tableau d'état S :***

Cette étape consiste de mélanger le tableau d'état S avec la clé donnée :

Clé [i]	Clé [0]	Clé [1]	Clé [2]	Clé [3]
La clé	2	0	1	6

Tableau IV.2: la clé donnée

Selon l'algorithme 2 suivant :

j := 0

Pour i de 0 à 3

j := (j + S[i] + clé [i mod longueur_clé]) mod 4

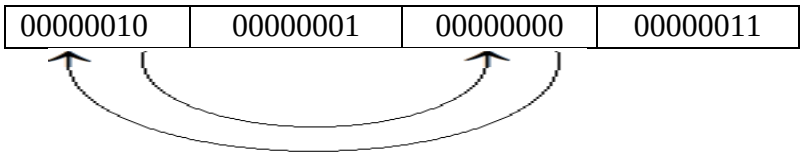
Permuter (S[i], S[j])

Fin pour.

- ***1ere iteration:***

j:= 0
Pour i = 0
j = (0 + S [0] + clé [0 mod 4]) mod 4
= (0 + 0 + clé [0]) mod 4
= (0 + 0 + 2) mod 4
= 2 mod 4
J= 2
Permuter (S [0], S [2])
Fin pour.

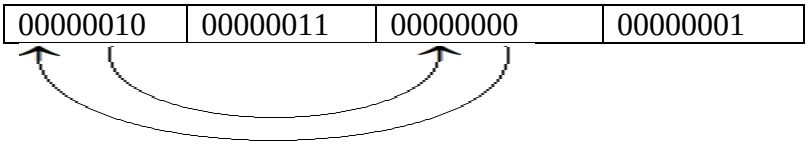
Dans la première itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 0 », avec le contenu de l'index « 2 ».



- **2eme iteration:**

j:= 2
Pour i: = 1
J: = (2 + S [1] + clé [1 mod 4]) mod 4
= (2 + 1 + clé [1]) mod 4
= (2 + 1 + 0) mod 4
= 3mod 4
J: =3
Permuter (S [1], S [3])
Fin pour.

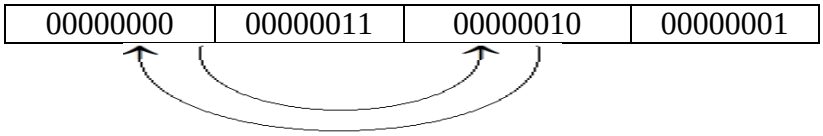
Dans la deuxième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 1 » avec le contenu de l'index « 3 ».



3eme iteration:

j :=3
Pour i = 2
j = (3 + S [2] + clé [2 mod 4]) mod 4
= (3 + 0 +clé [2]) mod 4
= (3 + 0 + 1) mod 4
= 4 mod 4
J=0
Permuter (S [2], S [0])
Fin pour.

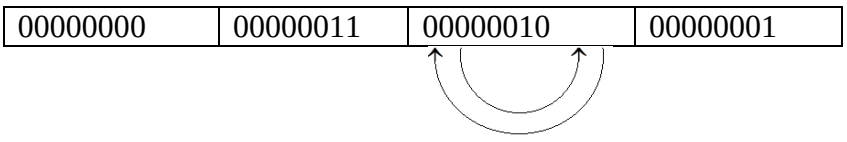
Dans la troisième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 2 », avec le contenu de l'index « 0 »



- **4eme iteration:**

j := 0
Pour i = 3
j = (0 + S [3] + clé [3 mod 4]) mod 4
= (0 + 1 + clé [3]) mod 4
= (0 + 1 + 6) mod 4
= 7 mod 4
J = 3
Permuter (S [3], S [3])
Fin pour.

Dans la quatrième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 3 », avec lui même, rien ne change.



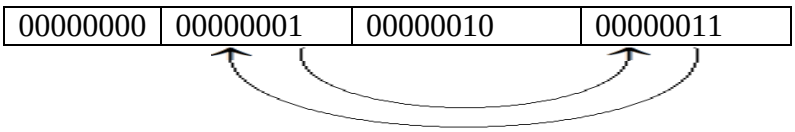
❖ **Etape 3 : générateur d'une suite pseudo aléatoire :**

Chaque octet de message clair va opérer avec un octet générer par cette étape, c'est à dire que si notre message clair à 10caractères, ce générateur va générer 10 octets pour chaque une selon l'algorithme 3 suivant :

i := 0
j := 0
Tant que générer un octet
i := (i + 1) mod 4
j := (j + S [i]) mod 4
permuter (S [i] , S [j])
octet_de_sortie := S [(S [i] + S [j]) mod 4]
Fin tant que.

➢ **1ere iteration:**

i := 0
j := 0
i := (0 + 1) mod 4= 1 mod 4 = 1
j := (0 + S[1]) mod 4 = (0 + 3) mod 4 = 3
Permuter (S [1], S [3])

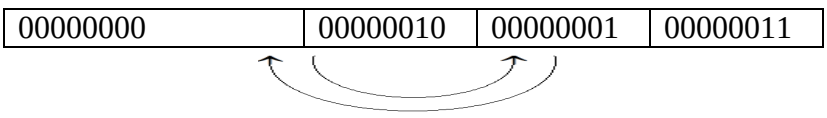


Octet de _sortie = S [(S[i] + S[j]) mod 4]
Octet de sortie = S[(S[1] + S[3]) mod 4] = S[(1 +3)mod4] = S[(4)mod4] =" 00000000"

" 00000000" va crypter le premier octet clair qui est "M"

➢ **2eme iteration:**

i = 1
j = 3
i = (1 + 1) mod 4 = 2 mod 4 = 2
j = (3 + S [2]) mod 4= (3 + 2) mod 4 = 5 mod 4= 1
Permuter (S [2], S [1])



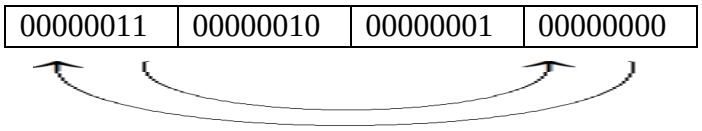
Octet de sortie = S[(S[2] + S[1]) mod 4] = S[(1 +2) mod4] = S[3] =" 00000011"

" 00000011" va crypter le deuxième octet clair qui est "a"

- **3eme iteration:**

i = 2
j = 1

i = (2 + 1) mod 4 = 3 mod 4 = 3
j = (1 + S [3]) mod 4 = (1 + 3) mod 4 = 4 mod 4 = 0
Permuter (S [3], S [0])

