MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU
FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE
DEPARTEMENT D'ELECTRONIQUE

Mémoire de fin d'études

Présenté en vue de l'obtention
du Diplôme d'Ingénieur d'Etat en Electronique

Option : Communication-Contrôle

Thème:

Cryptographie et sécurité des Réseaux Implémentation de l'AES sous MATLAB

Proposé et dirigé par :

M^{r} : Ziani Rezki

Conçu et réalisé par :

M^{r}: Allou Said
M$^{r.}$: Allouane Kahina

Promotion **2008**

Remerciement

Nous tenons avant tout de remercier le bon DIEU qui nous a donné la volonté et le courage pour réaliser ce modeste travail.

Du fond du cœur nous remercions nos chers parents qui nous ont toujours guidé, encouragé et qui ont fait de leurs mieux pour que nous arrivons là aujourd'hui.

Nous remercions vivement Mr ZIANI REZKI notre promoteur pour la précieuse assistance, sa disponibilité et son soutien qu'il nous à apporté tout au long de ce projet.

Que les membres de jury trouvent ici l'expression de nos sincères remerciements pour l'honneur qu'ils nous font en acceptant de juger notre travail.

Nous tenons aussi à remercier Mr M .LAHDIR ainsi que Mr M. LAZRI pour leurs aides et leurs assistantes au cours de ce projet.

Que cette page soit aussi le parfait témoignage de notre gratitude envers Mr S. LESLOUS pour son aide et à tous nos amis qui nous ont aidé, de près ou de loin, à élaborer ce travail.

Dédicaces

A mon père et ma mère,
A mes frères et sœurs,
A tous ceux qui sont chers,
Je dédie ce travail.

Kahina

A mes chers parents,
A mes frères et sœur,
A touts mes ami(e)s,
A ceux qui m'aiment,
Je dédie ce travail.

Saïd

SOMMAIRE

Chapitre II : Algorithmes de cryptage

Chapitre III : Description et programmation de l'AES

Chapitre IV : Application

INTRODUCTION GENERALE

Depuis l'antiquité l'homme est 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 informatique, accès non autorisé, la non identification et la fausse information…

Pour faire face à ces différentes menaces il faut mettre en œuvre un système de sécurité adéquat et robuste qui répond aux exigences et aux aspirations d'une politique de sécurité.

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…

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

Dans notre travail nous avons étudié les vulnérabilités qui mettent en péril la sécurité des réseaux et nous avons proposé des mesures de protection jugées importantes dont la plus essentielle est la cryptographie, particulièrement l'algorithme AES.

Pour cela nous avons structuré notre travail comme suit :

Le premier chapitre comporte un ensemble de notions et de généralités sur la sécurité des réseaux et la cryptologie. Un état de l'art de différents algorithmes de cryptographie existants dans le deuxième chapitre. Dans le troisième chapitre nous décrivons le fonctionnement ainsi que la programmation de l'algorithme AES et une application dans le quatrième chapitre. En fin, nous terminons avec une conclusion générale.

I. Généralités sur la sécurité :

I.1.Introduction :

La sécurité des systèmes informatiques et télécommunications vise à protéger l'accès et la manipulation des données et des ressources d'un système par des mécanismes d'authentification, d'autorisation et de contrôle d'accès, etc.

Néanmoins, avec l'ouverture et l'interconnexion des systèmes informatiques, des attaques exploitant les failles de ces systèmes et contournant leurs mécanismes de sécurité sont toujours possibles. Il n'est donc pas suffisant d'agir préventivement, c'est-à-dire de définir une politique de sécurité (en termes de confidentialité, d'intégrité et de disponibilité des données et ressources du système à protéger) et de mettre en œuvre des mécanismes implantant cette politique. Il faut aussi être capable de détecter toute tentative de violation de la politique de sécurité, c'est-à-dire toute intrusion.

Nous définissons dans ce chapitre le terme de sécurité des réseaux ainsi que les méthodes utilisées.

I.2 Qu'est-ce que la sécurité d'un réseau ?

La sécurité d'un réseau est un niveau de garantie que l'ensemble des machines du réseau fonctionnent de façon optimale et que les utilisateurs autorisés possèdent uniquement les droits qui leur ont été octroyés.

Il peut s'agir :

- ✓ D'empêcher des personnes non autorisées d'agir sur le système de façon malveillante.
- ✓ D'empêcher les utilisateurs d'effectuer des opérations involontaires capables de nuire au système.
- ✓ De sécuriser les données en prévoyant les pannes.
- ✓ De garantir la non interruption d'un service.

I. 3. Les menaces : [1]

I .3. 1. Description de la menace :

Action ou événement susceptible de se produire, de se transformer en agression contre un environnement ou des ressources et de porter préjudice a leur sécurité. D'où la menace porte atteinte a la confidentialité, l'intégrité ou la disponibilité d'un actif en exploitant une faiblesse d'un système d'une manière accidentelle ou intentionnelle.

I. 3. 2. Catégories de menaces :

- **Divulgation :**

Utilisation non autorisée des ressources du système d'information, entraînant la communication d'informations confidentielles à des tiers.

- **Interruption :**

Toute menace s'attaquant a la disponibilité d'une information ou d'un service (panne de courant, inondation, etc.).

- **Modification :**

Atteinte et altération de l'intégrité des données pendant la communication ou le stockage (Erreur d'entretien de données, piratage informatique,…).

- **Destruction :**

Tout ce qui participe à l'altération des données ou du système d'information (incendie, tremblement de terre, …).

- **Enlèvement :**

Englobe tout ce qui concerne le vol des données ou du système.

- **Répudiation :**

Les menaces de cette catégorie émanent d'utilisateurs qui nient avoir effectué une action, les autres parties étant dans l'incapacité de prouver le contraire.

I.4. Les risques:

Un risque est un danger éventuel plus ou moins prévisible. Il se mesure à la probabilité qu'il se produise, aux impacts et dommages consécutifs à sa réalisation. Il exprime la probabilité qu'une valeur soit perdue en fonction d'une vulnérabilité liée à une menace, ou à un danger.

I. 5. Les mesures de sécurité : [1]

Toutes les sphères d'activité sont concernées par la sécurité du système d'information. En fonction de son domaine d'application la sécurité se distingue en :

- Sécurité physique,
- Sécurité logique,
- Sécurité d'exploitation,
- Sécurité applicative,
- Sécurité des télécommunications.

I. 5.1 Sécurité physique:

Comme son nom l'indique, la sécurité physique concerne tous les aspects liés à la maîtrise des systèmes et de l'environnement dans lesquels ils se situent. Pour garantir la

disponibilité des ressources vitales du réseau, on applique les moyens de protection suivants :

- ✓ La protection des sources énergétiques (alimentation…),
- ✓ Les normes de sécurité,
- ✓ La protection de l'environnement (incendie, humidité, température,…),
- ✓ La limitation des accès physiques et l'utilisation de surveillance environnementale appropriée,
- ✓ La sûreté de fonctionnement des matériels (composants, câble,..).

I .5. 2 Sécurité de l'exploitation:

On entend par la sécurité d'exploitation tout ce qui touche au bon fonctionnement des systèmes. Cela comprend la mise en place d'outils et de procédures relatives aux méthodologies de l'exploitation, de maintenance, de test, de diagnostic et de mise à jour. En particulier, la sécurité d'exploitation dépend fortement de son degré d'industrialisation, qui est qualifié par le niveau de supervision des applications et d'automatisation des taches. Bien que relevant de la responsabilité de l'exploitation, ces conditions concernent très directement la conception et la réalisation des applications elles mêmes.

Les points clés de la sécurité de l'exploitation sont les suivants.

- ✓ Gestion des configurations et des mises à jour ;
- ✓ Automatisation, contrôle et sécurité de l'exploitation;
- ✓ Plan de sauvegarde.

I. 5. 3. Sécurité logique:

La sécurité logique fait référence à la réalisation des mécanismes de sécurité par logiciel. Elle repose principalement sur une mise en œuvre adéquate d'un processus de contrôle d'accès logique, lequel s'appuie sur un triple service d'identification, d'authentification et d'autorisation. La sécurité logique repose en grande partie sur les dispositifs mis en place pour garantir la confidentialité dans la cryptographie, une gestion

efficace des mots de passe et des procédures d'authentification, complétées par des mesures antivirus, et de sauvegarde des informations sensibles.

La sécurité logique vient d'être évoquée principalement, sous l'angle de la confidentialité. Elle doit être complétée par celui lié au développement, au cycle de vie et à l'intégration des applications.

I.5.4. Sécurité applicative:

La sécurité applicative comprend un développement pertinent de la sécurité logicielle ainsi que son intégration et exécution harmonieuse dans des environnements opérationnels .

Elle repose essentiellement sur :

- ✓ Une méthodologie de développement en particulier le respect des normes;
- ✓ L'intégration de mécanismes de sécurité, d'outils d'administration et de contrôle de qualité dans les applications,
- ✓ La robustesse des applications,
- ✓ Un plan d'assurance sécurité.

I.5.5 Sécurité des télécommunications :

La sécurité des télécommunications consiste à offrir à l'utilisateur final, c'est-à-dire aux applications communicantes, une connectivité fiable et de qualité de bout en bout. Pour cela il faut mettre en œuvre un canal de communication sûr entre les correspondants, quels que soient le nombre et la nature des éléments intermédiaires (réseaux ou systèmes) nécessaires à l'acheminement des données. Ceci implique la réalisation d'une infrastructure réseau sécurisée au niveau des accès, des protocoles de communication, des systèmes d'exploitation et des équipements.

I.5.6. Services de sécurité:[6]

En transmission de données, le terme sécurité recouvre tout ce qui concerne la protection de l'information. L'ISO (international standard organization) s'attache donc

à prendre toutes les mesures nécessaires a la sécurité des données durant leur transmission. Ses travaux ont donné naissance à un standard international : ISO 7498-2 (OSI basic Référence Model – Part 2 : Security Architecture).

Trois concepts ont été définis:

- ✓ Les fonctions de sécurité, qui sont déterminées par les actions pouvant compromettre la sécurité d'un établissement.
- ✓ Les mécanismes de sécurité, qui définissent les algorithmes à mettre en œuvre.
- ✓ Les services de sécurité, qui représentent les logiciels et le matériel mettant en Œuvre des mécanismes dans le but de mettre à la disposition des utilisateurs les fonctions de sécurité dont ils ont besoin.

Le système d'information est généralement défini par l'ensemble des données et des ressources matérielles et logicielles de l'entreprise permettant de les stocker ou de les faire circuler. Le système d'information représente un patrimoine essentiel de l'entreprise, qu'il convient de protéger et d'assurer:

- **La confidentialité :**

La confidentialité consiste à rendre l'information inintelligible à d'autres personnes que les seuls acteurs de la transaction.

- **L'intégrité :**

Vérifier l'intégrité des données consiste à déterminer si les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).

- **La disponibilité :**

L'objectif de la disponibilité est de garantir l'accès à un service ou à des ressources.

- **La non-répudiation :**

La non répudiation de l'information est la garantie qu'aucun des correspondants ne pourra nier la transaction.

- **L'identification et l'authentification :**

L'authentification consiste à assurer l'identité d'un utilisateur, c'est-à-dire de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être. Un contrôle d'accès peut permettre (par exemple par le moyen d'un mot de passe qui devra être crypté) l'accès à des ressources uniquement aux personnes autorisées.

- **Contrôle d'accès :**

Mécanisme permettant de prévenir contre l'utilisation non appropriée ou non autorisée d'une ressource (service, système programme, données).

La détermination des niveaux d'accès aux ressources dépend de l'évaluation de leurs niveaux de sensibilité. Cela est basé le plus souvent, sur une classification des données, des personnes et des flux informationnels.

I.5.7 Politique de sécurité : [3]

I.5.7.1 Définition :

Une politique de sécurité exprime la volonté des responsables de protéger les valeurs informationnelles et les ressources technologiques de l'organisation. Elle spécifie les moyens (ressources, procédures, outils,…) qui répondent de façon complète et cohérente aux objectif stratégiques de sécurité.

I.5.7.2 En quoi consiste la politique de sécurité ?

La politique de sécurité fait le lien entre la stratégie de sécurité de l'entreprise et la réalisation opérationnelle de la sécurité.

Elle permet de transcrire le travail effectué pour comprendre les risques et leurs impacts, en des mesures concrètes de sécurité. Elle donne de la cohérence à la gestion et contribue à adopter vis-à-vis des risques une attitude préventive et proactive, pas seulement réactive, ainsi une meilleure efficacité des mesures.

Une bonne définition et une bonne réalisation d'une politique de sécurité autorisent une certaine maîtrise des risques, tout en réduisant leur probabilité d'apparition.

I.5.7.3 Qui doit appliquer et gérer ces politiques ?

La personne ou le groupe chargé de gérer et d'entretenir le réseau et sa sécurité doivent avoir accès à toutes ses zones.

La fonction de gestion des politiques de sécurité doit donc être confiée à des personnes particulièrement dignes de confiance et disposant des compétences techniques nécessaires.

Cette personne ou ce groupe ne doit donc pas constituer une menace potentielle. Une fois désignés, les gestionnaires du réseau bénéficient d'outils logiciels sophistiqués leur permettant de définir, de distribuer, de renforcer et d'évaluer les politiques de sécurité au moyen d'interfaces utilisant le modèle d'un navigateur Internet.

I.6 La cryptographie :

- **Définition :**

Issu du grec cryptos (cacher) et graphie (écriture), La cryptographie est la science qui utilise les mathématiques pour le cryptage et le décryptage de données.
Elle permet ainsi de stocker des informations confidentielles ou de les transmettre sur des réseaux non sécurisés (tels que l'Internet), afin qu'aucune personne autre que le destinataire ne puisse les lire.

Alors que la cryptographie consiste à sécuriser les données, la cryptanalyse est l'étude des informations cryptées, afin d'en découvrir le secret. La cryptanalyse classique implique une combinaison intéressante de raisonnement analytique, d'application d'outils mathématiques, de recherche de modèle, de patience, de détermination et de chance. Ces cryptanalyses sont également appelés des pirates.

.

I.6.1 la cryptographie classique : [9]

I.6.1.1 Principe :

Un système cryptographique permet de transformer un message intelligible ou message en clair M en un message chiffré incompréhensible ou cryptogramme $C = E_{KC}(M)$. Ce procédé est désigné sous le terme de chiffrement ou " cryptage " la procédure de chiffrement utilise un algorithme et une clé de chiffrement K_C.Le message chiffré peut être mis sur un support non sécurisé. Le récepteur procédera au déchiffrement du message afin de retrouver le message en clair en appliquant la transformation inverse avec une clé de déchiffrement $K_d : M = D_{Kd}[E_{KC}(M)]$. (Figure I.1).

Les opérations de chiffrement reposent en général sur l'utilisation :

-D'un algorithme qui représente le modèle mathématique choisi ;

-D'une clé représentant le paramètre de mise en œuvre.

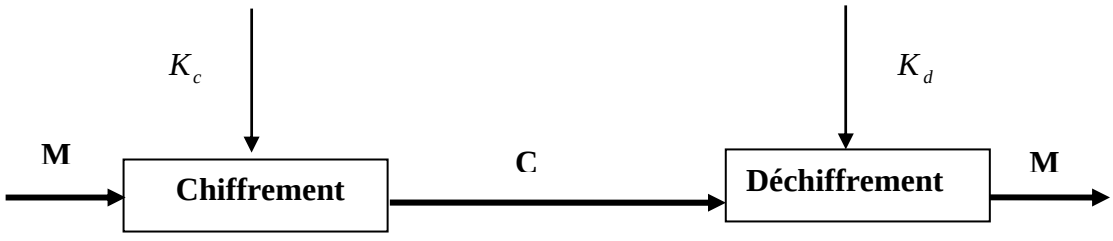


Figure. I.1. Système de cryptographie.

I.6.1.2 Systèmes de chiffrement :

On peut chiffrer un message selon deux procèdes :

-Chiffrement par bloc,

-Chiffrement par flux.

I.6.1.2.1 Chiffrement par bloc :

I.6.1.2.1.a Les modes de chiffrement : [17]

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) :**

C'est le mode opératoire le plus simple où chaque bloc est crypté séparément. L'inconvénient de ce mode est que deux blocs identiques engendrent deux blocs chiffrés identiques, ce qui le rend vulnérable. (Figure I.2)

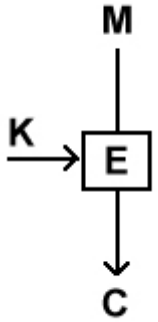


Figure. I.2. Mode ECB.

- **Mode CBC(cipher bloc chaining) :**

Méthode de chiffrement qui consiste à chiffrer un message **M** par bloc et dont le chiffrement dépend non seulement de la clé et du texte en clair mais aussi du bloc précèdent ou, lorsqu'il s'agit du premier bloc, d'un vecteur d'initialisation. (Figure I.3).