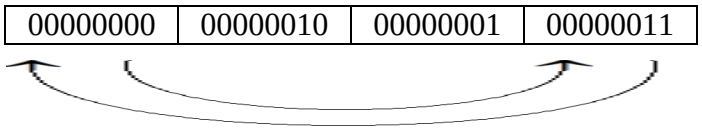


Octet de sortie = S[(S[3] + S[0]) mod 4] = S[0+3) mod4] = S[3] =" 00000000"

" 00000000" va crypter le troisième octet clair qui est "s"

- **4eme iteration:**

i = 3
j = 0
i = (3 + 1) mod 4= 4 mod 4= 0
j = (0 + S [0]) mod 4 = (0 + 3) mod 4 = 3 mod 4 = 3
Permuter (S [0], S [3])

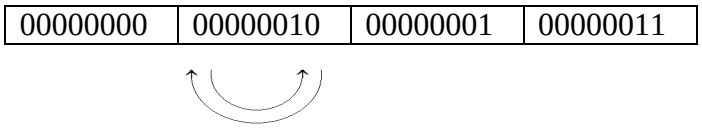


Octet de sortie = S[(S[0] + S[3]) mod 4] = S[0 +3) mod4] = S[3] =" 00000011"

" 00000011" va crypter le quatrième octet clair qui est "t"

- **5eme iteration:**

i = 0
j = 3
i = (0 + 1) mod 4 = 1 mod 4 = 1
j = (3 + S [1]) mod 4 = (3+ 2) mod 4 = 5 mod 4= 1
Permuter (S [1], S [1])



Octet de sortie = S[(S[1] + S[1]) mod 4] = S[2 +2)mod4] = S[0] =" 00000000"

" 00000000" va crypter le cinquième octet clair qui est "e"

- **6eme iteration:**

i = 1
j = 1
i = (1 + 1) mod 4 = 2 mod 4 = 2
j = (1 + S [2]) mod 4 = (1 + 1) mod 4 = 2mod 4 = 2
Permuter (S [2], S [2])

00000000	00000010	00000001	00000011

Octet de sortie = S[(S[2] + S[2]) mod 4] = S[1 +1)mod4] = S[2] =" 00000001"

" 00000001" va crypter le sixième octet clair qui est "r"

- **7eme iteration:**

i = 2
j = 2
i = (2 + 1) mod 4 = 3 mod 4= 3
j = (2 + S [3]) mod 4 = (2 + 3) mod 4 = 5 mod 4= 1
Permuter (S [3], S [1])

00000000	00000011	00000001	00000010

Octet de sortie = S[(S[3] + S[1]) mod 4] = S[2 + 3) mod 4] = S[1] =" 00000011"

" 00000011" va crypter le septième octet clair qui est "2"

- **8eme iteration:**

i = 3
j = 1
i = (3 + 1) mod 4= 4mod 4 = 0
j = (1 + S [0]) mod 4 = (1 + 0) mod 4 = 1 mod 4 = 1
Permuter (S [0], S [1])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[0] + S[1]) mod 4] = S[3 +0) mod4] = S[3] =" 00000010"

" 00000010" va crypter le huitième octet clair qui est "0"

- **9eme iteration:**

i = 0
j = 1
i = (0 + 1) mod 4= 1mod 4 = 1
j = (1 + S [1]) mod 4 = (1 + 0) mod 4 = 1 mod 4 = 1
Permuter (S [1], S [1])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[1] + S[1]) mod 4] = S[0 +0) mod4] = S[0] =" 00000011"

" 00000011" va crypter le huitième octet clair qui est "2"

- **10eme iteration:**

i = 1
j = 1
i = (1+ 1) mod 4= 2mod 4 = 2
j = (1 + S [2]) mod 4 = (1 + 1) mod 4 = 2 mod 4 = 2
Permuter (S [2], S [2])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[2] + S[2]) mod 4] = S[1 +1) mod4] = S[2] =" 00000001"

" 00000001" va crypter le huitième octet clair qui est "1"

La suite des octets pseudo aléatoire de dix octets qui permit de chiffrer le message "Master2021" sont respectivement :

- le premier octet : « 00000000 »
- le deuxième octet : « 00000011 »
- le troisième octet : « 00000000»
- le quatrième octet : « 00000011 »
- le cinquième octet : « 00000000 »
- le sixième octet : « 00000001 »
- le septième octet : « 00000011 »
- le huitième octet : « 00000010 »
- le neuvième octet : « 00000011»
- le dixième octet : « 00000001 »

❖ *Etape 4 le cryptage :*

Enfin, la quatrième et la dernière étape s'agit de crypter le mot « Master2021» avec les dix octets générés précédemment via l'opération XOR :

- le premier octet : « 00000000 » XOR « 01001101 » ⟹ 01001101 « M »
- le deuxième octet : « 00000011 » XOR « 01100001 » ⟹ 01100010 « b »
- le troisième octet : « 00000000» XOR « 01110011 » ⟹ 01110011 « s »
- le quatrième octet : « 00000011 » XOR « 01110100 » ⟹ 01110111 « w »
- le cinquième octet : « 00000000 » XOR « 01100101 » ⟹ 01100101 « e »
- le sixième octet : « 00000001 » XOR « 01110010 » ⟹ 01110011 « s »
- le septième octet : « 00000011 » XOR « 00110010 » ⟹ 00110001 « 1 »
- le huitième octet : « 00000010 » XOR « 00110000 » ⟹ 00110010 « 2 »
- le neuvième octet : « 00000011» XOR « 00110010 » ⟹ 00110001 « 1 »
- le dixième octet : « 00000001 » XOR « 00110001 » ⟹ 00110000 « 0 »

Le message crypter c'est : Mbswes1210

IV.4. Déchiffrement :

Le message chiffré : **Mbswes1210** la clé de déchiffrement c'est **2016**

- **Etape 1 : Initialisation du tableau d'état S :**

Aussi dans cet exemple on suppose que la taille du tableau d'etat est de 4

Pour bien comprendre réellement le principe et le déroulement du fonctionnement de l'algorithme de déchiffrement RC4, on représente les valeurs, ainsi que les opérations en binaire.

Initialisation du tableau d'état selon l'algorithme 1 :

Pour i de 0 à 3

S [i] := i

Fin pour

Le tableau d'état S devient comme suit :

00000000	00000001	00000010	00000011
0	1	2	3

- **Etape 2 : la permutation de tableau d'état S**

Cette étape consiste de mélanger le tableau d'état S avec la clé donnée

Clé [i]	Clé [0]	Clé [1]	Clé [2]	Clé [3]
La clé	2	0	1	6

Tableau IV.3: la clé donnée

Selon l'algorithme 2 suivant :

j := 0

Pour i de 0 à 3

j := (j + S[i] + clé [i mod longueur_clé]) mod 4

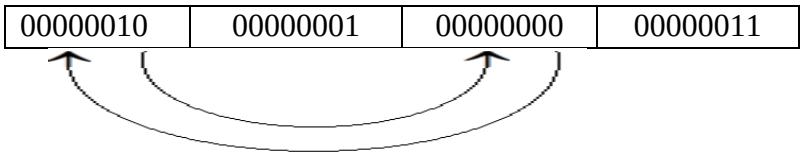
Permuter (S[i], S[j])

Fin pour

- ***1ere iteration:***

```
  j:= 0
Pour i = 0
  j = (0 + S [0] + clé [0 mod 4]) mod 4
     = (0 + 0 + clé [0]) mod 4
     = (0 + 0 + 2) mod 4
     = 2 mod 4
   J= 2
  Permuter (S [0], S [2])
Fin pour.
```

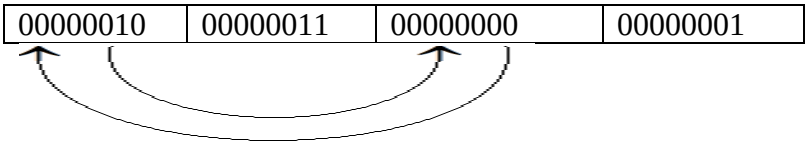
Dans la première itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 0 », avec le contenu de l'index « 2 ».



- **2eme iteration:**

```
   j:= 2
Pour i: = 1
   J: = (2 + S [1] + clé [1 mod 4]) mod 4
     = (2 + 1 + clé [1]) mod 4
     = (2 + 1 + 0) mod 4
     = 3mod 4
   J: =3
  Permuter (S [1], S [3])
Fin pour.
```

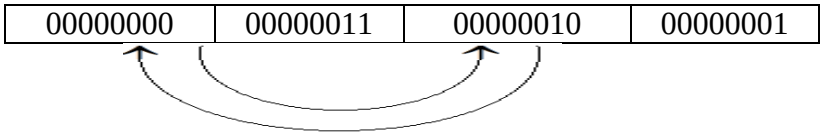
Dans la deuxième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 1 » avec le contenu de l'index « 3 ».



3eme iteration:

j :=3
Pour i = 2
j = (3 + S [2] + clé [2 mod 4]) mod 4
= (3 + 0 +clé [2]) mod 4
= (3 + 0 + 1) mod 4
= 4 mod 4
J=0
Permuter (S [2], S [0])
Fin pour.

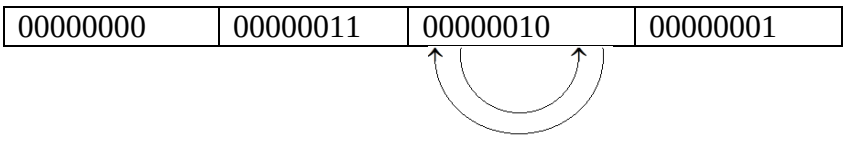
Dans la troisième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 2 », avec le contenu de l'index « 0 »



- **4eme iteration:**

j := 0
Pour i = 3
j = (0 + S [3] + clé [3 mod 4]) mod 4
= (0 + 1 + clé [3]) mod 4
= (0 + 1 + 6) mod 4
= 7 mod 4
J = 3
Permuter (S [3], S [3])
Fin pour.

Dans la quatrième itération, le tableau d'état permutera le contenu qui se trouve dans l'index « 3 », avec lui même, rien ne change.



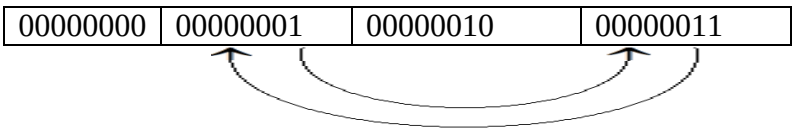
❖ **Etape 3 : générateur d'une suite pseudo aléatoire :**

Chaque octet de message clair va opérer avec un octet générer par cette étape, c'est à dire que si notre message clair à 10caractères, ce générateur va générer 10 octets pour chaque une selon l'algorithme 3 suivant :

i := 0
j := 0
Tant que générer un octet
i := (i + 1) mod 4
j := (j + S [i]) mod 4
permuter (S [i] , S [j])
octet_de_sortie := S [(S [i] + S [j]) mod 4]
Fin tant que.

➢ **1ere iteration:**

i := 0
j := 0
i := (0 + 1) mod 4= 1 mod 4 = 1
j := (0 + S[1]) mod 4 = (0 + 3) mod 4 = 3
Permuter (S [1], S [3])

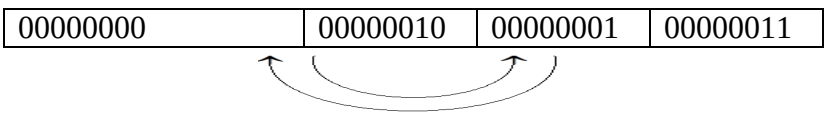


Octet de _sortie = S [(S[i] + S[j]) mod 4]
Octet de sortie = S[(S[1] + S[3]) mod 4] = S[(1 +3)mod4] = S[(4)mod4] =" 00000000"

" 00000000" va crypter le premier octet clair qui est "M"

➢ **2eme iteration:**

i = 1
j = 3
i = (1 + 1) mod 4 = 2 mod 4 = 2
j = (3 + S [2]) mod 4= (3 + 2) mod 4 = 5 mod 4= 1
Permuter (S [2], S [1])



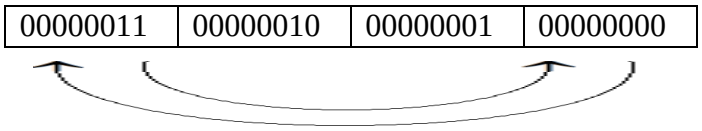
Octet de sortie = S[(S[2] + S[1]) mod 4] = S[(1 +2) mod4] = S[3] =" 00000011"

" 00000011" va crypter le deuxième octet clair qui est "a"

- **3eme iteration:**

i = 2
j = 1

i = (2 + 1) mod 4 = 3 mod 4 = 3
j = (1 + S [3]) mod 4 = (1 + 3) mod 4 = 4 mod 4 = 0
Permuter (S [3], S [0])

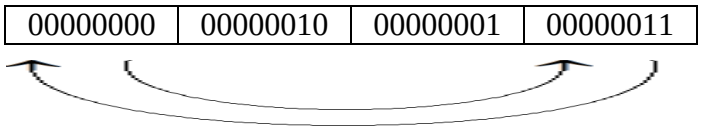


Octet de sortie = S[(S[3] + S[0]) mod 4] = S[0+3) mod4] = S[3] =" 00000000"

" 00000000" va crypter le troisième octet clair qui est "s"