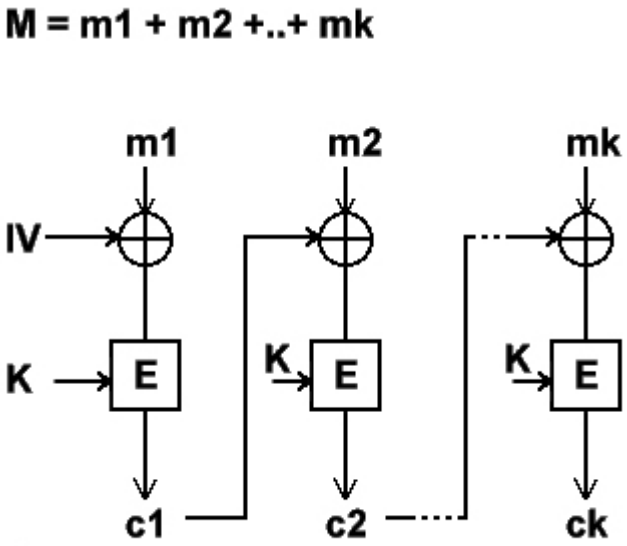


Figure I.3. Mode CBC.

- **Mode CFB (cipher feed back):**

Le mode CFB est basé sur un rebouclage de N bits de cryptogramme, initialisé en entrée avec un vecteur d'initialisation et combiné en sortie a l'aide d'un ou- exclusif avec le message clair. (Figure I.4).

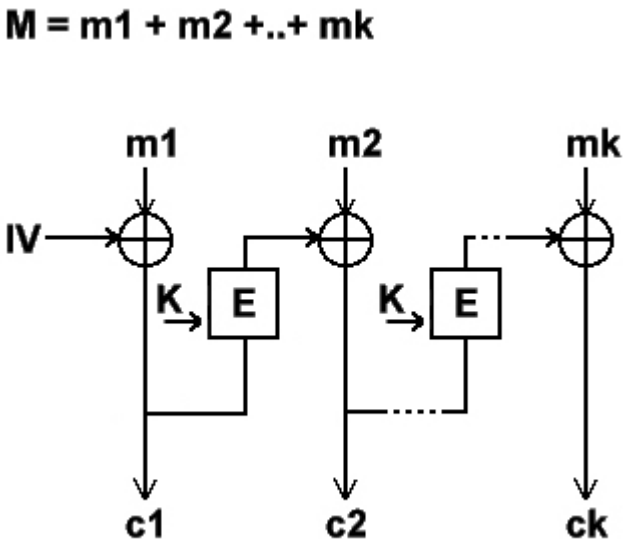


Figure. I.4.Mode CFB.

- **Mode OFB (output feed back):**

Ce mode est basé sur un rebouclage de N bits les plus significatifs du bloc de sortie, et combiné avec le message clair à l'aide d' un ou- exclusif. (Figure I.5).

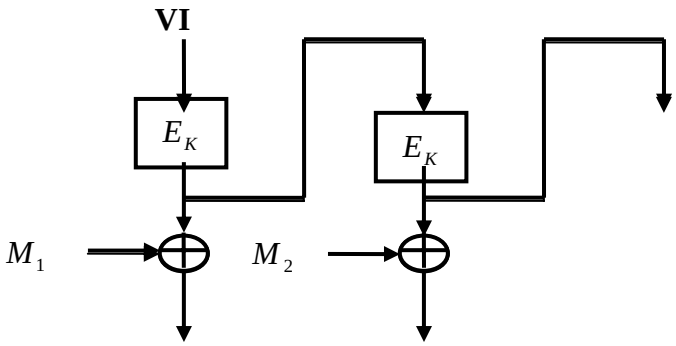


Figure 1.5. Mode OFB

I.6.1.2.1.b Chiffrement par substitution :

Chaque caractère du message en clair et provenant d'un alphabet noté A, est remplacé par le caractère correspondant appartenant à un ou plusieurs alphabets de

substitution. L'alphabet de substitution est un réarrangement de l'ordre lexicographique des caractères de A. Il existe quatre types de substitution : simple, homophonique, polyalphabétique et polygrammique.

a. Substitution simple :

On parle de substitution simple ou de chiffrement monoalphabétique lorsque chaque lettre du message clair est toujours remplacée par le même symbole. On obtient ainsi une bijection entre les lettres claires et les symboles de l'alphabet de chiffrement.

Soit A un alphabet à n caractères :$\{ a_O, a_1, ..., a_{N-1} \}$, et soit S l'alphabet de substitution à n caractères $\{ s_O, s_1, ..., s_{N-1} \}$ où à chaque élément de A on associe un seul et unique élément de S.

Exemple : En prenant les alphabets A et S suivants :

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

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

Le chiffrement du message M nous donne:

M: C O N F I D E N T I E L

C: S M L B G A R L T G R J

b. Substitution homophonique :

C'est le même principe que la substitution simple si ce n'est que la transformation n'est plus bijective. A une lettre il peut correspondre plusieurs autres caractères (homophones) choisis aléatoirement. Ainsi le décryptage se fait en suivant le bon sens de la phrase ou du mot.

Exemple :

Supposons que les lettres de l'alphabet sont chiffrées par des entiers compris entre 0 et 99, et que le nombre d'homophones affectés à une lettre est proportionnel à sa fréquence relative. Soit la table d'homophones suivants :

Lettres	homophones
E	17 19 84 41 56 60 67 83
I	08 22 53 65 88 90
C	03 44 76
N	02 09 15 27 32 40 59
O	01 11 23 28 42 54 70 80
D	33 91 45 58 64 78
F	05 10 20 29

Soit le message M : C O N F I D E N C E

Le message chiffré C est: 76 11 59 05 65 91 17 27 03 41

c. Substitution polyalphabétique :

Pour ce faire, on utilise plusieurs tables de substitution simple, et la table utilisée dépend de la position de la lettre dans le texte. Par exemple, on prend les 3 tables de correspondance suivantes, C étant l'alphabet clair ; C1, C2 et C3 les alphabets cryptés :

C	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
C1	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
C2	a	z	e	r	t	y	u	i	o	p	q	s	d	f	g	h	j	k	l	m	w	x	c	v	b	N
C3	m	l	k	j	h	g	f	d	s	q	y	t	r	e	z	a	u	i	o	p	n	b	v	c	x	W

Maintenant, imaginons que l'on veut crypter le mot CONFIDENTIEL on change de table à toutes les lettres et l'on boucle tous les 3 caractères (car nous n'avons ici que 3 tables), ce qui nous donne :

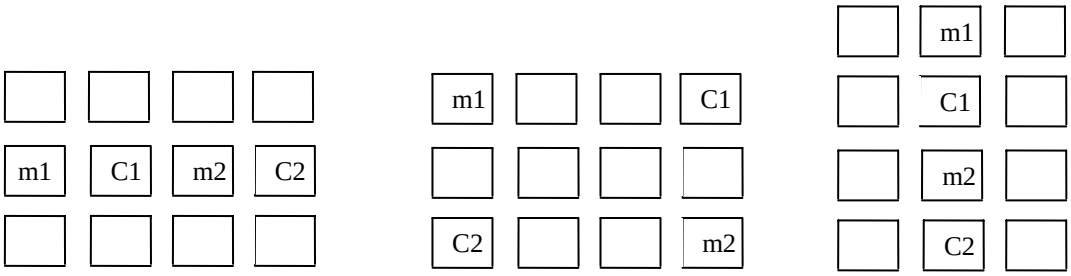
C O N F I D E N T I E L

P G E S O J R F P V T T

d. Substitution polygrammique :

Dans un chiffrement polygrammique, un groupe de m lettres est chiffré par un groupe de m symboles. Parmi les nombreux exemples de tels chiffrements, on citera le chiffrement de Playfair.

Dans le chiffrement de Playfair, la clé est constituée d'une matrice 5×5.chaque paire de caractères $m_1 m_2$ du message est chiffré selon les 3 règles suivantes, basées sur la position de $m_1 m_2$ dans la matrice.



Règle 1 **Règle 2** **Règle 3**

Exemple :

Soit la matrice suivante construite à l'aide d'un mot-clé « CESAR »:

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

Soit le message **M :** CO NF ID EN TI EL

Le message chiffré **C** est **:** BV LH KB RK OM SK

I.6.1.2.1.c Chiffrement par permutation :

Le chiffrement par permutation consiste à changer l'ordre des lettres du message selon un procédé fixé à l'avance.

Choisissons une clé, par exemple : BONJOUR A TOUS. On réécrit les lettres de cette clé dans l'ordre alphabétique de façon à les associer à des nombres.

A -B -J- N- O - O - O - R- S - T- U - U
1- 2- 3- 4 - 5 - 6 - 7 - 8 - 9 -10 -11-12

La clé alphabétique BONJOUR A TOUS nous donne ainsi la clé numérique suivante :

2 -5 - 4 -3 - 6 -11 -8 - 1 -10 - 7 – 12 – 9

Ecrivons un message M dans un tableau en utilisant cette clé. Si Le message en clair est M=UNIVERSITE MOULOUD MAMMERI. On écrit:

2 - 5 - 4 -3 -6 -11 - 8 - 1 -10 - 7 -12 - 9
U- N- I- V- E- R- S- I- T- E- M- O
U - L- O-U- D- M- A- M- M- E- R- I

Le texte est en suite envoyé colonne par colonne en suivant l'ordre de la clé. On envoie la première colonne UU puis la deuxième NL et ainsi de suite. Le message chiffré sera :

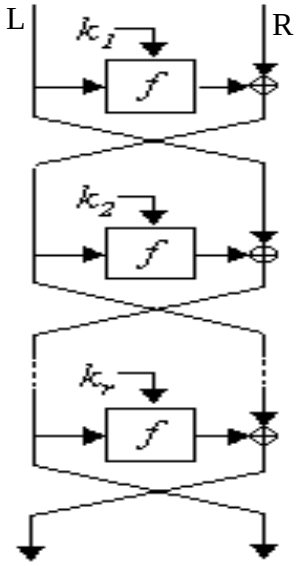
UU NL IO VU ED RM SA IM TM EE MR OI.

I.6.1.2.1.d Chiffrement par blocs avec itération:

Un algorithme de chiffrement par blocs avec itération est un algorithme qui chiffre les blocs par un processus comportant plusieurs rondes. Dans chaque ronde, la même transformation est appliquée au bloc, en utilisant une sous-clé dérivée de la clé de

chiffrement. En général, un nombre de rondes plus élevé garantit une meilleure sécurité, au détriment des performances.

Un cas particulier d'algorithmes de chiffrement par blocs avec itération est la famille des chiffres de Feistel (figure 6). Dans un chiffre de Feistel, un bloc de texte en clair est découpé en deux ; la transformation de ronde est appliquée à une des deux moitiés, et le résultat est combiné avec l'autre moitié par un ou exclusif. Les deux moitiés sont alors inversées pour l'application de la ronde suivante. Un avantage de ce type d'algorithmes est que le chiffrement et le déchiffrement sont structurellement identiques**.**



$$L_I = R_{I-1}$$

$$R_I = L_{I-1} \oplus F(R_{I-1}, K_I)$$

- Figure. I.6. Principe du chiffre de Feistel.

I.6.1.2.2 Chiffrement par flux : [8]

Les algorithmes de chiffrement par flux (Stream chiper) peuvent être définis comme étant des algorithmes de chiffrement par blocs, où le bloc a une dimension unitaire (1 bit, 1 octet, etc.) ou relativement petite. Leurs avantages principaux viennent du fait que la transformation (méthode de chiffrement) peut être changée à chaque symbole du texte

clair et du fait qu'ils soient extrêmement rapides. De plus, ils sont utiles dans un environnement où les erreurs sont fréquentes car ils ont l'avantage de ne pas propager les erreurs (diffusion, voir annexe). Ils sont aussi utilisés lorsque l'information ne peut être traitée qu'avec de petites quantités de symboles à la fois, comme par exemple dans le cas où l'équipement n'a pas de mémoire physique ou une mémoire tampon très limitée.

Il existe une différence entre le chiffrement par blocs et le chiffrement par flux dans la mesure où les chiffrements par bloc agissent sur des données avec une transformation fixe des grands blocs de données en clair ; les chiffrements par flux agissent avec une transformation variant avec le temps des chiffres en clairs individuels.

I.6.1.3 Cryptage symétrique : [3]

Le cryptage à clé privée ou symétrique est basé sur une clé (ou algorithme) partagée entre les deux parties communicantes. Cette même clé sert à crypter et décrypter les messages. Les algorithmes de chiffrement les plus connus sont : Kerberos, DES (Data Encryption Standard).

Le principal problème est le partage de la clé : Comment une clé utilisée pour sécuriser peut être transmise sur un réseau insécurisé ? La difficulté engendrée par la génération, le stockage et la transmission des clés (on appelle l'ensemble de ces trois processus le management des clés : Key management) limite le système des clés privées surtout sur Internet. (Figure I.7).

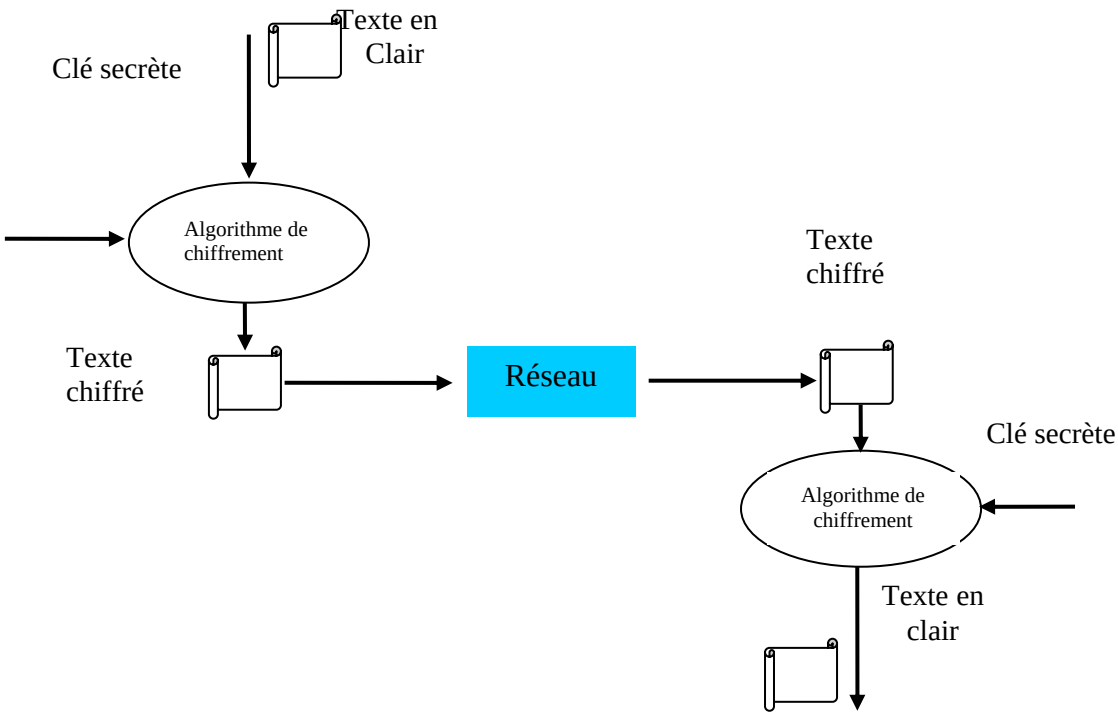


Figure I.7.Algorithme de chiffrement symétrique.

I.6.1.4 Cryptage asymétrique : [3]

Ce système de cryptage utilise deux clés différentes pour chaque utilisateur : L'une est privée et n'est connue que de l'utilisateur ; l'autre est publique et donc accessible à tout le monde.

Les clés, publique et privée, sont mathématiquement liées par l'algorithme de décryptage de telle manière qu'un message crypté avec une clé publique ne puisse être décrypté qu'avec la clé privée correspondante. Une clé est donc utilisée pour le cryptage et l'autre pour le décryptage. (Figure. I.8).

Ce cryptage présente l'avantage de permettre le placement des signatures numériques dans le message et ainsi permettre l'authentification de l'émetteur. Le principal avantage du cryptage à clé publique est de résoudre le problème de l'envoi de clé privée sur un réseau non sécurisé. Bien que plus lent que la plupart des cryptages à clé privée il reste préférable pour 3 raisons :

- Plus évolutif pour les systèmes possédant des millions d'utilisateurs.
- Authentification plus flexible.

- Supporte les signatures numériques.