- **4eme iteration:**

i = 3
j = 0
i = (3 + 1) mod 4= 4 mod 4= 0
j = (0 + S [0]) mod 4 = (0 + 3) mod 4 = 3 mod 4 = 3
Permuter (S [0], S [3])

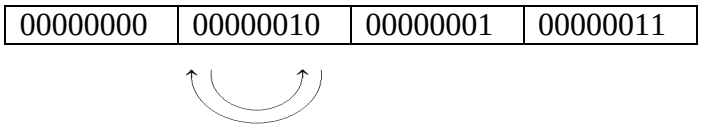


Octet de sortie = S[(S[0] + S[3]) mod 4] = S[0 +3) mod4] = S[3] =" 00000011"

" 00000011" va crypter le quatrième octet clair qui est "t"

- **5eme iteration:**

i = 0
j = 3
i = (0 + 1) mod 4 = 1 mod 4 = 1
j = (3 + S [1]) mod 4 = (3+ 2) mod 4 = 5 mod 4= 1
Permuter (S [1], S [1])



Octet de sortie = S[(S[1] + S[1]) mod 4] = S[2 +2)mod4] = S[0] =" 00000000"

" 00000000" va crypter le cinquième octet clair qui est "e"

- **6eme iteration:**

i = 1
j = 1
i = (1 + 1) mod 4 = 2 mod 4 = 2
j = (1 + S [2]) mod 4 = (1 + 1) mod 4 = 2mod 4 = 2
Permuter (S [2], S [2])

00000000	00000010	00000001	00000011

Octet de sortie = S[(S[2] + S[2]) mod 4] = S[1 +1)mod4] = S[2] =" 00000001"

" 00000001" va crypter le sixième octet clair qui est "r"

- **7eme iteration:**

i = 2
j = 2
i = (2 + 1) mod 4 = 3 mod 4= 3
j = (2 + S [3]) mod 4 = (2 + 3) mod 4 = 5 mod 4= 1
Permuter (S [3], S [1])

00000000	00000011	00000001	00000010

Octet de sortie = S[(S[3] + S[1]) mod 4] = S[2 + 3) mod 4] = S[1] =" 00000011"

" 00000011" va crypter le septième octet clair qui est "2"

- **8eme iteration:**

i = 3
j = 1
i = (3 + 1) mod 4= 4mod 4 = 0
j = (1 + S [0]) mod 4 = (1 + 0) mod 4 = 1 mod 4 = 1
Permuter (S [0], S [1])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[0] + S[1]) mod 4] = S[3 +0) mod4] = S[3] =" 00000010"

" 00000010" va crypter le huitième octet clair qui est "0"

- **9eme iteration:**

i = 0
j = 1
i = (0 + 1) mod 4= 1mod 4 = 1
j = (1 + S [1]) mod 4 = (1 + 0) mod 4 = 1 mod 4 = 1
Permuter (S [1], S [1])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[1] + S[1]) mod 4] = S[0 +0) mod4] = S[0] =" 00000011"

" 00000011" va crypter le huitième octet clair qui est "2"

- **10eme iteration:**

i = 1
j = 1
i = (1+ 1) mod 4= 2mod 4 = 2
j = (1 + S [2]) mod 4 = (1 + 1) mod 4 = 2 mod 4 = 2
Permuter (S [2], S [2])

00000011	00000000	00000001	00000010

Octet de sortie = S[(S[2] + S[2]) mod 4] = S[1 +1) mod4] = S[2] =" 00000001"

" 00000001" va crypter le huitième octet clair qui est "1"

La suite des octets pseudo aléatoire de dix octets qui permit de chiffrer le message "Master2021" sont respectivement :

- le premier octet : « 00000000 »
- le deuxième octet : « 00000011 »
- le troisième octet : « 00000000»
- le quatrième octet : « 00000011 »
- le cinquième octet : « 00000000 »
- le sixième octet : « 00000001 »
- le septième octet : « 00000011 »
- le huitième octet : « 00000010 »
- le neuvième octet : « 00000011»
- le dixième octet : « 00000001 »

- **Etape 4 le décryptage :**

Enfin, la quatrième et la dernière étape s'agit de decrypter le mot « Mbswes» avec les dix octets générés précédemment via l'opération XOR :

- le premier octet : « 00000000 » XOR « 01001101 » ⟹ 01001101 « M »
- le deuxième octet : « 00000011 » XOR « 01100010 » ⟹ 01100001 « a»
- le troisième octet : « 00000000» XOR « 01110011 » ⟹ 01110011 « s »
- le quatrième octet : « 00000011 » XOR « 01110111 » ⟹ 01110100 « t»
- le cinquième octet : « 00000000 » XOR « 01100101 » ⟹ 01100101 « e »
- le sixième octet : « 00000001 » XOR « 01110011 » ⟹ 01110010 « r »
- le septième octet : « 00000011 » XOR « 00110001 » ⟹ 00110010 « 2 »
- le huitième octet : « 00000010 » XOR « 00110010» ⟹ 00110000 « 0 »
- le neuvième octet : « 00000011» XOR « 00110001 » ⟹ 00110010 « 2 »
- le dixième octet : « 00000001 » XOR « 00110000» ⟹ 00110001 « 1 »

Le message décrypter c'est : Master2021

IV.5. La simulation sous MATLAB :

MATLAB est un langage de programmation de quatrième génération et un environnement de développement ; il est utilisé à des fins de calcul numérique. Développé par la société The Math Works, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes, de créer des interfaces utilisateurs, et peut s'interfacer avec d'autres langages comme le C, C++, Java, et Fortran. Les utilisateurs de MATLAB (environ un million en 2001) sont de milieux très différents comme l'ingénierie, les sciences et l'économie dans un contexte aussi bien industriel que pour la recherche.

IV.6. Le cryptage d'une image avec le RC4 :

Le cryptage d'une image avec le RC4 a le même principe que le message, mais l'image plus compliqué que le message car l'image utilise les pixels, un pixel a 24bit (3octets = RGB), on a réalisé une application de cryptage d'une image sous MATLAB.

IV.7. Le décryptage d'une image avec le RC4 :

Le décryptage d'une image est le même que le chiffrement sauf qu'on prend l'image chiffré au lieu d'image clair et la clé de chiffrement/déchiffrement.

IV.8. le cryptage et décryptage des messages (textes) :

- le code de cryptage de mot " cryptage "

```
Clear all; clc

l = 2048/8;
%K = randi(256,[1,l],'uint8')-1;

K = [2 0 1 6];
l = length(K);

%initialisation du tableau S

S = zeros(1,256);
for i=0:255
    S(i+1) = i;
end

j=0;
for i=0:255
    j = mod((j + S(i+1) + K(mod(i,l)+1)),256);

    %permutation : echanger S(i) et S(j)

       aux = S(i+1);
    S(i+1) = S(j+1);
    S(j+1) = aux;
end

%Production de la suite chiffrante

i = 0;
j = 0;
C = 0;

txt = ' cryptage '

n = length(txt);

for r=1:n
```

```
        i = mod(i + 1,256);
        j = mod(j + S(i+1),256);

        aux = S(i+1);
        S(i+1) = S(j+1);
        S(j+1) = aux;

        C = S(mod(S(i+1)+S(j+1),256)+1);

        txt(r) = bitxor(uint8(txt(r)),uint8(C));

    end

txt

f = fopen('txt_crypt.txt','w');

fprintf(f,'%s',txt);

fclose(f);
```

- le cryptage de mot "cryptage" :

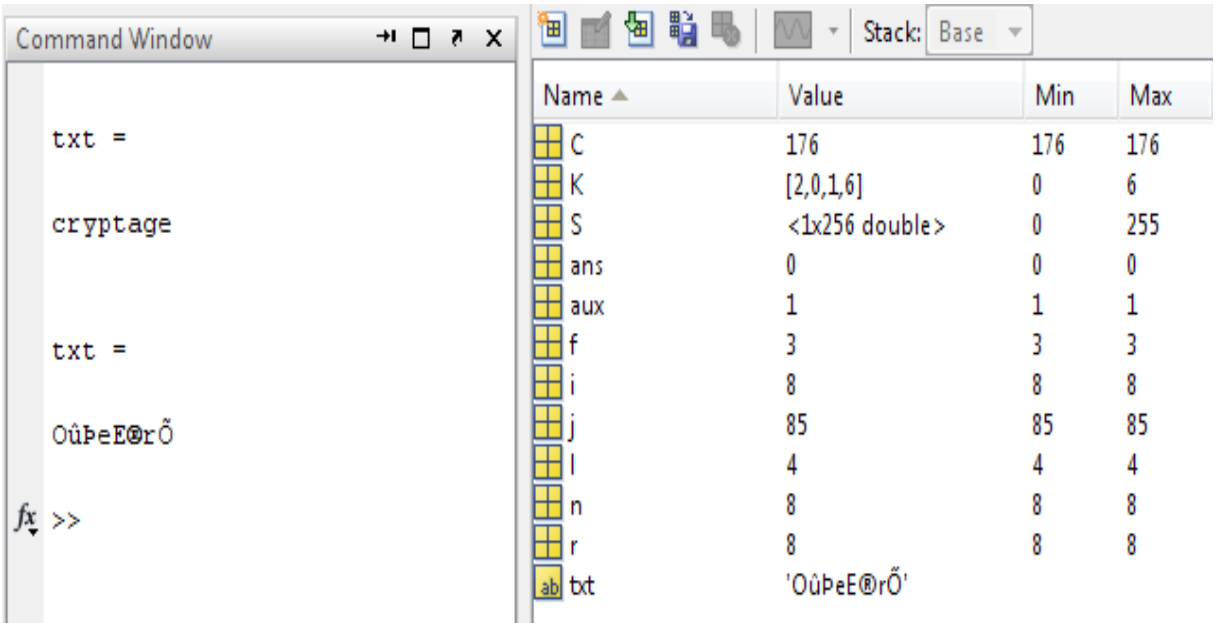


Figure IV.1: le cryptage de mot "cryptage"

- le décryptage de mot "cryptage" :

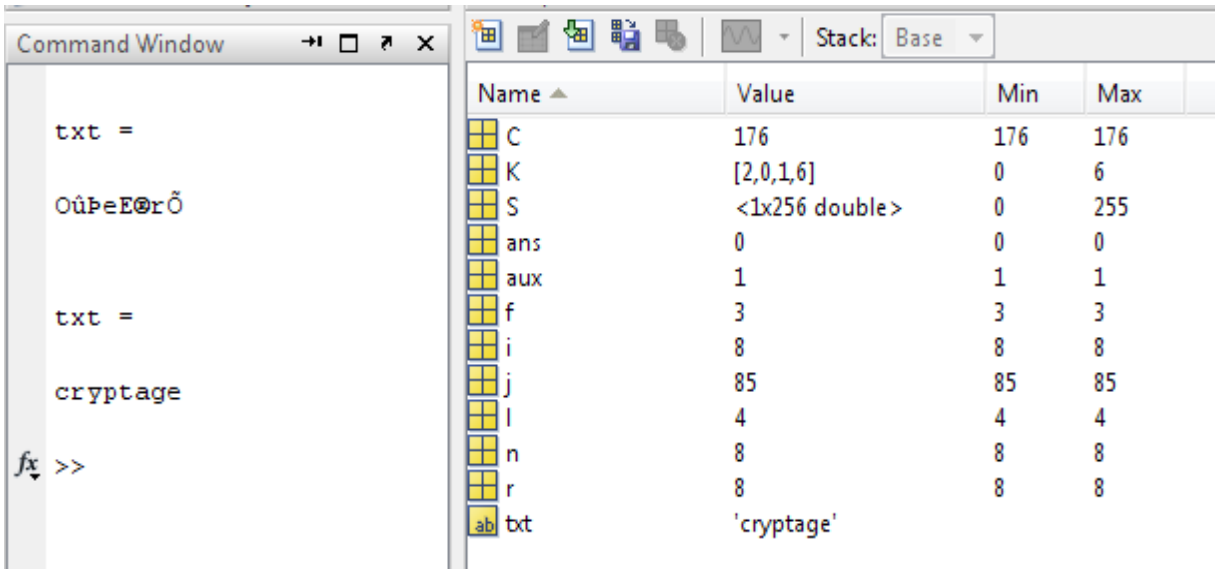


Figure IV.2: le décryptage de mot "cryptage"

- le cryptage de mot "Master avec la clé 2021" :