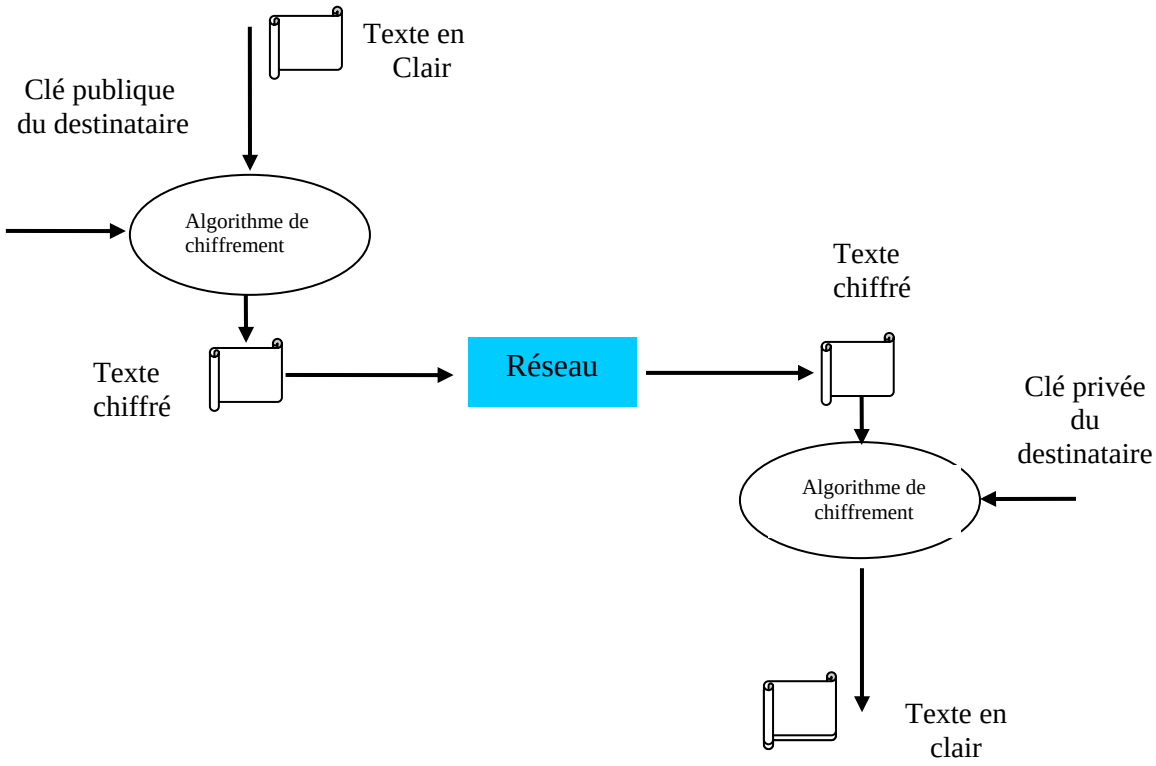


Figure. I.8.algorithme de chiffrement asymétrique.

I.6.1.5 La clé :

On appelle clé une valeur utilisée dans un algorithme de cryptographie, afin de chiffrer un texte. Il s´agit en fait d´un nombre complexe dont la taille se mesure en bits. On peut imaginer que la valeur numérique correspondant à 1024 bits est absolument gigantesque. Plus la clé est grande, plus elle contribue à élever la sécurité à la solution.Toutefois, c´est la combinaison d´algorithmes complexes et de clés importantes qui seront la garantie d´une solution bien sécurisée. Les clés doivent être stockées de manière sécurisée et de manière à ce que seul leur propriétaire soit en mesure de les atteindre et de les utiliser.

I.6.1.6 Gestion des clés :

La mise en œuvre d'un système de cryptographie doit passer par la diffusion des clés aux correspondants concernés. Dans le cas de la cryptographie à clé secrète, ceci se

traduit par le choix d'une clé secrète commune et son envoi aux deux correspondants doit se faire en utilisant un canal sûr (envoi postal, téléphonie, porteur). Dans le cas de la cryptographie à clé publique, les correspondants doivent s'échanger leurs clés publiques, l'entité qui s'occupe de la conservation et de la divulgation des clés publiques est appelée gestionnaire des clés. Le gestionnaire peut être, par exemple, un serveur ou les clés publiques peuvent être consultées.

La distribution des clés constitue ainsi un problème majeur en matière de cryptographie.

I.6.1.7 Signature numérique :

L'un des principaux avantages de la cryptographie de clé publique est qu'elle offre une méthode d'utilisation des signatures numériques. Celles-ci permettent au destinataire de vérifier leur authenticité, leur origine, mais également de s'assurer qu'elles sont intactes. Ainsi, les signatures numériques de clé publique garantissent l'authentification et l'intégrité des données. Elles fournissent également une fonctionnalité de non répudiation, afin d'éviter que l'expéditeur ne prétende qu'il n'a pas envoyé les informations.

Ces fonctions jouent un rôle tout aussi important pour la cryptographie que pour la confidentialité.

La signature numérique d'un document se fait en deux temps. Tout d'abord, il est appliqué une fonction mathématique dite ‘’de hachage’’, qui va à partir d'un texte générer une suite, de nombres binaires, de taille fixe et ce quelque soit la taille du texte de départ. Cela constitue l'empreinte du texte. Si un caractère du texte change (‘’a’’ devient ‘’A’’) l'empreinte ne sera pas la même. Ensuite l'empreinte est chiffrée avec la clé privée de l'utilisateur, puis attachée au texte original et enfin le message est envoyé.

I.6.1.8 La fonction de hachage : [12]

Une fonction de hachage est aussi appelée fonction de hachage à sens unique ou "one-way hash function" en anglais. Ce type de fonction est très utilisé en cryptographie, principalement dans le but de réduire la taille des données à traiter par l'algorithme de chiffrement. En effet, la caractéristique principale d'une fonction de hachage est de

produire un haché des données, c'est-à-dire un condensé de ces données. Ce condensé est de taille fixe, dont la valeur diffère suivant la fonction utilisée.

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.

I.6.1.9 Certification des clés :

Afin d’éviter toute tentative de déguisement, il faut mettre en place un dispositif de certification permettant de vérifier que la clé publique est bien associée au détenteur légitime et d’authentifier le gestionnaire de clés. Un certificat est un document émis par une institution reconnue et attestant de la propriété d’une clé publique par un correspondant. L’infrastructure qui assure la gestion et la distribution des clés la plus connue est **PKI** (Public Key Infrastructure). (figure.I.9).

Les certificats doivent être infalsifiables, pouvoir être obtenus en toute sûreté et crées de telle façon que personne d’autre que leur destinataire légitime ne puisse les utiliser. Un

certificat comprend en général les éléments suivants : la clé publique, le nom du propriétaire, la date d'expiration de la clé, le nom du responsable du certificat et le numéro de série de celle-ci.

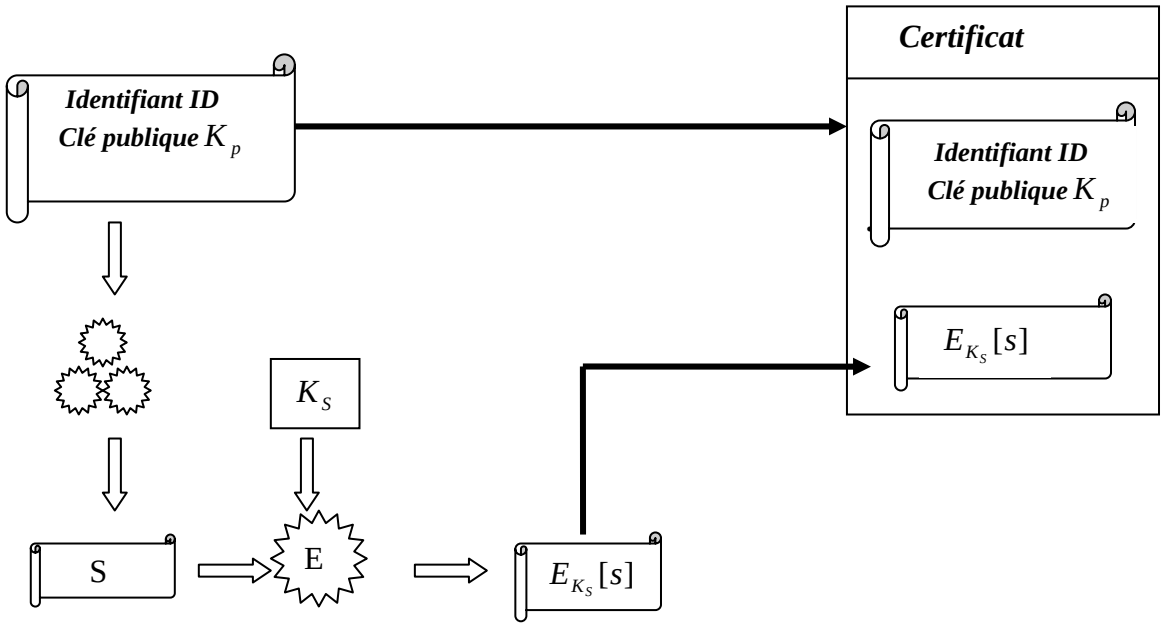


Figure. I.9. Procédé de certification de clé.

I.6.2 Cryptographie quantique : [7]

La cryptographie quantique, se base sur la physique notamment sur le concept d'incertitude telle que révélé par Werner Heisenberg.

I.6.2.a 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 bits 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.

I.6.2.b 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.

I.6.2.c 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.

I.6.2.d 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.

I.7 Conclusion :

Dans un environnement informatique et de télécommunication, la mise en œuvre d'un système de sécurité et des techniques de chiffrement permet d'assurer la confidentialité des données et de vérifier l'intégrité et l'authentification des entités. Pour cela les systèmes font appelle aux différents algorithmes de chiffrement existant qui seront l'objet d'étude du chapitre suivant.

II. Les Algorithmes de cryptage :

Les systèmes de chiffrement font appel à des algorithmes de chiffrement souvent complexes qui modifient, à l'aide d'une clé de chiffrement plus ou moins longue, les caractères à protéger pour générer des données aléatoires. Ils se composent de deux principales classes: symétrique et asymétrique.

II.1 Algorithmes symétriques :

Ce sont les algorithmes basés sur le chiffrement et déchiffrement à clé secrète.

II.1.1 Algorithme DES (Data encryption standard) : [11]

Adopté en 1977 par NIST, le DES est un système de chiffrement par bloc de 64 bits selon les Modes (ECB, CBC, CBF, OFB) et une clé de 64 bits. Le DES peut être mis en œuvre par logiciel ou par microcircuits et peut être utilisé pour la protection des mots de passe, le chiffrement de bout en bout et le chiffrement des fichiers stockés sur des supports amovibles.

a. Fonctionnement:

Le DES opère sur des blocs de 64 bits avec des clés de 64 bits dont 8 bits de parité. Il est basé sur un ensemble de transformations composées de substitution, de transposition et d'opérations non linéaires (figure II.1).

Après une permutation initiale IP, le bloc est découpé en deux parties gauche et droite de 32 bits. Il y aura 16 rondes où les données sont combinées avec la clé, après le $16^{éme}$ ronde, les parties gauches et droites sont transposées et rassemblées puis une permutation finale IP^{-1}.

Au cours de chaque ronde, la partie droite est soumise à une fonction F composée de quatre opérations :

1. Extension à 48 bits en utilisant des tables E ;
2. Combinaison avec la clé secondaire K_1 à l'aide d'un « ou exclusif » ;

3. Décomposition en 8 mots de 6 bits associés à 8 tables de substitution S1 à S8, chaque mot servant à fournir les numéros de ligne et de colonne (figure II.2) ;
4. Extraction de chaque table de substitution S1 à S8 d'un mot sur 4 bits.

En termine par une permutation P du 32 bits obtenus.

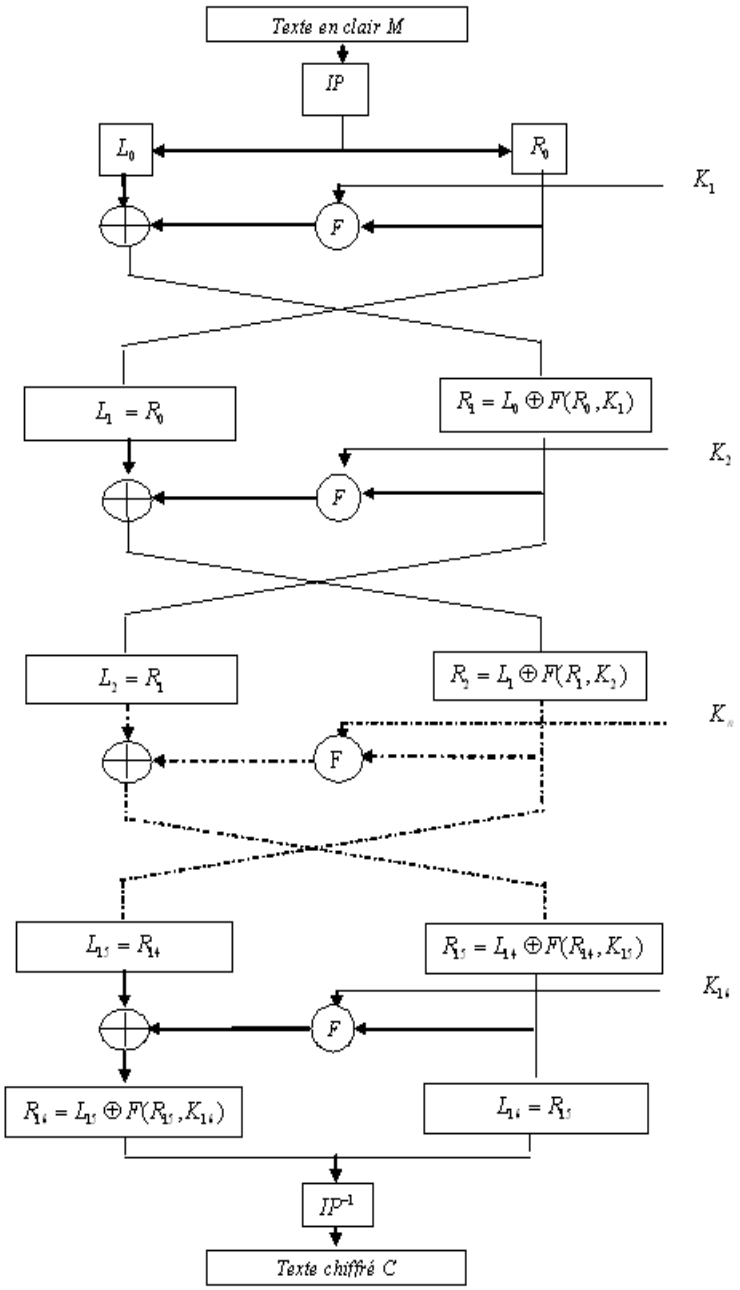


Figure II.1 fonctionnement du DES.

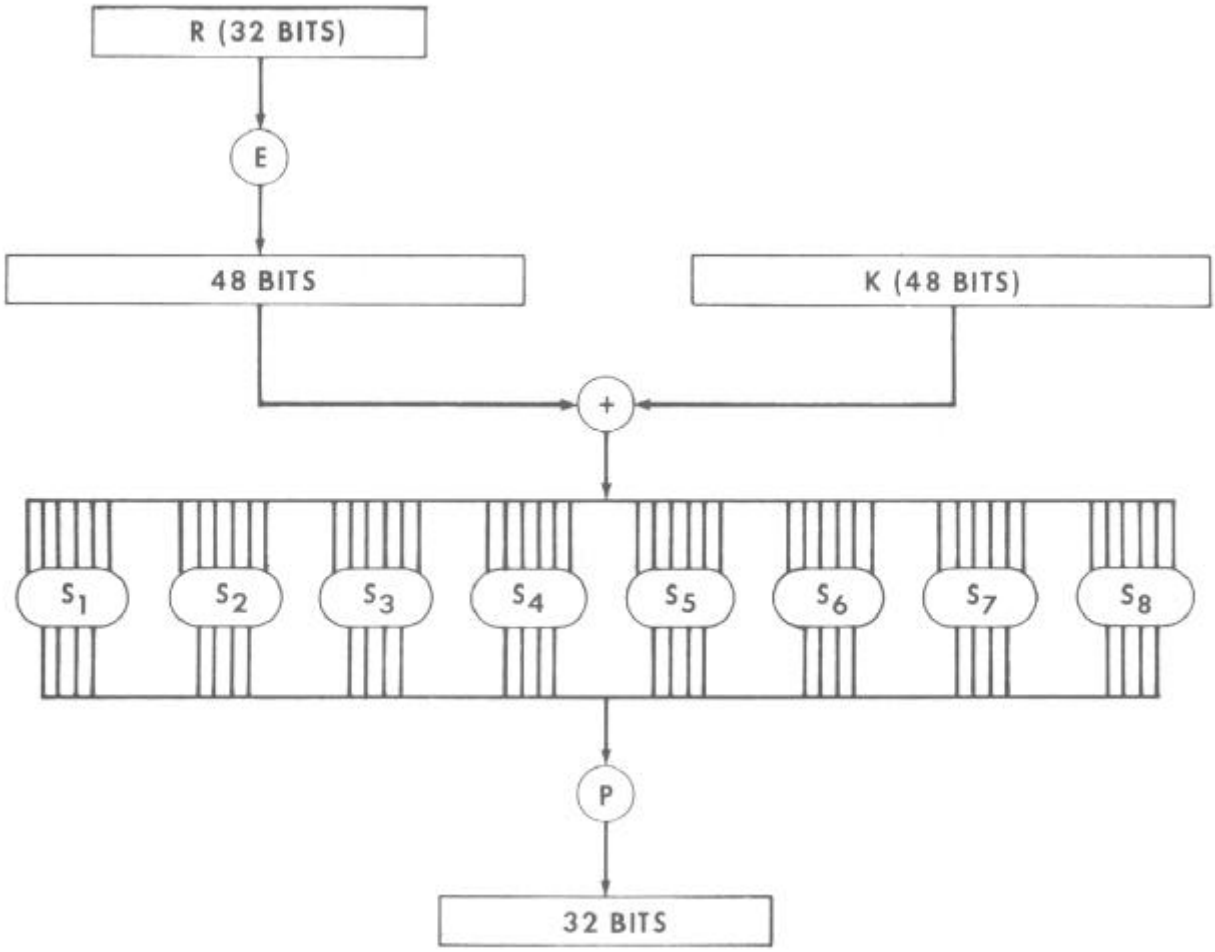


Figure II.2 Table de substitution S.

a.1 1*ére* Étape Permutation initiale :

Les 64 bits du bloc d'entrée subissent la permutation initiale présentée par la matrice suivante :

IP

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

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

a.2 De la $2^{éme}$ à la $17^{éme}$ étape:

Les 64 bits initiaux de données sont divisés en 2 blocs (L et R).