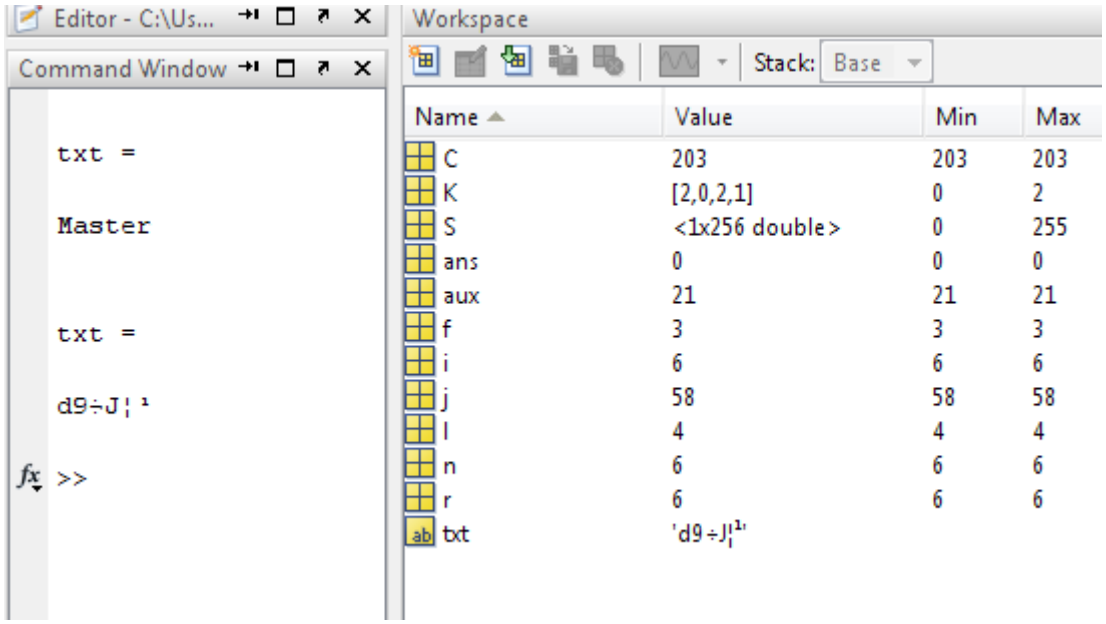


Figure IV.3 : le cryptage de mot "Master"

- le décryptage de mot "Master2021" :

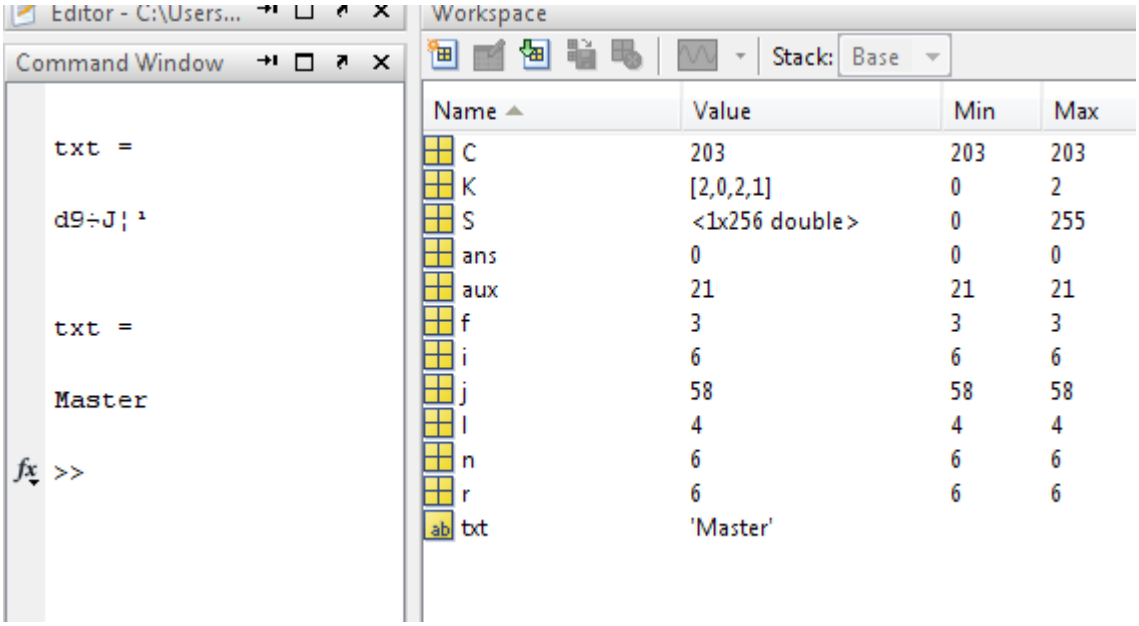


Figure IV.4 : le décryptage de mot "Master"

IV.9.le cryptage et décryptage des images :

- Le cryptage d'une image, alors notre image en clair est :

Figure IV.5 : l'image d'un réseau en clair

La clé secrète de cryptage : 2 0 1 6

L’image chiffrée :

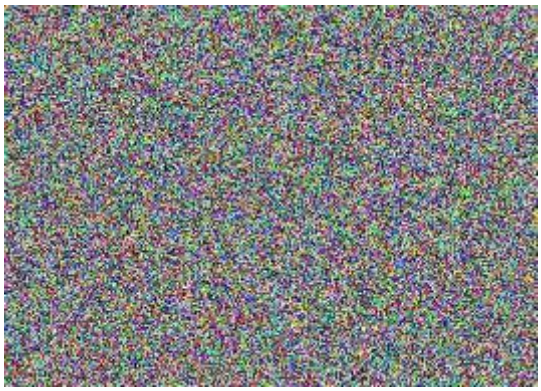


Figure IV.6: l'image d’un réseau crypté

Le décryptage est l'inverse de cryptage, l'image cryptée avec la clé, en utilisant la clé secrète, nous obtenons l'image claire.

- Le décryptage de l’image chiffré :

Figure IV.7: l'image d’un réseau déchiffrée

- Le cryptage d'une image en clair est :

Figure IV.8 : l'image d’un paysage en clair

La clé secrète de cryptage : 2 0 1 6

L’image chiffrée :

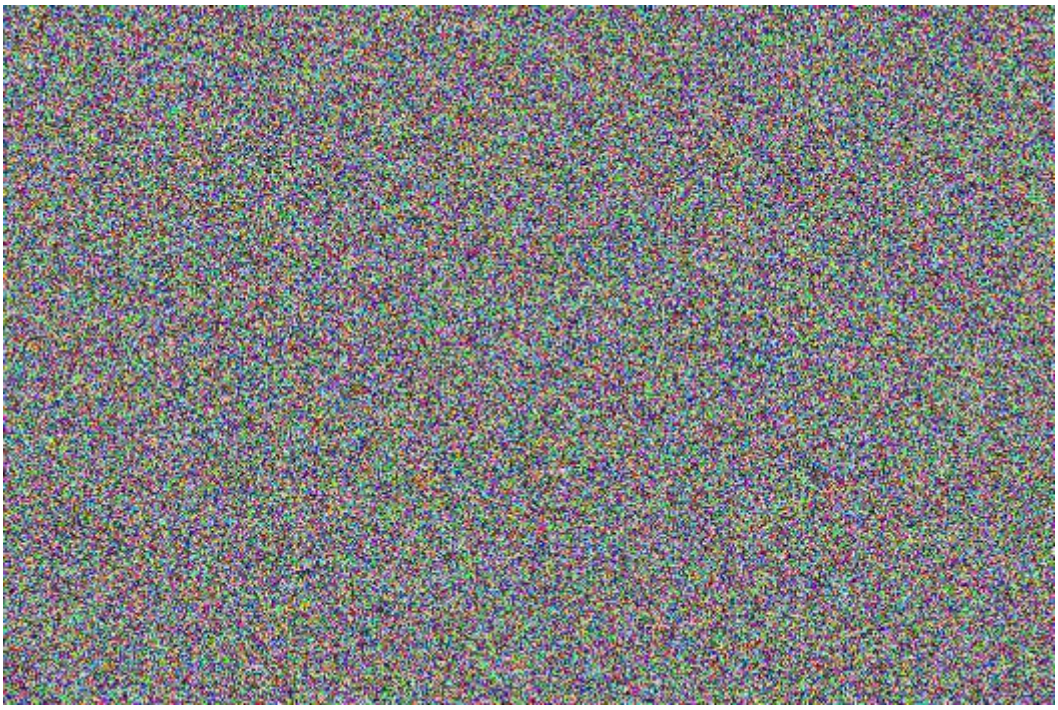


Figure IV.9: l'image d’un paysage crypté

Le décryptage est l'inverse de cryptage, l'image cryptée avec la clé, en utilisant la clé secrète, nous obtenons l'image claire.

- Le décryptage de l'image chiffré :

Figure IV.10: l'image d'un paysage dé**crypté**

Conclusion

Dans ce chapitre, nous avons fait une étude et simulation du cryptosystème symétrique par flot RC4, cette étude et simulation nous a permis de maitriser l'algorithme RC4 sur des exemples concrets, le cryptage et décryptage des messages (textes), des images.

Conclusion générale :

Conclusion générale :

Le développement de l'utilisation des réseaux a mis la sécurité au premier plan. Le point principal de la sécurisé des réseaux consiste à connaitre ses faiblesses et ses limites pour les surveiller particulièrement.

La cryptographie est utilisée pour dissimuler des données à certains utilisateurs et garantir que seuls les destinataires légitimes auront la possibilité de les consulter, donc l'utilisateur de la cryptographie doit savoir utiliser les méthodes cryptographiques pour garantir cette notion.

Dans ce travail de mémoire de master, nous avons commencé par une introduction général, ensuite nous avons décrit certains algorithmes de chiffrement classique tels que celui de Vigenère et César, et ceux de chiffrement moderne tels que DES, Diffie-Hellman, AES, RC4.

Le chapitre 4 a été dévolu à l'étude détaillée de l'algorithme de chiffrement par flot le rc4 avec des exemples de chiffrement et déchiffrement de texte et d'images avec des simulations sous le logiciel MATLAB.

Nous suggérons la continuation de ce travail par une implémentation de ce cryptosystème sur FPGA et un chiffrement et déchiffrement d'une vidéo et d'étudier en détail la cryptographie quantique et leurs applications dans le domaine pratique, afin de la combiner avec le RC4.

La cryptographie en dépit de son antiquité et de son importante évolution, de la cryptographie classique (code de César et autres) à la cryptographie moderne (DES, AES, RSA et autre) à la cryptographie quantique, elle reste toujours en développement.

Les résultats de simulationpar logiciel Matlab sur des données et des images montrent que le cryptosystème RC4 analysé est d'un interet certain et qu'il faut améliorer d'avantage en auguementant la taille de la clé de chiffrement.

Bibliographie :

[1] MARIUS Lulian « pro cryptographie and cryptanalysis with C++ 20 »

[2] THOMAS BAIGNKRES ,PASCAL JUNOD ,YI LU ,JEAN MONNERAT ET SERGE VAUDENAY ," a classical introduction to cryptography exercise book" - 2006

[3] DIDIER MÜLLER," Une application intéressante des matrices : le chiffre de HILL"- 2002

[4] DELFS H, KNEBL H « introduction to cryptographie » 2002

[5] JEFFRY Hoffstein, JILL Pipher , JOSEPH H. Silverman "An introduction to mathematical cryptography" Springer 2008

[6] NGUY EN PHONG QUANG ," Théorie et Pratique de la Cryptanalyse à Clé Publique" 2007

[7] HOAI MINH, "Modélisation et Optimisation non convexe basées sur la programmation DC et DCA pour la résolution de certaines classes des problèmes en Fouille de Données et Cryptologie". 2007

[8] Aicha TEKKOUK, « Etude et Implémentation d'une méthode cryptanalyse pour le chiffrement continu » Mémoire du Magister, Université des sciences et de la technologie d'Oran, 2010

[9] Nadia EL MRABET « le concept fondamentaux de la cryptographie greyc » Polycopié de cours université de Cean France 2010

[10] Melle.BEN AMMAR Asma Melle .HADDOUCHE Khalissa, « Amélioration de la génération des sous clés de l'algorithme cryptographique DES » Mémoire de Master, Université Akli Mohend OULHADJ-Bouira, 2017

[11] Mohamed BOUTORA Djouher BEN AMI, «Conception, Etude et Réalisation d'un Cryptosystème Hybride de Transmission d'Images. » Mémoire de fin d'études de Master Académique Université Mouloud MAMMERI TIZI-OUZOU, 2015

[12] S. BEKHOUCH, " Fondements mathématiques et fonctionnement du standard de chiffrement avancé Rijndael (AES) ", Mémoire de Magister, Université des Sciences et de la Technologie Houari Boumediene, (Algérie), 2006.

[13] FELLAH ouahiba, « Etude et simulation du crypto système RC4. » Mémoire de fin d'études pour l'obtention du diplôme de Master Université M'HAMED BOUGUERRA de Boumerdes, 2013

[14] Allou Said; Allouane Kahina, « Cryptographie et sécurité des Réseaux Implémentation de l'AES sous MATLAB » Mémoire de fin d'études présenté en vue de l'obtentin du diplôme d'Ingénieur d'Etat en Electronique , Université Mouloud MAMMERI, TIZI-OUZOU, 2008.

[15] S. BOUALLAGUI, "Techniques d'optimisation déterministe et stochastique pour la résolution de problèmes difficiles en cryptologie ", Thèse de Doctorat, Institut national des sciences appliquées de ROUEN, 05. Juillet. 2010.

[16] ram-0000 « introduction à la cryptographie » article publié le 8 janvier 2009

[17] M. VIDEAU, "Critère de sécurité des algorithmes de chiffrement à clé secrète ", Thèse de Doctorat, Université de Paris 6, (France), 10. Novembre. 2005.

[18] Niels FERGUSON ;Bruce SCHNEIER ; Tadayoshi KOHNO « Cryptography Engineering Design Principales and Practical Applications", Wiley Publishing 2010

[19] Sarah Haddad "Implémentation sur circuit programmable de type FPGA de la multiplication modulaire pour le protocole cryptographie RSA" mémoire master2 UNIVERSITE M'HAMMED BOUGARA Faculté Des Sciences 2013.