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

Les blocs **L** et **R** sont soumis à un ensemble de transformations itératives appelées rondes à l'aide des sous clés générées à partir de la clé principale. L'ensemble des transformations se traduit par les équations suivantes :

- $L_I = R_{I-1}$
- $R_I = L_{I-1} \oplus F(R_{I-1}, K_I)$.

Où :

i =1……..16.

F : fonction mathématique qui représente l'élément important sur lequel repose la sécurité du DES composée de :

1. Extension de R_{i-1} de 32 bits à 48 bits en utilisant la table E ;

E

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

2. La matrice résultante de 48 bits est appelée **E** [R_{i-1}]. L'algorithme DES procède ensuite à un OU exclusif entre la clé $\mathbf{K}_i$ et **E** [R_{i-1}], le résultat est une matrice de 48 bits ;
3. **E** [R_{i-1}] est ensuite partagé en 8 blocs de 6 bits, noté Bj (j=1……8) qui sera l'entrée d'une table de substitution S ;
4. chaque table reçoit une information de 6 bits et envoie en sortie une information de 4 bits.

Le bloc en entrée de 6 bits est traité comme suit : le premier et le dernier bit du bloc unis représentent un nombre compris entre 0 et 3 qui désignent le numéro de la ligne de la table Si, les 4 bits restants représentent un nombre compris entre 0 et 15 qui donne le numéro de la colonne.

On regroupe les mots de 4 bits de chaque table pour former un mot de 32 bit qui subira une permutation selon la table P.

P

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

a.3 Permutation finale :

Le contenu du bloc de pré-sortie, est permuté une dernière fois. Cette permutation correspond à l'inverse de la permutation initiale.

IP^{-1}

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

b. Génération des clés :

La clé initiale est de 64 bits, à partir de laquelle en génère 16 sous clés K_i chacune sur 48 bits, en suivant les étapes ci-dessous :

1. Enlever les bits de parités afin d'obtenir une clé de longueur de 56 bits ;
2. Application d'une première permutation notée CP-1 dont la matrice est présentée ci-dessous ;

CP-1

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

3. On divise la matrice CP-1 en deux matrices G et D de 28 bits chacune ;

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

4. Les blocs subissent un décalage à gauche, puis regrouper pour former un bloc de 56 bits. Ce dernier subira une permutation CP-2 fournissant en sortie un bloc de 48 bits, représentant la clé Ki.

CP-2

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

Des itérations de l'algorithme permettent de donner les 16 sous clés $\mathbf{K}_1$ à $\mathbf{K}_{16}$ utilisées dans l'algorithme DES.

c. Performance de DES :

Pour casser le DES il faut $2^{56} = 7{,}2.10^{16}$ clés possibles pour trouver la clé de cryptage. Mais une longueur de 128 bits augmenterait considérablement la résistance du DES. La sécurité du DES dépend de la sécurité des clés utilisées.

II.1.2 Le triple DES : [15]

Le triple DES est venu pour améliorer les performances du DES parce que la faiblesse du DES est la faible longueur de sa clé. Le 3DES combine plusieurs chiffrement DES pour obtenir un système de chiffrement global avec une longue clé de 112 et 168 bits utilisant 2 ou 3 clé différentes. Le Triple DES est généralement utilisé avec seulement deux clés différentes. Le mode d'usage standard est de l'utiliser en mode EDE (Encryption, Decryption, Encryption, c'est-à-dire Chiffrement, Déchiffrement, Chiffrement) ce qui le rend compatible avec DES quand on utilise trois fois la même clé.

Une clé triple DES est donc composée de deux DES et fait 112 bits ce qui rend le 3DES hors porté d'une attaque exhaustive. On peut concevoir une variante à 3 clés différentes mais reste vulnérable à une attaque s'appuyant sur l'un des messages intermédiaires.

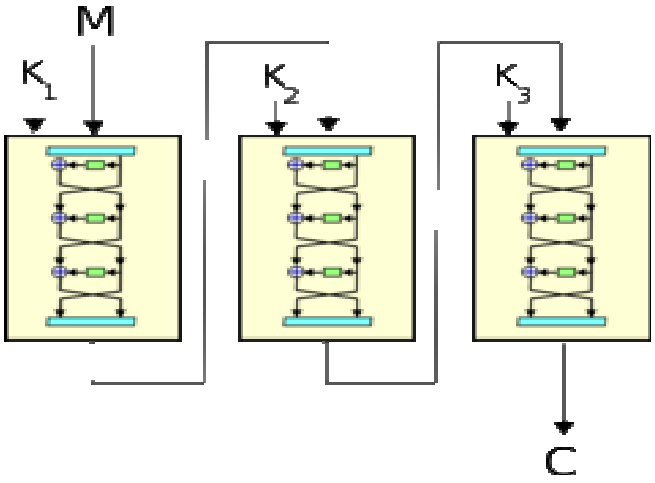


Figure II.3 Triple DES.

II.1.3 IDEA : [4]

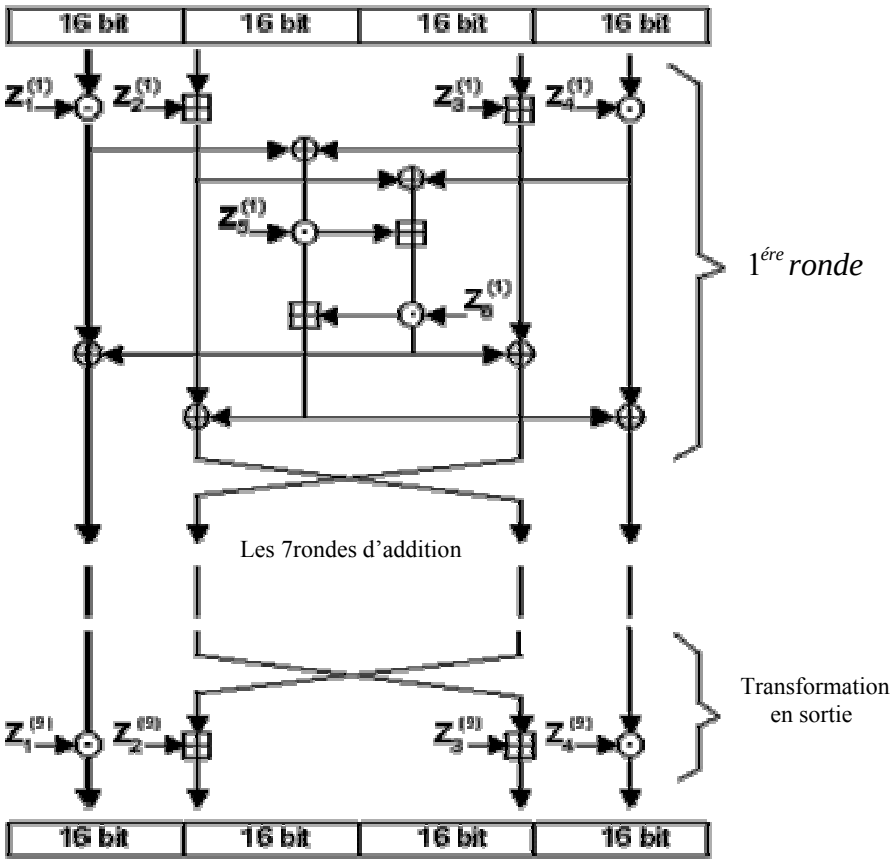
IDEA (International Data Encryption Algorithm) est un algorithme de chiffrement symétrique. Il manipule des blocs de texte en clair de 64 bits et utilise une clé de longueur de 128 bits (qui doit être choisie aléatoirement) pour le chiffrement des données, et on a besoin de la même clé secrète pour les déchiffrer. Le processus se résume à huit étapes de chiffrement identiques (les rondes), suivies d'une transformation au bloc de sortie.

IDEA utilise à la fois la confusion et la diffusion (voir annexe). L'algorithme est basé sur le mélange d'opérations de différents groupes algébriques, et toutes ces opérations sont facilement réalisables à la fois en logiciel et en matériel.

Le bloc de 64 bits en entrée est tout d'abord divisé en quatre blocs de 16 bits chacun. À noter que toutes les opérations algébriques utilisées dans le chiffrement fonctionnent avec des blocs de 16 bits (Figure II.4).

Un processus produit, pour chacun des 8 rondes, 6 sous-clés de 16 bits chacune selon la clé de 128 bits. Avec quatre autres clés générées pour la transformation finale à la suite des huit rondes, un total de 52 sous-clés doit être généré.

Texte clair (64 bis)



Texte chiffré (64 bits)

Figure. II.4 Fonctionnement d'IDEA.

- ⊕ OU exclusif
- ⊞ addition modulo 2^{16} ;
- ⊙ multiplication modulo $2^{16}+1$.

Dans le premier ronde, les quatre premières sous-clés sont combinées avec tout d'abord deux blocs de 16 bits du texte clair selon l'addition modulo 2^{16}, et avec deux autres blocs de texte clair en utilisant la multiplication modulo $2^{16} + 1$. Les résultats sont ensuite traités plus loin, où deux autres sous-clés sont employées et l'opération du troisième groupe algébrique, le OU exclusif bit à bit, est utilisée. À la fin du premier ronde de chiffrement, quatre valeurs de 16 bits sont produites et elles sont utilisées comme valeurs d'entrées au deuxième ronde de chiffrement, dans un ordre interchangé.

Le même processus s'effectue pour la suite des rondes, les quatre derniers blocs de 16 bits sont combinés avec les quatre dernières sous clés. Le résultat final forme quatre blocs chiffrés de 16 bits, donc 64 bits de texte chiffré.

Le déchiffrement s'effectue essentiellement de la même manière que le chiffrement. La seule différence est que les 52 sous-clés sont générées de façon inverse au chiffrement. Aussi les blocs de texte chiffré doivent être traités dans l'ordre inverse du chiffrement pour inverser parfaitement le processus de chiffrement**.**

II.1.4 RIJNDAEL : [4]

Rijndael fut conçu par Joan Daemen et Vincent Rijmenle dans le but de devenir un candidat à AES (Advanced Encryption Standard) et adopté en 2000 après un appel d'offre par NIST (National Institute of Standards and Technology) pour élaborer un algorithme de chiffrement plus performant que le DES.

Le chiffrement a une longueur de bloc variable, une longueur de clé variable et un nombre de rondes variables. Par contre, Rijndael version "AES" est restreint à des longueurs de clé de 128, 192 et 256 bits avec une longueur de bloc fixée à 128 bits.

Comme la plupart des chiffrements par blocs modernes, le chiffrement s'effectue en deux parties : une procédure d'expansion de la clé et la fonction principale de chiffrement.

La fonction de chiffrement se divise en trois : une transformation initiale avec la clé (l'étape "Add Round Key", bloc XOR clé), une série de rondes puis une transformation finale.

Dans un ronde, quatre transformations sont appliquées au bloc à chiffrer :

1. SubBytes : il s'agit d'une substitution non-linéaire pendant laquelle chaque octet est remplacé par un autre octet choisi dans une table particulière (une Boite-S) ;
2. ShiftRows est une étape de transposition où chaque élément de la matrice est décalé cycliquement à gauche d'un certain nombre de colonnes ;
3. MixColumns effectue un produit matriciel en opérant sur chaque colonne (vu alors comme un vecteur) de la matrice ;
4. AddRoundKey qui combine par addition chaque octet avec l'octet correspondant dans une clé de ronde obtenue par diversification de la clé de chiffrement.

Rijndael a de bonnes performances aussi bien en matériel qu'en logiciel sous plusieurs environnements.

Le temps de calcul de diversification de la clé en sous-clés est excellent. Ceci permet d'obtenir de bonnes performances pour le changement de contexte de sécurité.
Rijndael a une grande flexibilité. Il a été conçu pour être flexible en termes de taille des clés et du bloc de chiffrement. L'algorithme peut être modifié en utilisant différents nombres de tours.

II.1.5 BLOWFISH : [4]

Blowfish est un algorithme de chiffrement par blocs. Il utilise une taille de bloc de 64 bits et la clé de longueur variable peut aller de 32 à 448 bits. Elle est basée sur l'idée qu'une bonne sécurité contre les attaques de cryptanalyse peut être obtenue en utilisant de très grandes clés pseudo-aléatoires.

Blowfish présente une bonne rapidité d'exécution excepté lors d'un changement de clé, il est environ 5 fois plus rapide que Triple DES et deux fois plus rapide que l'IDEA. Il demeure encore solide du point de vue cryptographique avec relativement peu d'attaques efficaces sur les versions avec moins de tours. La version complète avec 16 tours est à ce jour entièrement fiable et la recherche exhaustive reste le seul moyen pour l'attaquer.

Il y a deux parties dans l'algorithme : une première partie qui manipule l'expansion de la clé et une deuxième partie qui manipule le chiffrement des données.
Le principe de l'algorithme est basé sur la génération de deux ensembles de clés, il utilise un tableau P de 18 entrées et 4 S-Boxes de 256 éléments chacune. Les S-Boxes acceptent un mot de 8 bits en entrée et produisent une sortie de 32 bits. Une entrée du tableau P est utilisée à chaque tour. Au tour final la moitie du bloc des données subit un XOR avec un des deux éléments restant de P.
La préparation de la structure à partir de la clé commence par l'initialisation du tableau P et des S-Boxes.

On opère ensuite un XOR entre la clé secrète et les entrées du tableau P, un bloc de 64 bits tous à zéro est ensuite chiffré. Le résultat remplace le premier et le deuxième élément du P, on continue ainsi jusque au remplacement de tous les éléments du tableau P et des S-Boxes.

Mettre en application l'algorithme de Blowfish semble une option pratique pour le chiffrement de données étant donné qu'il est destiné à être rapide, compact, simple et relativement sûr.

II.1.6 E0: [15]

E0 est l'algorithme de chiffrement à flot utilisé pour protéger la confidentialité des communications dans le protocole de transmission sans fil de courte portée Bluetooth. Le Bluetooth fonctionne suivant le principe d'un protocole de type maître-esclave.

Il génère une suite pseudo aléatoire avec laquelle on effectue un XOR avec les données qui sont transmises sous forme de trames initialisées au moyen d'une clé secrète qui reste fixe tout au long de la session. La clé peut avoir une taille variable mais sa longueur est généralement 128 bits.

À chaque itération, E0 génère un bit grâce à quatre registres à décalage de longueurs différentes (25, 31, 33, 39 bits) et deux états internes de 2 bits chacun. À chaque coup d'horloge, les registres sont décalés et les deux états sont mis à jour en utilisant l'état courant, l'état précédent et les valeurs présentes dans les registres à décalage. Quatre bits sont extraits des quatre registres à décalage et sont additionnés. L'algorithme effectue ensuite un XOR entre cette somme et la valeur du registre de 2 bits, le premier bit ainsi obtenu est la sortie pour le chiffrement. E0 se divise en trois parties :

- Préparation de la clé (payload Key generator) ;
- Génération du flux (keystream generator) ;
- Chiffrement.

La préparation de l'état initial utilise la même structure que la génération du flux de bits aléatoires. On est donc en présence de deux E0 couplés. Un état initial de 132 bits est produit par le premier stage à partir de quatre entrées (clé de 128 bits, adresse Bluetooth sur 48 bits et compteur du maître de 26 bits). Le résultat passe ensuite dans une opération polynomiale et on obtient une clé que l'on transmet à l'étape suivante, celui qui va générer le flux utilisé pour le chiffrement. Ces bits sont introduits dans les registres à décalage de la deuxième étape. On produit ensuite 200 bits pseudo-aléatoires grâce à 200 coups d'horloge du générateur, les derniers 128 bits sont insérés dans les registres à décalage, c'est l'état initial du générateur par flot.

Sa faiblesse réside dans le fait que le générateur par combinaison de registres utilisé est vulnérable à plusieurs attaques. On peut mentionner par exemple une attaque linéaire, des attaques algébriques et algébriques rapides, et des attaques par corrélation rapides.

Mais la plupart de ces attaques nécessitent la connaissance d'un grand nombre de bits successifs de suite chiffrante, ce qui n'est pas possible dans le contexte pratique du protocole Bluetooth puisque l'état initial du générateur est modifié tous les 2745 bits.

II.1.7 KASUMI : [2]

KASUMI (aussi connu sous le nom de A5/3) est un algorithme de chiffrement par bloc utilisé dans le cadre de la téléphonie mobile avec la norme 3GPP. Il est employé pour la confidentialité et l'intégrité de 3GPP.

L'algorithme à été conçu par le Security Algorithm Group of Experts (SAGE), qui fait partie du groupe de standards européens ETSI. Le SAGE a optimisé le MISTY1pour une implémentation matérielle. De ce fait KASUMI et MISTY1 sont très similaires. KASUMI travail sur des blocs de 64 bits et une clé de 128 bits. Cet algorithme put être divisé en deux parties :

- La gestion de la clé,
- Le chiffrement et le déchiffrement.