ANNEXE A :

Le Programme de Chiffrement Et déchiffrement de texte

✸ Le programme de chiffrement et déchiffrement de texte :

```
Clear all;

clc;

l = 2048/8;

%K = randi(256,[1,l],'uint8')-1;

K = [2 0 2 1];

l = length(K);

%initialisation du tableau S

S = zeros(1,256);

for i=0:255

   S(i+1) = i;

End.

j=0;

for i=0:255

   j = mod((j + S(i+1) + K(mod(i,l)+1)),256);

   %permutation : echanger S(i) et S(j)

   aux = S(i+1);

   S(i+1) = S(j+1);

   S(j+1) = aux;

end

%Production de la suite chiffrante

i = 0;
```

```
j = 0;

C = 0;

txt = 'Master'


n = length(txt);

for r=1:n

        i = mod(i + 1,256);

        j = mod(j + S(i+1),256);

        aux = S(i+1);

        S(i+1) = S(j+1);

        S(j+1) = aux;

        C = S(mod(S(i+1)+S(j+1),256)+1);

        txt(r) = bitxor(uint8(txt(r)),uint8(C));

    end

txt

f = fopen('txt_crypt.txt','w');

fprintf(f,'%s',txt);

fclose(f);
```

ANNEXE B:

Le programme de chiffrement Et déchiffrement d'une image

✸ Le code de chiffrement et déchiffrement d'une image:

```
clear all;
clc;

l = 2048/8;
%K = randi(256,[1,l],'uint8')-1;

K = [2 0 1 6];
l = length(K);

%initialisation du tableau S

S = zeros(1,256);
for i=0:255
   S(i+1) = i;
end

j=0;
for i=0:255
   j = mod((j + S(i+1) + K(mod(i,l)+1)),256);

   %permutation : echanger S(i) et S(j)
   aux = S(i+1);
   S(i+1) = S(j+1);
   S(j+1) = aux;
```

```
end



%Production de la suite chiffrante


i = 0;
j = 0;
C = 0;


img = imread('Arbre.png');
img1 = img;


imshow(img);


n = size(img,1);
m = size(img,2);


for r=1:n
   for p=1:m


      i = mod(i + 1,256);
      j = mod(j + S(i+1),256);


      aux = S(i+1);
      S(i+1) = S(j+1);
      S(j+1) = aux;
```

```
    C = S(mod(S(i+1)+S(j+1),256)+1);

    img(r,p,1) = bitxor(uint8(img(r,p,1)),uint8(C));

    i = mod(i + 1,256);
    j = mod(j + S(i+1),256);

    aux = S(i+1);
    S(i+1) = S(j+1);
    S(j+1) = aux;

    C = S(mod(S(i+1)+S(j+1),256)+1);

    img(r,p,2) = bitxor(uint8(img(r,p,2)),uint8(C));

    i = mod(i + 1,256);
    j = mod(j + S(i+1),256);

    aux = S(i+1);
    S(i+1) = S(j+1);
    S(j+1) = aux;

    C = S(mod(S(i+1)+S(j+1),256)+1);

    img(r,p,3) = bitxor(uint8(img(r,p,3)),uint8(C));
end
```

```
end

figure

imshow(img);


imwrite(img,'Arbre cry.png');
```

ANNEXE C:

Arithmétique modulaire :

Notion de nombre premier :

Soit p un entier positif, **p ≥ 2**. Ont dit que **p** est premier si :

- Ses diviseurs positifs sont **1** et **p**.

Soit **p** un nombre premier et **a, b** deux entiers tels que :

- **p/ab**, Alors **p/a** ou **p/b**.

Fonction d'Euler:

Fonction d'Euler : C'est le nombre d'éléments qui sont premiers avec n

Soit **n** un nombre entier, on appelle fonction d'Euler le nombre **φ(n)** tel que :

$$\mathbf{n= \prod_i p_i^{\alpha_i}}$$

$$\boldsymbol{\varphi}(\mathbf{n})=\prod_{\mathbf{i}}(\mathbf{p_i}-\mathbf{1})\ \mathbf{p_i^{\alpha_{i-1}}}$$

D'une façon générale:

- Si n est premier, **φ(n)= n-1.**
- Si n est le produit de deux nombres p et q premiers, alors : **φ(n)= (p-1) × (q-1).**

Théorème d'Euler :

Le théorème d'Euler est défini comme suit.

Si **a** et **n** sont premiers entres eux, ces derniers sont liés par l'expression suivante :

$$a^{\varphi(n)} = 1\ mod\ n$$

PETIT THEOREME DE FERMAT :

Soit **a** et **p**, premier entre eux.

$$a^{p-1} \equiv 1[p]$$

L'épreuve :

- Si **(a, p) = 1** ⟹ $a^{\alpha(p)} \equiv 1[p]$
- Si **p** premier [condition de Fermat] ⟹ **) = p-1.**

$$a^{p-1} \equiv 1[p]$$

Inversion modulaire :

- **a** et **b** sont inverse si : **a*b** $\equiv 1[n]$

Théorème d'Euclide :

Soit **a, b** $\in$ **Z** tel que **b** $\neq$ **0** et **r** le reste de la division euclidienne de **a** par **b**. Alors

PGCD (a, b) = PGCD (b, r).

L'algorithme d'Euclide consiste alors à réitéré les manipulations suivantes :

• Effectuer la division Euclidienne de **a** par **b.** soit **r** le reste.

• Remplacer **a** par **b** et **b** par **r** (on a donc **0 ≤ r < b** d'après la définition de la division euclidienne.

- Le PGCD est le dernier reste nul.

La complexité de l'algorithme d'Euclide réside dans le cas ou a > b.

Division Euclidienne:

Soit **a** et **b** deux entiers tels que **b** $\neq$ **0**. Il existe un unique couple d'entiers **(q, r)** tels que :

a = bq + r et **0 ≤ r > b.**

On appelle **q** le quotient de **a** par **b** et **r** le reste de la division de **a** par b.

Théorème de Bézout :

Soient **a** et b deux entiers relatifs. Si **PGCD (a, b) = d** alors $\exists\ (x, y) \in \mathbb{Z}^*$ tel que **d = ax + by**

Identité de Bézout :

Deux entiers relatifs ***a*** et ***b*** sont premiers entre eux si et seulement s'il existe deux entiers relatifs ***x*** et ***y*** tels que : ***ax* + *by* = 1.**

Théorème de Gauss:

Soient (a, b, c) ∈ $\mathbb{Z}^3$

- Si **PGCD (a, b) = 1** et **a/bc** alors **a/c.**