❖ Gestion de la clé :

K est la clé secrète de 128 bits et chaque octet de K est noté k[i]. EK est la clé étendue et chaque élément de EK représente deux octets et noté EK[i], K [0……15] est copie dans EK [0……..7] puis l'extension de la clé est produite à partir de EK [0……..7] en utilisant une fonction FI et est stocké dans Ek [8…….15].

❖ Le chiffrement :

Le chiffrement utilise deux fonctions :

• La fonction F0 :

La fonction prend deux paramètres. Le premier est une entrée de 32 bits, l'autre est un index de EK noté k. F0 renvoie un buffer de 32 bits de données .

- **La fonction FL :**

L'entrée de la fonction FL comporte 32 bits et une sous-clé qui est divisée en deux sous-clés de 16 bits $KL_{i,1}, KL_{i,2}$ avec $KL_i = KL_{i,1} \parallel KL_{i,2}$ l'entrée I des données est divisé en deux parties de 16 bits : L et R avec I= $L \parallel R$ .la sortie de la fonction FL sera les 32 bits $L' \parallel R'$.

$\parallel$: Concaténation de deux opérandes.

II.1.8 RC4 : [4]

Le RC4 a été conçu en 1987 par Ron Rivest ; il fait partie des algorithmes dits de chiffrement en continu. Il est basé sur des permutations aléatoires, avec des opérations sur des octets.

Le RC4 est employé dans la sécurité des réseaux sans fil (WEP dans le WIFI) ainsi que dans plusieurs applications commerciales, par exemple dans le protocole SSL et dans Oracle Secure SQL. Il s'exécute très rapidement dans les logiciels.

L'algorithme a une longueur de clé variable (de 1 à 256 octets). La clé est utilisée pour initialiser une "table d'états" de 256 octets. La table d'état est employée pour la génération d'octets pseudo-aléatoires et ensuite pour produire le flux pseudo-aléatoire avec lequel le texte clair sera transformé avec l'opération de l'OU-Exclusif.

Pour le chiffrement deux étapes sont nécessaires : l'initialisation à l'aide de la clé et le chiffrement du texte clair.

La première étape génère deux tableaux de 256 octets en fonction de la clé : un tableau K initialisé avec les octets de la clé et un tableau P (appelé table d'états, laquelle sera le flux appliqué sur le texte clair) initialisé avec les nombres de 0 à 255 permutés pseudo-aléatoirement selon le tableau K.

Exemple des tableaux (clé de 40 bits).

K=clé(0),clé(1),clé(2),...,clé(39),clé(0),clé(1),...,clé(39)}

P={34,55,228,0,...,4}

La deuxième étape consiste à des permutations pour effectuer le chiffrement. À noter que les additions sont toutes exécutées modulo 256. Le chiffrement est relativement simple.

Le RC4 est populaire en étant extrêmement rapide (environ dix fois plus rapide que le DES), et relativement sûr.

II.1.9 RC2 : [4]

Le RC2 a été conçu en 1989. Il avait été programmé pour être efficace avec les processeurs de 16 bits comme remplacement au DES. Il opère sur des blocs de 64 bits. La grandeur de la clé est variable, de 1 octet (8 bits) à 128 octets (1024 bits). Habituellement, l'algorithme s'applique avec une clé de 64 bits.

Le procédé nouveau qu'a apporté cet algorithme a été d'offrir aux utilisateurs la possibilité de choisir la longueur de la clé. Cette propriété est maintenant offerte dans plusieurs chiffrements par blocs, qu'est importante dans les applications commerciales.

Il y a deux parties dans l'algorithme, soit une procédure d'expansion de la clé et une procédure de chiffrement. Ces deux parties utilisent une table de substitution (S-Box) qui spécifie une permutation aléatoire d'entiers de 0 à 255.

Le chiffrement est défini en gros par 2 opérations, appelées MIX et MASH, sur 4 vecteurs de 16 bits contenant au total 64 bits de texte clair.

- MIX : fonction de permutations et de substitutions ;
- MIXING : 4 fonctions MIX appliquées à chacun des 4 vecteurs de 16 bits ;
- MASH : fonction de permutations et de substitutions ;
- MASHING : 4 fonctions MASH appliquées à chacun des 4 vecteurs de 16 bits.

Les principales étapes suivais pour le chiffrement sont comme suit :

- Initialiser les 4 vecteurs de 16 bits chacun, ils contiennent un bloc de 64 bits de texte clair ;
- Procéder à l'expansion de la clé : initialiser les 64 sous-clés avec la clé ;

- Effectuer 5 fois la fonction MIXING (5 rondes) ;
- Effectuer 1 fois la fonction MASHING (1 ronde) ;
- Effectuer 6 fois la fonction MIXING (6 rondes) ;
- Effectuer 1 ronde de fonction MASHING (1 ronde) ;
- Effectuer 5 fois la fonction MIXING (5 rondes).

II.1.10 RC5 : [4]

Le RC5 a été conçu en 1995. Il a l'avantage d'avoir une longueur de bloc de données variable, un nombre de rondes variable et une clé de longueur variable. Ainsi, l'utilisateur a le contrôle sur le rapport entre la vitesse d'exécution et la sécurité de son chiffrement. En général, une longue clé et un nombre élevé de rondes assurent une plus grande sécurité. La taille des blocs de données pour sa part accommode différentes architectures de systèmes.

La simplicité de l'algorithme du RC5 rend son implémentation facile et, le plus important, rend son analyse plus aisée. De plus, la forte utilisation des décalages de bits (appelés rotations) dans le chiffrement prévient l'usage de la cryptanalyse linéaire et différentielle.

En plus du mode ECB (Electronic Code Book), le RC5 est aussi utilisé avec le mode CBC (Cipher-Block Chaining).

a. Chiffrement :

Pour le chiffrement il existe deux parties : soit une procédure d'expansion de la clé et une procédure de chiffrement, les opérations utilisées sont l'addition modulo $2^{NBRDEBITS}$, le ou exclusif (XOR) et un décalage de bits vers la gauche.

b. En équation :

Soient K [0], K [1], ..., K[n] les sous-clés dérivées de la procédure d'expansion de la clé et A et B les deux parties d'un bloc de texte clair à chiffrer.

A=A+K [0],

B=B+K [1],

Pour i allant de 1 jusqu'au nombre de rounds

A = ((A XOR B) <<< B) + K [2i]

B = ((B XOR A) <<< A) + K [2i + 1]

Fin Pour**.**

II.1.11 RC6 : [4]

RC6 est un algorithme de chiffrement par bloc publié en 1998 et dérivé de RC5. Il se base sur un bloc de 128 bits et supporte des clés de 128, 192 et 256 bits.

RC6 est similaire à RC5 dans sa structure de part la présence de rotations qui dépendent des données, les opérations d'addition modulaire et de XOR. En fait, RC6 pourrait être considéré comme deux chiffrements RC5 entrelacés. Une modification apparaît dans RC6 : il utilise une opération de multiplication absente dans RC5, ceci a pour but de rendre la rotation dépendante de chaque bit du mot, au lieu d'une dépendance concernant uniquement quelques bits de poids faible.

En équation :

Soient K[0], K[1], ..., K[n] les sous-clés dérivées de la procédure d'expansion de la clé et A, B,C et D les quatre parties d'un bloc de texte clair de 128 bits à chiffrer.

B = B + K [0],

D = D + K [1]

Pour i allant de 1 jusqu'au nombre de rounds "r"

t = (B x (2B + 1)) <<< $\log_2$ (nombre de bits des blocs);

u = (D x (2D + 1)) <<< $\log_2$ (nombre de bits des blocs);

A = ((A XOR t) <<< u) + K [2i];

C = ((C XOR u) <<< t) + K [2i + 1];

(A, B, C, D) = (B, C, D, A)

Fin Pour

A = A + K (2r + 2)

C = C + K (2r + 3).

II.1.12 CAST-128: [15]

CAST-128 (ou CAST5) est un algorithme de chiffrement par bloc utilisé par plusieurs logiciels dont certaines versions de PGP et GnuPG.
Une version avec une clé plus grande, CAST-256 (ancien candidat pour AES), a été dérivée à partir de CAST-128.

CAST-128 est basé sur un réseau de Feistel de 12 ou 16 tours avec un bloc de 64 bits. La taille de la clé varie entre 40 et 128 bits (par incrément de 8 bits). La version complète avec ses 16 tours est utilisée quand la clé est supérieure à 80 bits. L'architecture interne du chiffrement comprend des S-Boxes de 8x32 éléments dont le contenu provient de fonctions dites courbe, des rotations qui varient selon la clé, des additions et des soustractions. Il y a trois types de tours mais ils ne varient que sur le choix exact de l'opérateur (addition, soustraction ou XOR).

II.1.13 SEAL : [4]

SEAL (Software-optimized Encryption Algorithm) est un algorithme de chiffrement par flux**,** conçu pour IBM en 1993.Il travaille sur deux phases : une phase d'initialisation et une phase de chiffrement.

Dans sa phase d'initialisation, il utilise l'algorithme SHA pour générer pseudo-aléatoirement les valeurs des tables utilisées pour former le flux à l'aide de la clé secrète.

Son fonctionnement de base est similaire aux chiffrements de flux populaires : un flux est généré et l'opération du OÙ-Exclusif est appliquée entre le flux et le texte clair pour former le texte chiffré. Il est important que le flux soit unique à chaque utilisation de l'algorithme.

L'algorithme emploie beaucoup d'équations à deux opérandes (A = A + B) plutôt qu'à trois (A = B + C). Ainsi il y a une économie de mémoire. Le chiffrement est une série d'équations simples utilisant les opérations ET, OU, OU-Exclusif et le décalage de bits vers la droite.

II.2 Algorithme asymétrique :

Ce sont des algorithmes basés sur le chiffrement à clé publique et déchiffrement à clé privée.

Le concept de cryptographie à clé publique fut inventé par Whitfield Diffie et Martin Hellman en 1976, dans le but de résoudre le problème de distribution des clés posé par la cryptographie à clé secrète. De nombreux algorithmes permettant de réaliser un cryptosystème à clé publique ont été proposés depuis. Ils sont le plus souvent basés sur des problèmes mathématiques difficiles à résoudre. Seuls quelques algorithmes sont utilisables à la fois pour le chiffrement et pour la signature : RSA, El Gamal et …

II.2.1 RSA : [9]

Le RSA est le plus célèbre et le plus répondu des algorithmes asymétriques. Il Fut conçu par Ron Rivest, Adi Shamir et Len Alderman, en 1977. Cet algorithme est basé sur la factorisation des nombres premiers.

Grâce à sa théorie, le RSA sert aussi bien à effectuer le chiffrement des données de taille réduite mais permet également d'assurer le service d'authentification.

II.2.1.a Principe de RSA :

Ce système repose sur la génération d'une paire de clé : la clé K_C pour le chiffrement ou la signature des données confidentielles, qu'est publique et la clé K_D pour le déchiffrement qu'est privée.

L'émetteur chiffre le message à envoyer avec la clé publique du destinataire, à la réception ce dernier le déchiffre avec sa clé privée.

Grâce à ce procédé de chiffrement, tous les correspondants ont librement accès aux clés publiques qui sont stockées dans des annuaires, alors que la clé secrète est générée localement par chaque correspondant et n'est jamais communiquée.

II.2.1.b Génération des clés :

Choisir au hasard deux grands nombres premiers *P* et Q. Il faut que P et Q contiennent au moins 100 chiffres décimaux chacun.

- Calculer N le produit des deux nombres N=P.Q, il est difficile de retrouver P et Q de N, (P et Q sont gardés secrets).
- Calculer le nombre d'Euler du nombre N : φ(N)= (P-1) (Q-1).
- Choisir un nombre entier $K_C \langle \varphi$**(**N) et premier avec φ(N).
- Calculer K_D tel que $K_D = K_C^{-1} \bmod \varphi$(N).
- Choisir un des deux nombres (K_C, K_D) comme clé secrète K_S et l'autre comme clé publique K_P.

On aura alors :

Le chiffrement C est obtenu par :

$$C = M^{K_C} \bmod N$$

Pour le déchiffrement on obtient le message clair en calculant :

$$C^{Kd} \bmod N \equiv M^{Kc.Kd} \bmod N \equiv M$$

Exemple : chiffrement de BONJOUR

A veut envoyer le message «Bonjour » à **B**, sans que personne ne le sache. **B** doit au préalable construire ses clés publique et privée :

Par exemple il choisit p = 79, q = 127.

Ainsi, N = p × q = 79 × 127 = 10 033, et (p-1) × (q-1) = 78 × 126 = 9828.

Plusieurs valeurs sont possibles pour K_c

Disons qu'il prend K_c = 97.

Par un calcul, il trouve que K_D = 2533**,**

puisque 2533 × 97 ≡1 (mod 9828). (Pour le calcul, il faut utiliser l'algorithme d'Euclide, voir annexe).

En suite **B** envoie les nombres N = 10 033 et K_c = 97 à **A** pour qu'il puisse lui envoyer le message de manière confidentielle. Maintenant **A** peut chiffrer le message :

D'abord, il convertit le message en nombres, en associant à chaque lettre sa position dans l'alphabet. Ainsi,
B=02, O=15, N=14, J=10, U=21, R=18.

Et,

B O N J O U R = 02 15 14 10 15 21 18, qu'on notera **M** pour alléger. Il sépare ensuite le message **M** en blocs de 4 chiffres : 0002 1514 1015 2118.

A cadence de façon mathématique le message.

$2118^{97} = 9253(\text{mod}10033)$;
$1015^{97} = 8000(\text{mod}10033)$;
$1514^{97} = 2500(\text{mod}10033)$;
$0002^{97} = 9208(\text{mod}10033)$.
Le message codé est donc M^{Kc} = 9208 2500 8000 9253.

II.2.1.c Performances de RSA :

La sécurité du RSA repose essentiellement sur la difficulté à factoriser de grands nombres premiers. En effet, si un attaquant factorise N= P*Q de la clé publique, il peut alors déduire directement $\varphi(N) = (P-1)(Q-1)$ et donc calculer la clé privée à partir de la clé publique par l'algorithme d'Euler étendu (voir annexe). Donc si on dispose d'un algorithme rapide pour factoriser de grands entiers, casser le RSA devient facile.

Après plus de vingt ans de recherche, aucun moyen plus efficace que la factorisation de N n'a été publié pour casser le RSA.

II.2.2 Protocole d'échange de clé publique Diffie_ Hellman : [9]

C'est en effet la première technique qui a été proposée pour constituer un système de clé publique, basé sur la difficulté de calcule du logarithme discret.

La notion du logarithme discret peut être définie de la manière suivante **:**
Considérons d'abord la racine primitive d'un nombre p qui désigne un nombre dont les puissances générées sont tous les nombres compris entre 1 et p-1. Ce qui veut dire, si *a* est

une racine primitive du nombre p, les nombres $a \ mod \ p$, $a^2 \bmod p, ... a^{p-1} \bmod p$ sont distincts et constitué d'entiers de 1 à p-1 :

Pour tout entier *b* et une racine primitive a d'un nombre premier p, on peut trouver un exposant unique *i* tel que : $b = a^i \bmod p$ avec $0 \leq i \geq (p-1)$. L'exposant i ainsi défini désigne le logarithme discret de *b* pour la base a, mod p.

II.2.2.a Principe :

Soit p un nombre premier et α < p sa racine primitive et soit X, Y deux parties communicantes qui vont échanger leurs clés.

X choisit un nombre secret *a* tel que *a* < p et calcule sa clé publique $A = \alpha^a \bmod p$, Y choisit pour sa part un nombre *b* secret pour calculer sa clé privée $B = \alpha^b \bmod p$.

X et Y s'échangent leurs clés publiques puis chacun calculera sa clé privée.

X: $K = B^a \bmod p$;

Y: $K = A^b \bmod p$.

II.2.3 EL GAMAL : [8]

C'est un algorithme à clé publique présent à la base de la norme U.S de signature électronique. Il fut inventé par Taher El Gamal en 1984. Il est basé sur la difficulté de calculer des logarithmes discrets.

II.2.3.a Principe :

-Soit P un entier très grand et P-1 doit avoir un grand facteur premier ;

- Soit g $\in Z^*_P$ un élément primitif ;
- Soit s un nombre et $\beta = g^S$;
- La clé publique est alors le triplet $K_E = (P, g, \beta)$;
-La clé secrète est le nombre $K_D = s$.

Chiffrement : soit M le texte clair à chiffrer et k $\in Z_{P-1}$ un nombre aléatoire secret :

$E_{Ke}(M) = (y_1, y_2)$. Avec $\begin{cases} y_1 = g^k \bmod p \\ y_2 = M\beta^K \bmod p \end{cases}$

Déchiffrement : pour $y_1, y_2 \in Z^*_P$, on définit :

$$D_{Kd}(y_1, y_2) = y_2.(y_1^S)^{-1}$$

En effet, $y_2.(y_1^S)^{-1} = M.\beta^k.(g^{k.S})^{-1} = M.g^{S.k}.(g^{k.S})^{-1} = M$

II.2.4 protocol PGP (pretty good privacy): [13]

Le PGP est la solution la plus connue des usagers pour rendre confidentielle la transmission de message et authentifier l'émetteur. Elle fut développée et mise à la disposition sur internet dés 1991 par son auteur P Zimmermann.

PGP combine à la fois les meilleures fonctionnalités de la cryptographie conventionnelle (symétrique) et de la cryptographie à clé publique. PGP est un cryptosystème hybride.

Il s'agit d'une solution qui s'exécute sur un grand nombre de plates-formes dont Windows, Unix, Linux …elle est disponible gratuitement.

Quand un utilisateur chiffre du texte clair avec PGP, ce dernier compresse d'abord le texte clair. La compression de données économise du temps de transmission par modem et de l'espace disque et, ce qui est plus important, il renforce la sécurité cryptographique. La plupart des techniques de cryptanalyse exploitent les redondances trouvées dans le texte clair pour craquer le texte chiffré. La compression réduit ces redondances dans le texte clair, ce qui augmente grandement la résistance à la cryptanalyse. (Les fichiers qui sont trop petits pour être compressés ou qui ne se compressent pas bien ne sont pas compressés.)

PGP crée ensuite une clé de session, qui est une clé secrète qui ne sert qu'une fois. Cette clé est générée d'une manière aléatoires, elle fonctionne avec un algorithme de chiffrement conventionnel très sûr et rapide (IDEA) qui chiffre le texte clair; le résultat est le texte chiffré. Une fois que les données sont chiffrées, la clé de session est elle-même chiffrée avec la clé publique du destinataire. Cette clé de session chiffrée par la clé publique est transmise avec le texte chiffré au destinataire.

Le déchiffrement fonctionne de la manière inverse. La copie de PGP du destinataire utilise la clé privée de celui-ci pour retrouver la clé de session temporaire, que PGP utilise ensuite pour déchiffrer le texte chiffré de manière symétrique.

La combinaison des deux méthodes de chiffrement associe la commodité du chiffrement à clé publique avec la vitesse du chiffrement conventionnel. Le chiffrement conventionnel est environ 1000 fois plus rapide que le chiffrement à clé publique. Le chiffrement à clé publique fournit quant à lui une solution aux problèmes de distribution de la clé et de transmission des données. Utilisées toutes les deux, la performance et la distribution de la clé sont améliorées sans aucun sacrifice sur la sécurité.

II.3 Conclusion :

Il n'y a pas de sécurité absolue, mais des éléments de sécurité relatifs. La recherche d'algorithmes, Protocoles et architectures de sécurité doit donc viser à faire en sorte qu'il ne soit pas rentable par rapport à l'information convoitée.

On constate donc qu'il y a nécessairement des solutions différentes selon la valeur de l'information à transmettre et le choix de l'algorithme est déterminant.

Pour la suit du travail nous allons détailler l'algorithme AES qui répond aux besoins de sécurité et aux différentes contraintes.

III.1 Description : [10]

Comme tout système de cryptographie symétrique l'AES dispose de deux entrées, à savoir le texte clair à chiffrer et la clé de cryptage. L'AES fixe la taille des blocs du texte clair à 128 bits représentés par Nb, (Nb est le nombre de mots de 32 bits dans un bloc) et utilise des clés de longueur de 128, 192 où 256 bits. La longueur de la clé est représentée d'une façon similaire par Nk = 4, 6 ou 8.

L'AES exécute une séquence de rondes qui sont détaillées dans la suite. On note Nr le nombre de rondes, ce nombre dépend des valeurs de Nb et Nk. Les différentes configurations possibles sont illustrées dans le tableau suivant :

	Nk	Nb	Nr
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Détail des configurations possibles.

Dans la suite on prendra une clé de longueur de 128 bis.

L'AES opère sur une matrice de 4×4 notée « state » obtenue par découpage du texte clair en blocs de 128 bits. Ensuite on va mettre chaque bloc du texte et la clé sous forme matricielle de Nk lignes et Nb colonnes. L'unité de base à traiter dans l'algorithme et l'octet (byte), une séquence de 8 bits traité comme une seule entité et touts les éléments sont considérés comme des éléments d'un champ fini : champ de Galois GF (2^8). (voir annexe).

III.2 Le chiffrement :

Le chiffrement consiste en une addition initiale de la clé avec la matrice state, cette opération notée Add_RoundKey, est suivie de Nr-1 rondes, constituée chacune de quatre étapes :

- Sub_Bytes
- Shift_Rows

- Mix_Columns
- Add_RoundKey

Et une ronde finale ou l'étape Mix_Columns est omise.

III.2. a La transformation Sub_Bytes :

L'étape Sub_Bytes correspond à la seule transformation non-linéaire de l'algorithme. Chaque élément de la matrice state est substitué selon une table de substitution notée S_Box (figure III.1). Cette table est construite on composant deux transformations :

1. Prendre l'inverse de l'octet dans $\frac{Z_2(x)}{m(x)Z_2(x)}$, qui est la recherche d'un multiplicatif inverse dans GF (2^8) (i.e. pour tout élément b de GF (2^8) il existe son inverse b^{-1} tel que $b \bullet b^{-1} = 1$). (Voir annexe)

2. Appliquer une transformation affine pour chaque élément trouvé dans $\frac{Z_2(x)}{m(x)Z_2(x)}$, qui est donnée par la fonction suivante :

Pour $0 \leq i < 8$

$$S' = S_i \oplus S_{(i+4) \bmod 8} \oplus S_{(i+5) \bmod 8} \oplus S_{(i+6) \bmod 8} \oplus S_{(i+7) \bmod 8} \oplus C_i$$

où S_i est le i ^{ème} bit de l'octet et c_i le i ^{ème} bit d'un octet c qui vaut $(01100011)_2$ $(63)_H$.
Cette transformation affine peut prendre la forme matricielle suivante :

$$b = f(a) \Longleftrightarrow \begin{bmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Pour la substitution on procède de deux manières : soit on prend chaque élément de la matrice state et on le considère comme étant un polynôme de GF (2^8). Ensuite on prend son inverse puis on calcule l'image du résultat obtenu par la fonction affine définie précédemment, soit on substitue directement chaque élément de la table de substitution pré calculée tel que le premier caractère en hexadécimal de chaque élément indique une ligne de la S_box tandis que le deuxième indique la colonne, l'intersection de la lige et de la colonne donne l'élément à substituer.

Cette opération sera assurée par la fonction Sub_Bytes et on choisit la deuxième méthode qui est plus rapide et exécutable en temps réel.

La fonction Sub_Bytes consiste à substituer chaque élément de la matrice state à partir de la table S_Box, où cette dernière est générée par la fonction S_Box_gen. La fonction S_Box_gen fait appel à d'autres fonctions : Find_inverse (pour trouver l'inverse), Affin_trans (transformation affine) et une fonction S_Box_inversion pour générer l'inverse de la S_Box nécessaire pour le déchiffrement.

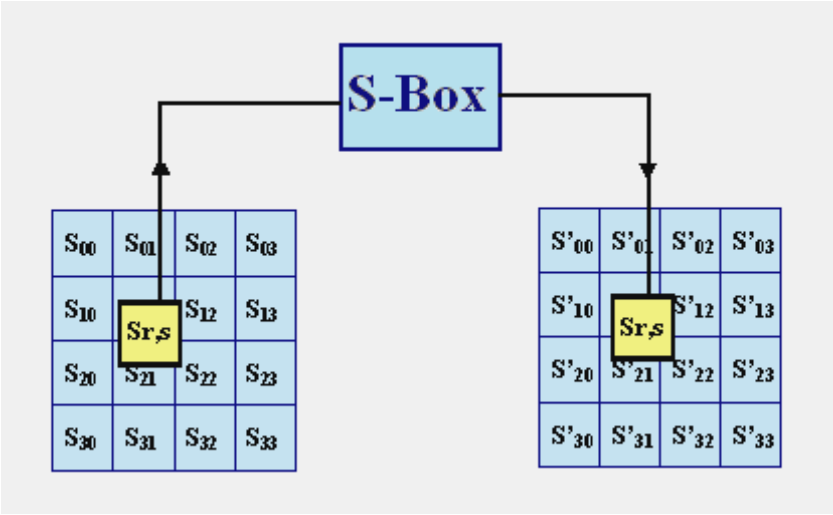


Figure III.1 Principe de la fonction Sub_Bytes.

III.2.b La transformation Shift_Rows :

Elle permet de diffuser l'information entre les colonnes. Shift_Rows applique une rotation circulaire vers la gauche sur chaque ligne de la matrice (figure III.2). La valeur de la rotation dépend de la taille des blocs d'entrée et du numéro de la ligne.

Pour $0 < r < 4$ et $0 \leq s < Nb = 4$

$$S''_{rs} = S'_{r(c+shift(r,4) \bmod 4)}$$

Avec : shift (1,4) =1, shift (2,4) =2, shift (3,4) =3.
Les décalages se font comme suit :

- La première ligne n'est pas décalée.
- La deuxième ligne est décalée de 1 octet vers la gauche.
- La troisième ligne est décalée de 2 octets vers la gauche.
- La quatrième ligne est décalée de 3 octets vers la gauche.

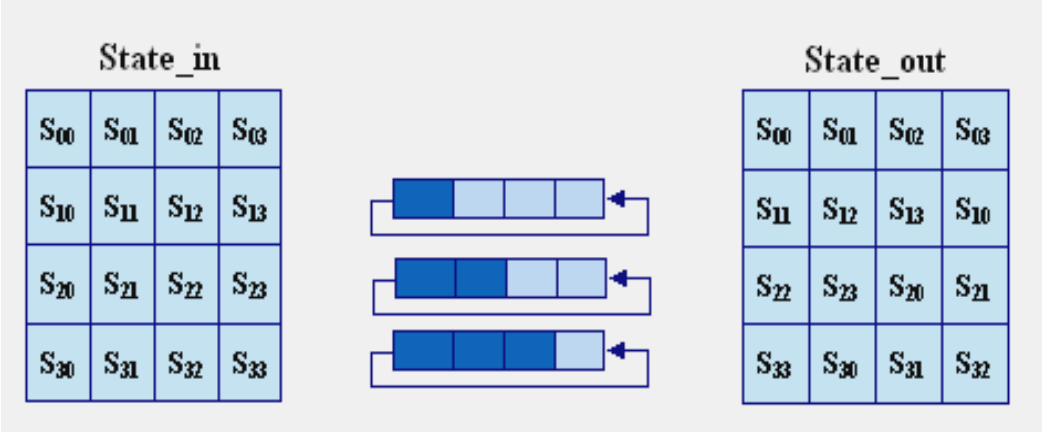


Figure III.2. Principe de la fonction Shift_Rows.

Son implémentation est assurée par la fonction Shift_Rows qui fait appel à une fonction de décalage cycle. La fonction cycle permute cycliquement les lignes d'entrée dans un sens avec un nombre de positions déterminées, tel que la première ligne n'est pas décalée, la deuxième est décalée d'une seule position vers la gauche, la troisième et la quatrième sont décalées respectivement de deux et trois positions.

III.2.c La transformation Mix_Columns :

La transformation Mix_Columns traite chaque colonne comme un polynôme S(x) de degré 3 à coefficients dans GF (2^8). Cette étape consiste alors à effectuer pour chaque colonne une multiplication avec un polynôme *a(x)* fixe, suivie d'une réduction modulo le polynôme $X^4 + 1$. (Figure III.3).

$$a(x) = (03)x^3 + (01)x^2 + (01)x + (02).$$

$$S'(x) = (03x^3 + 01x^2 + 01x + 02) \times S(x) \bmod (x^4 + 1).$$

Cette opération s'écrit :

$$\begin{pmatrix} S'_0 \\ S'_1 \\ S'_2 \\ S'_3 \end{pmatrix} = \begin{pmatrix} 03 & 01 & 01 & 02 \\ 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \end{pmatrix} \times \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{pmatrix}$$

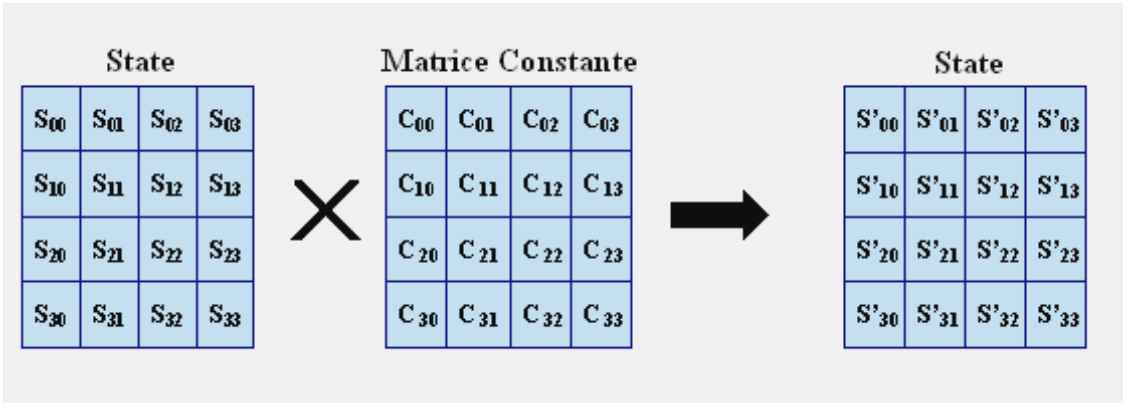


Figure III.3 Principe de la fonction Mix_Columns.

L'implémentation de cette fonction est basée sur les propriétés mathématiques de champs de Galois.

La fonction Mix_Columns consiste à faire une multiplication entre la matrice state et la matrice polynomiale constante poly_mat a généré et de même pour son inverse.

La matrice poly_mat est générée par la fonction poly_mat_gen et qui génère encore la matrice Inv_poly_mat utilisée dans le processus de déchiffrement (Inv_Mix_Columns).

Les deux matrices poly_mat et Inv_poly_mat sont de dimension 4×4 où chaque ligne est une permutation de la précédente suivant la fonction cycle.

III.2.d La transformation Add_RoundKey :

Add_RoundKey permet de combiner la matrice state avec la clé de tour. A partir de la clé maîtresse on dérive des clés de tour qui ont la même taille que la matrice state.

L'application de Add_RoundKey consiste à faire le ou-exclusif entre l'état interne et la clé de tour. (figure III.4).

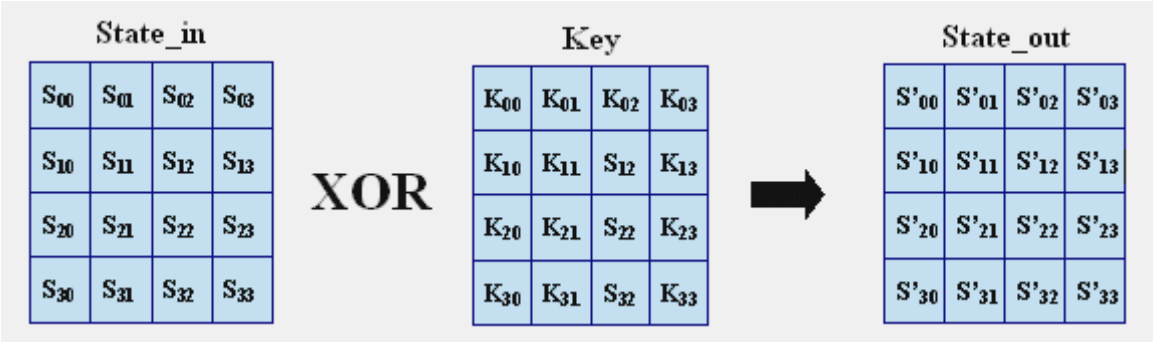


Figure III.4 Principe de la fonction Add_RoundKey.

III.3 Gestion de clés :

Pour assurer le cryptage des données, l'algorithme AES nécessite 11 clés : la clé originale et 10 autres clés secondaires dérivées de l'originale.

Pour calculer les clés secondaires, on prend les deux matrices représentées par la figure III.5.

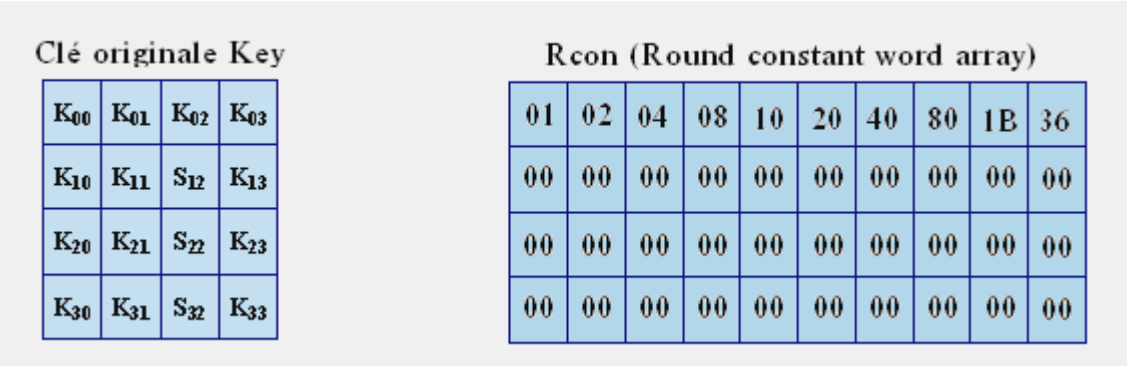


Figure III.5 Présentation de la clé originale et de la matrice constante Rcon.

La matrice Rcon est construite à partir de la formule suivante :

$$Rcon[i] = [X^{i-1}, 00, 00, 00]$$

Pour $i \leq 1$, (X=02). On utilisant les propriétés mathématiques de GF (2^8). (voir annexe).

Les quatre vecteurs composant la matrice Key sont appelés Words (*W*); ce sont des mots de 32 bits. Pour calculer le vecteur suivant on a la condition :

- Si mod (i, 4)=1.

Par exemple pour calculer le cinquième vecteur on prend le vecteur W_{i-1} et on va lui appliquer une opération appelée Rot_Word qui consiste en un simple décalage des quatre octets du vecteur vers le haut (figure III.6). Au résultat on va appliquer encore l'opération Sub_word qui n'est que la transformation Sub_Bytes qu'on a déjà défini.

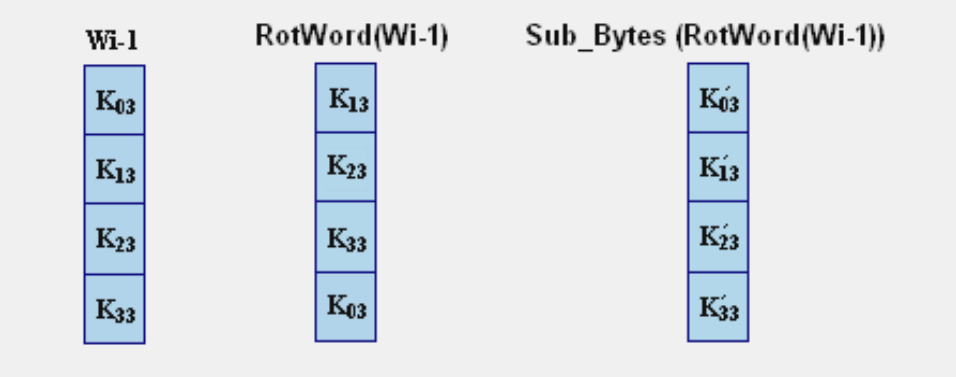


Figure III.6 Première étape de la génération du cinquième vecteur.

Le vecteur obtenu est additionné modulo *m(x)* avec le vecteur W_{i-4} ainsi que le premier vecteur de la matrice Rcon (figure III.7).

La routine Rot_Word prend en entrée un mot de 4 octets $[a_0, a_1, a_2, a_3]$ et lui applique la permutation circulaire $[a_1, a_2, a_3, a_0]$.

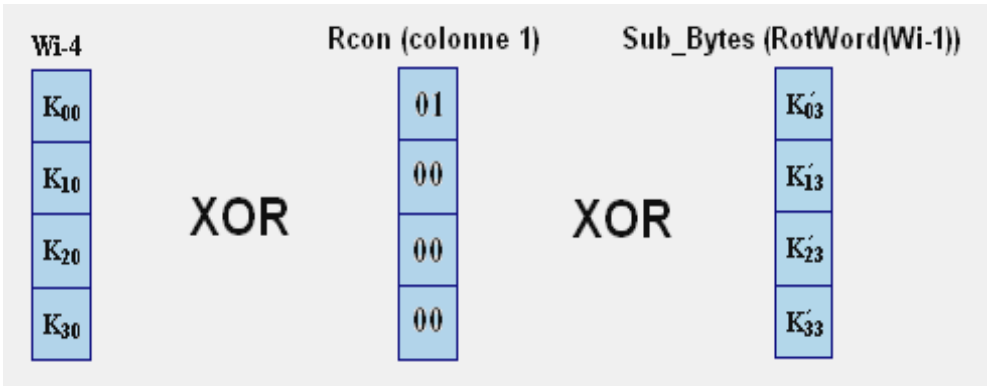


Figure III.7 Deuxième étape de la génération du cinquième vecteur.

Ainsi le vecteur résultant constitue le premier des quatre vecteurs de la première clé secondaire. Il sera le W_{i-1} nécessaire pour le calcul du prochain vecteur.

- Si mod (i, 4) ≠ 1 :

On effectue seulement un ou exclusif entre le W_{i-1} et W_{i-4}.

L'implémentation de l'expansion de la clé est assurée par la fonction Key_expansion qui fait appel à son tour à la clé initiale, la fonction Sub_Bytes, la table S_Box et Rcon générés auparavant ainsi que la fonction Rot_ Word qui décale les éléments de la ligne d'une matrice d'une seule position.

Les éléments de la matrice constante sont tous à zéro à part la première ligne qui contient des multiplicatifs de 2. Rcon est donnée par la fonction de génération rcon_gen.

III.4. Déchiffrement :

La routine de chiffrement peut être inversée et réordonnée pour produire une routine de déchiffrement où Les transformations Inv_Sub_Bytes, Inv_Shift_Rows, Inv_Mix_Columns sont respectivement l'inverse de Sub_Bytes, Shift_Rows et Mix_Columns.

III.4.a La transformation Inv_Sub_Bytes :

L'opération Inv_Sub_Bytes consiste à effectuer la même manipulation que Sub_Bytes

mais à partir de la table inverse S_box, générée par l'inversion de la table S_box. La transformation affine inverse est donnée par:

$$b = f^{-1}(a) \Longleftrightarrow \begin{bmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

III.4.b La transformation Inv_Shift_Rows :

La transformation Inv_Shift_Rows consiste à faire un décalage à droite des trois derniers lignes de la matrice state.

Pour : $0 < r < 4$ et $0 \leq s < Nb=4$.

$$S''_{r,(s+shift\ (r,4)\bmod 4)} = S'_{rs}$$

III.4.c La transformation Inv_Mix_Columns:

La transformation Inv_Mix_Columns est l'inverse de la transformation Mix_Columns, traitant chaque colonne comme un polynôme de degré 3, on calcule le produit de ce polynôme avec un polynôme fixe $a^{-1}(x)$:

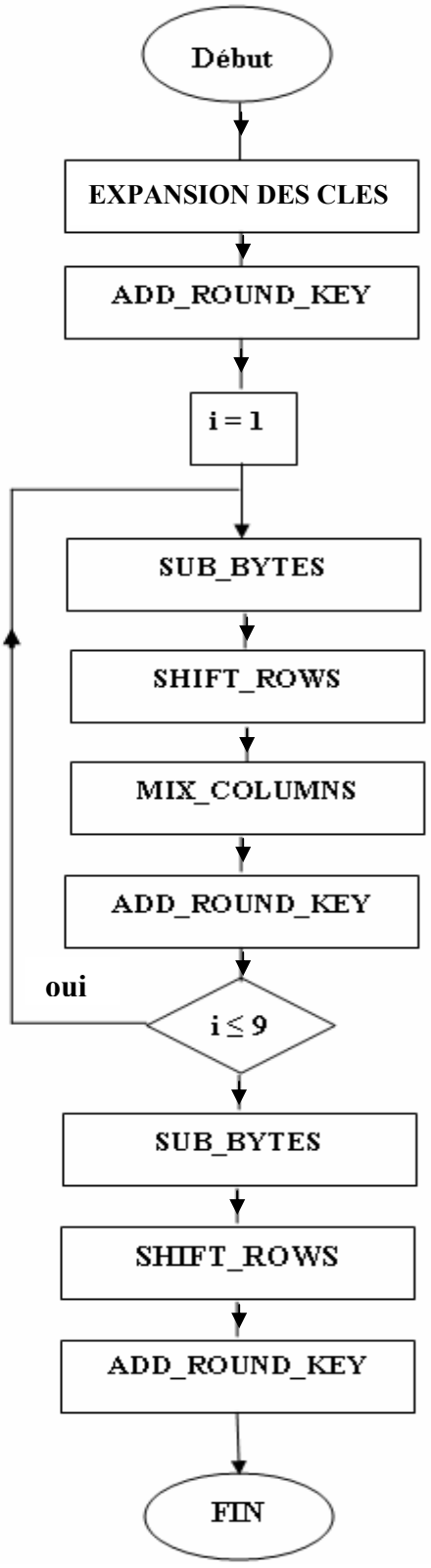
$$a^{-1}(x) = (0\times 0B)X^3 + (0\times 0D)X^2 + (0\times 09)X + (0\times 0E).$$

Ces opérations peuvent être mises sous forme matricielle :

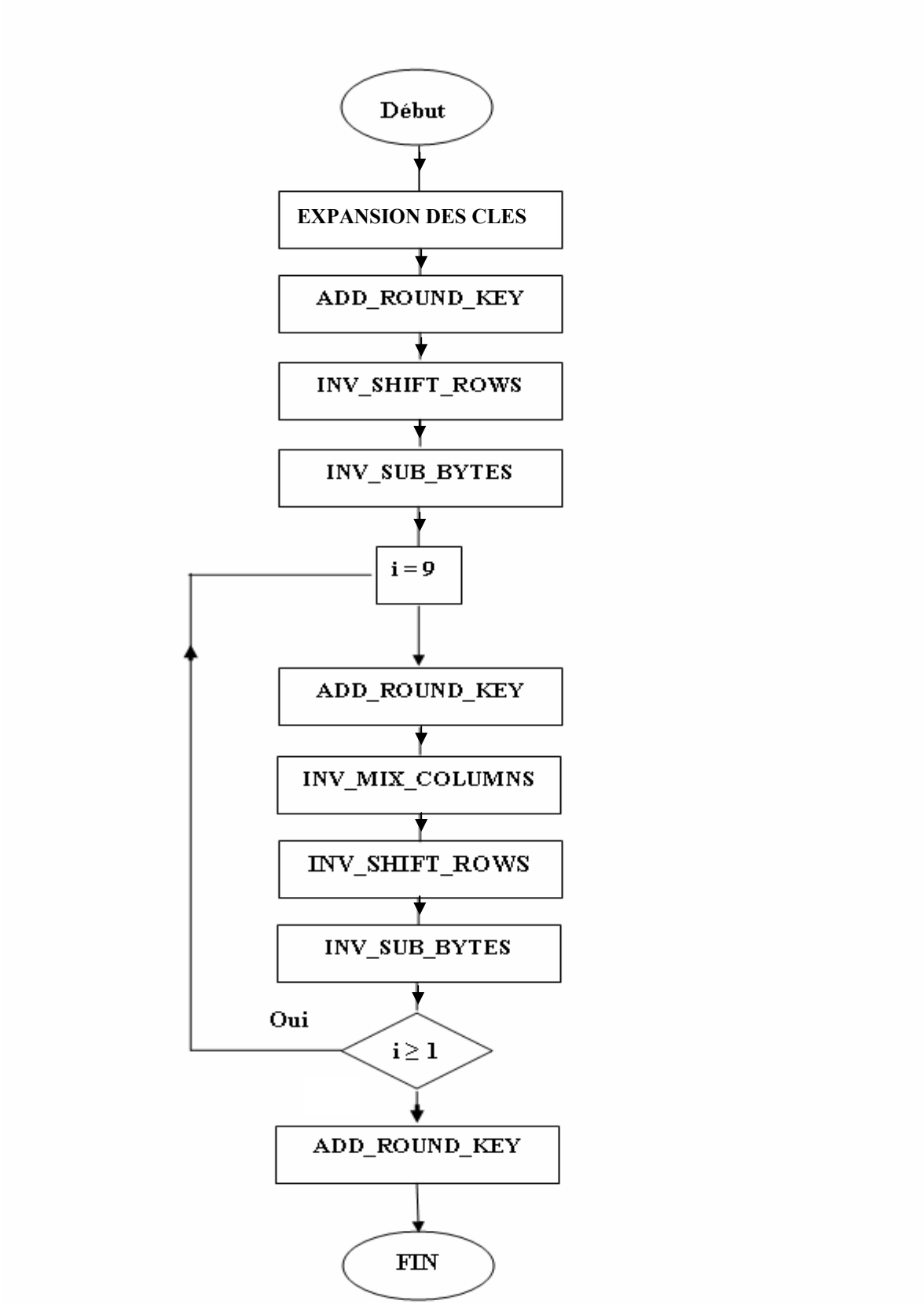
$$\begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \times \begin{pmatrix} S'_0 \\ S'_1 \\ S'_2 \\ S'_3 \end{pmatrix}$$

III.5 Programmation :

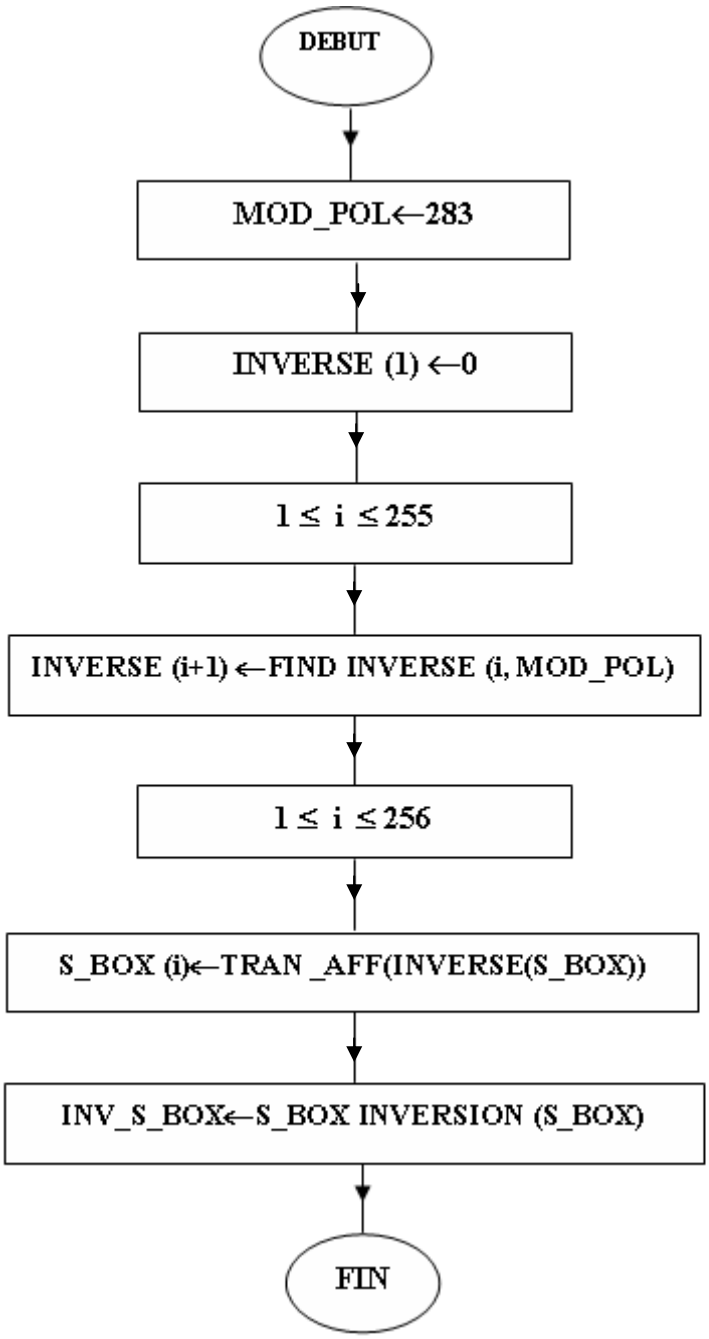
Nous donnons ci après les organigrammes pour les différentes fonctions.



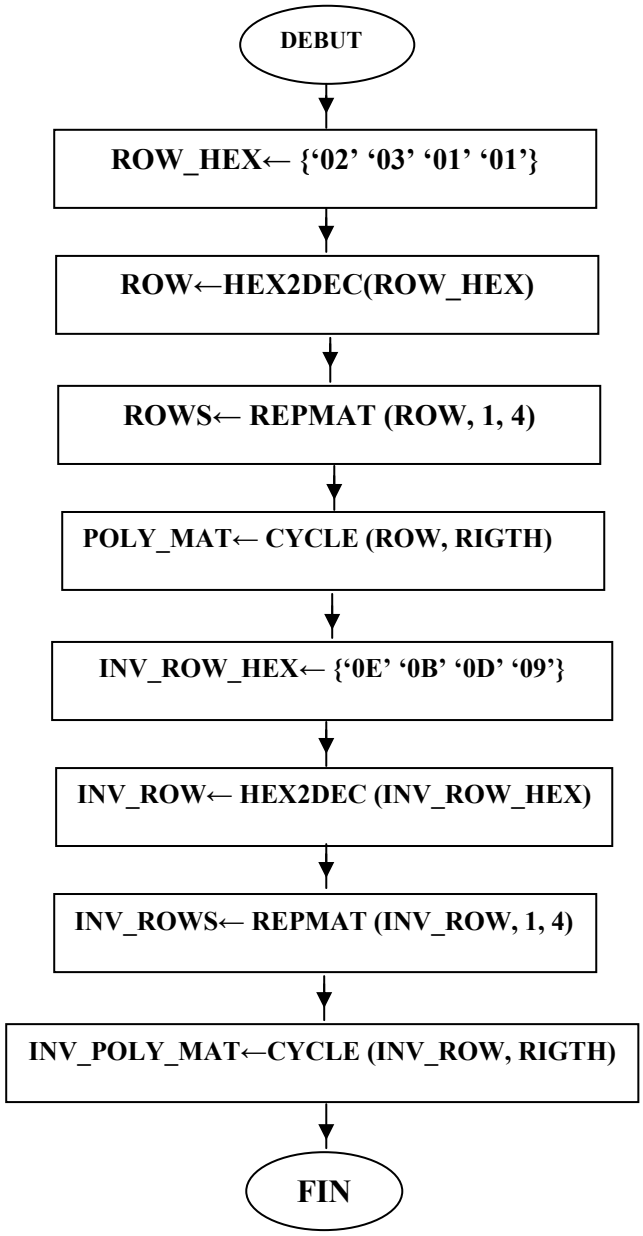
Organigramme de chiffrement.



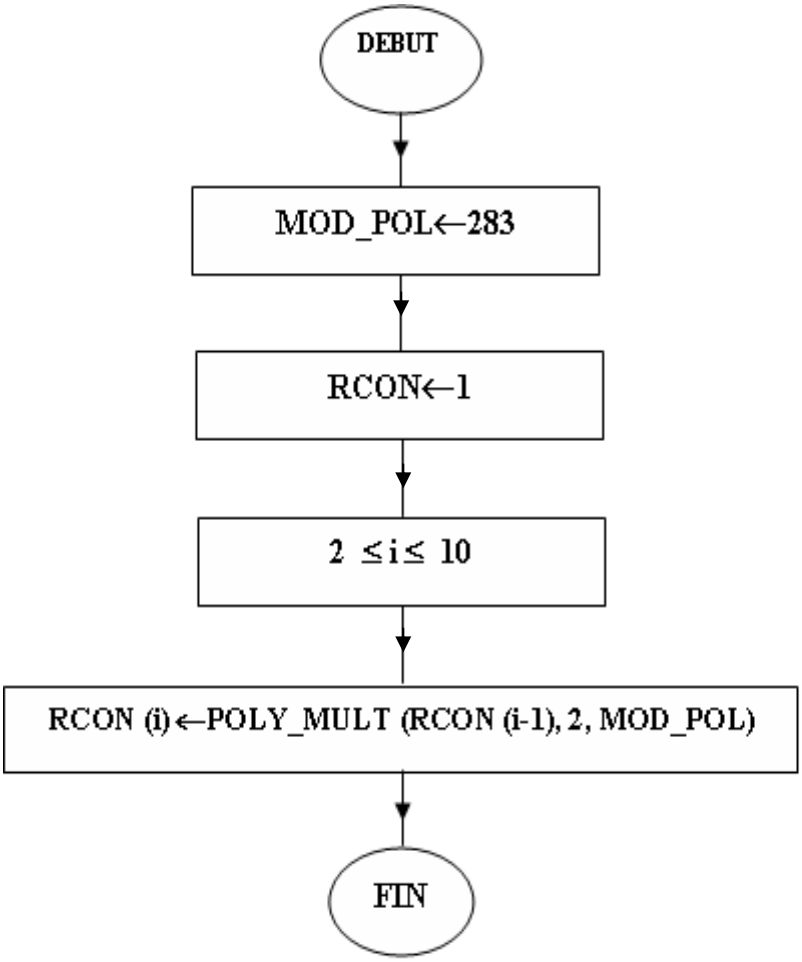
Organigramme de déchiffrement.



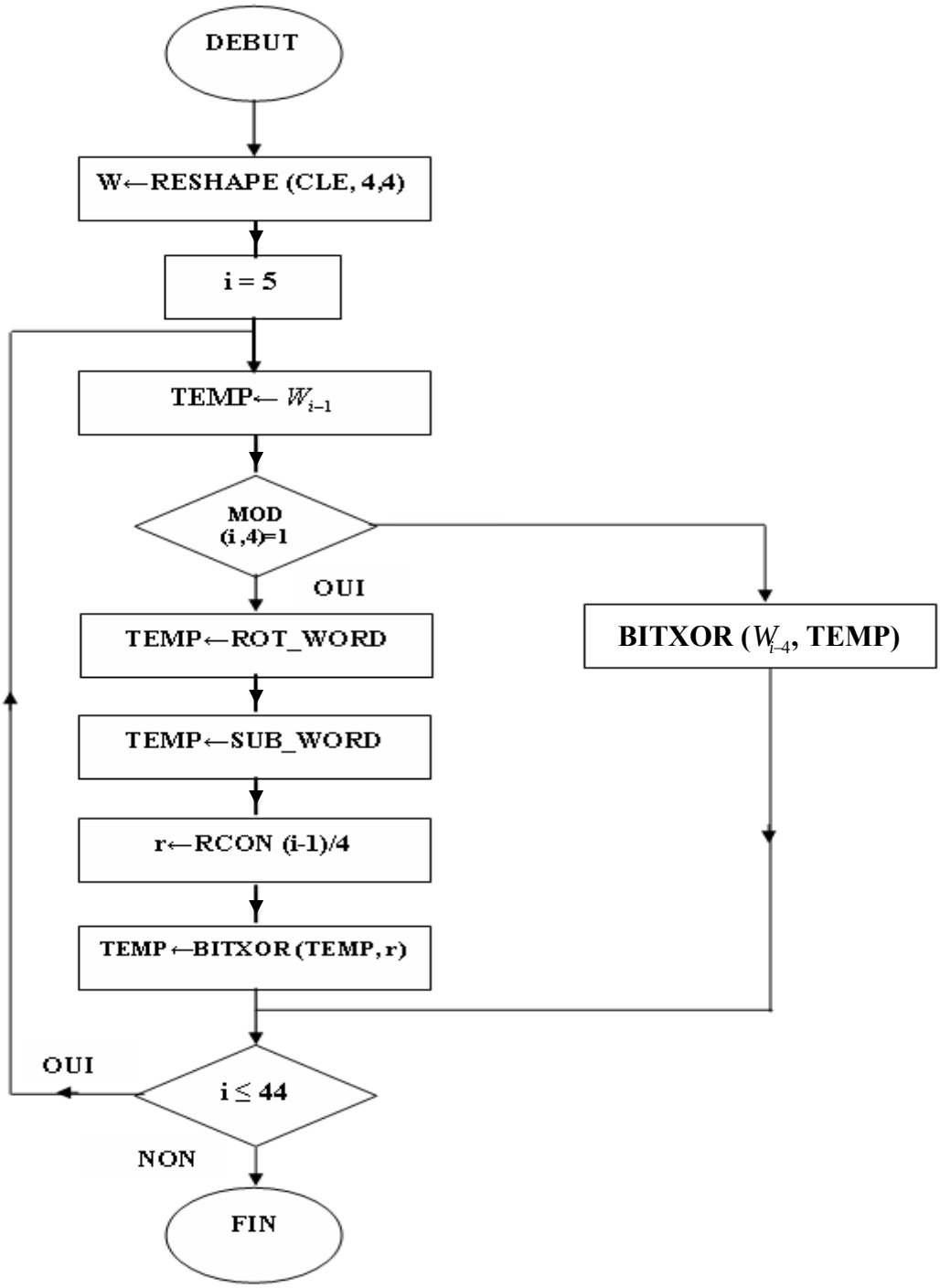
Organigramme de la fonction S_Box_gen.



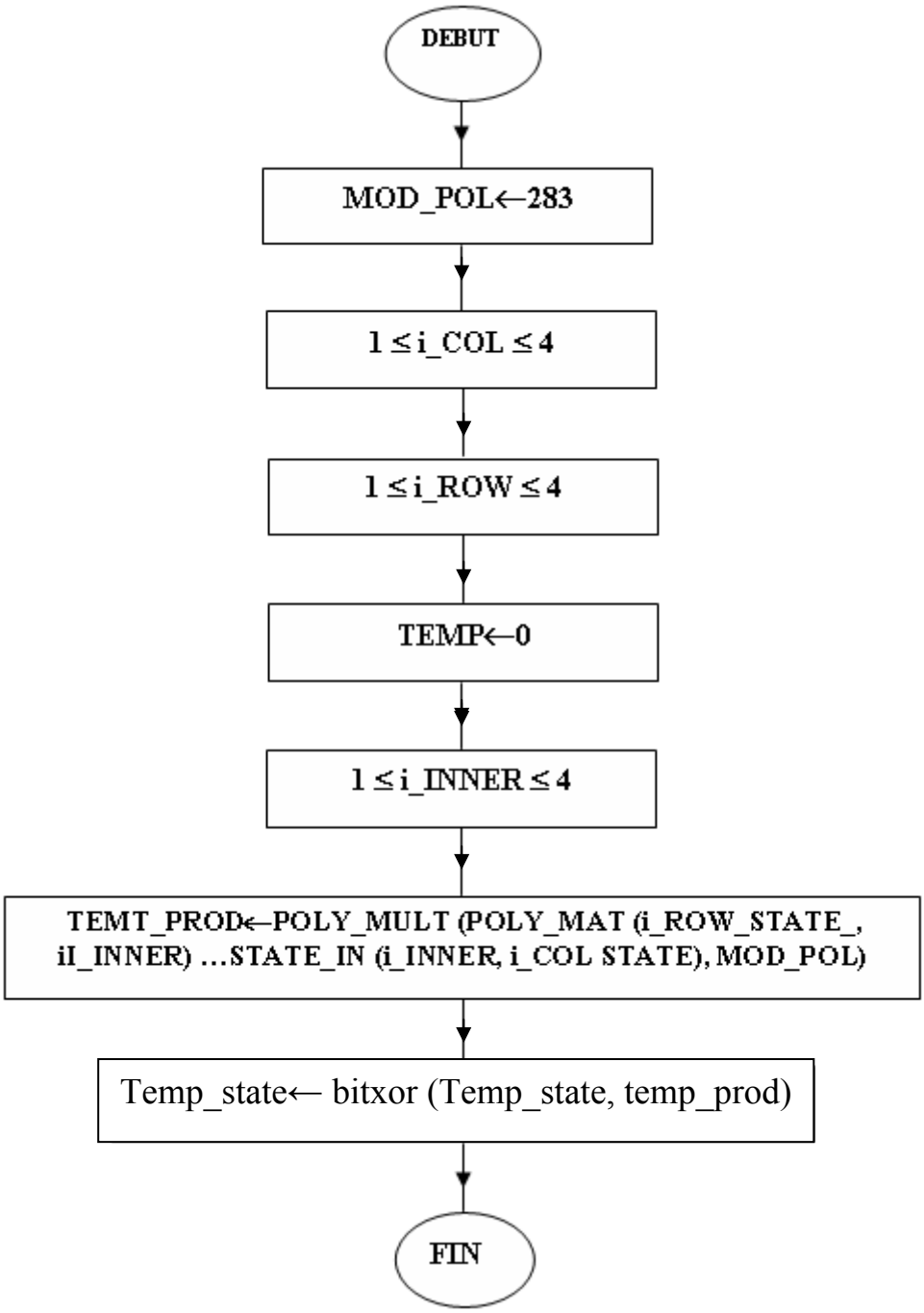
Organigramme de la fonction poly_mat_gen.



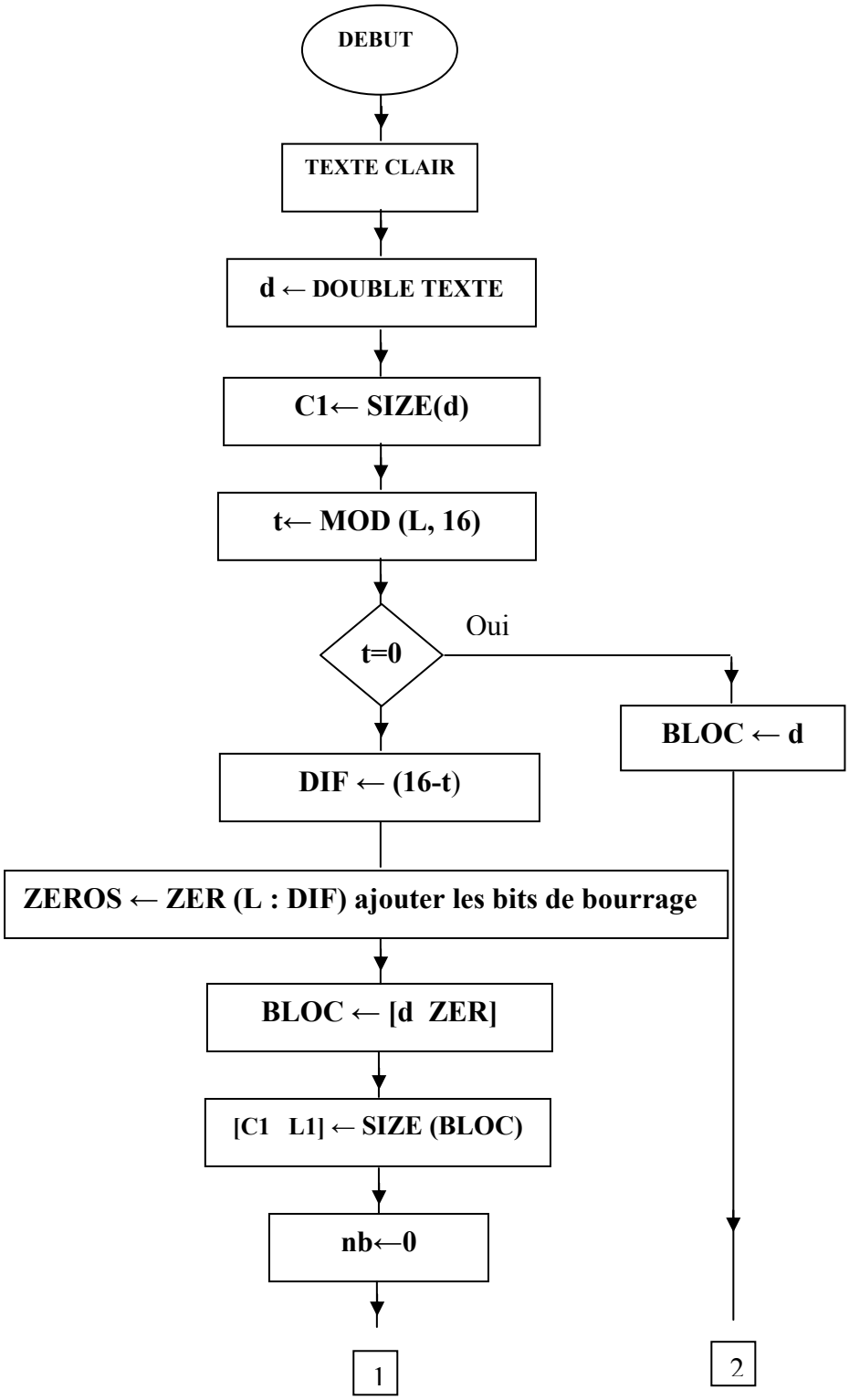
Organigramme de la fonction Rcon _gen.

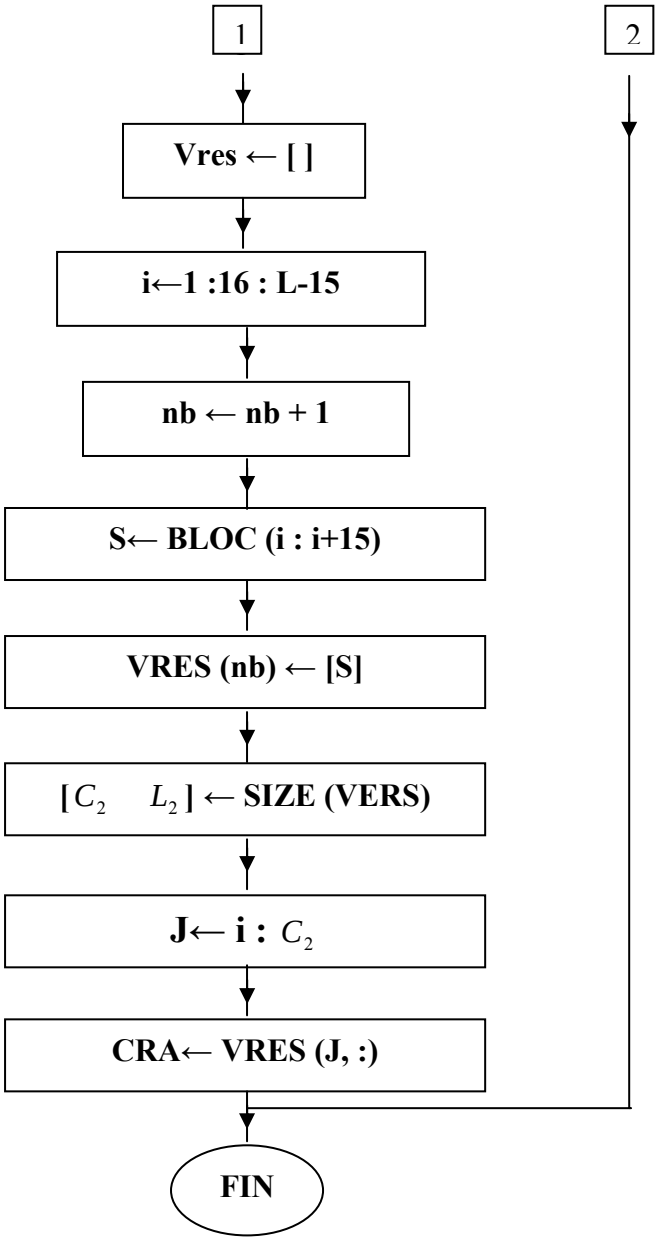


Organigramme d'expansion de la clé.

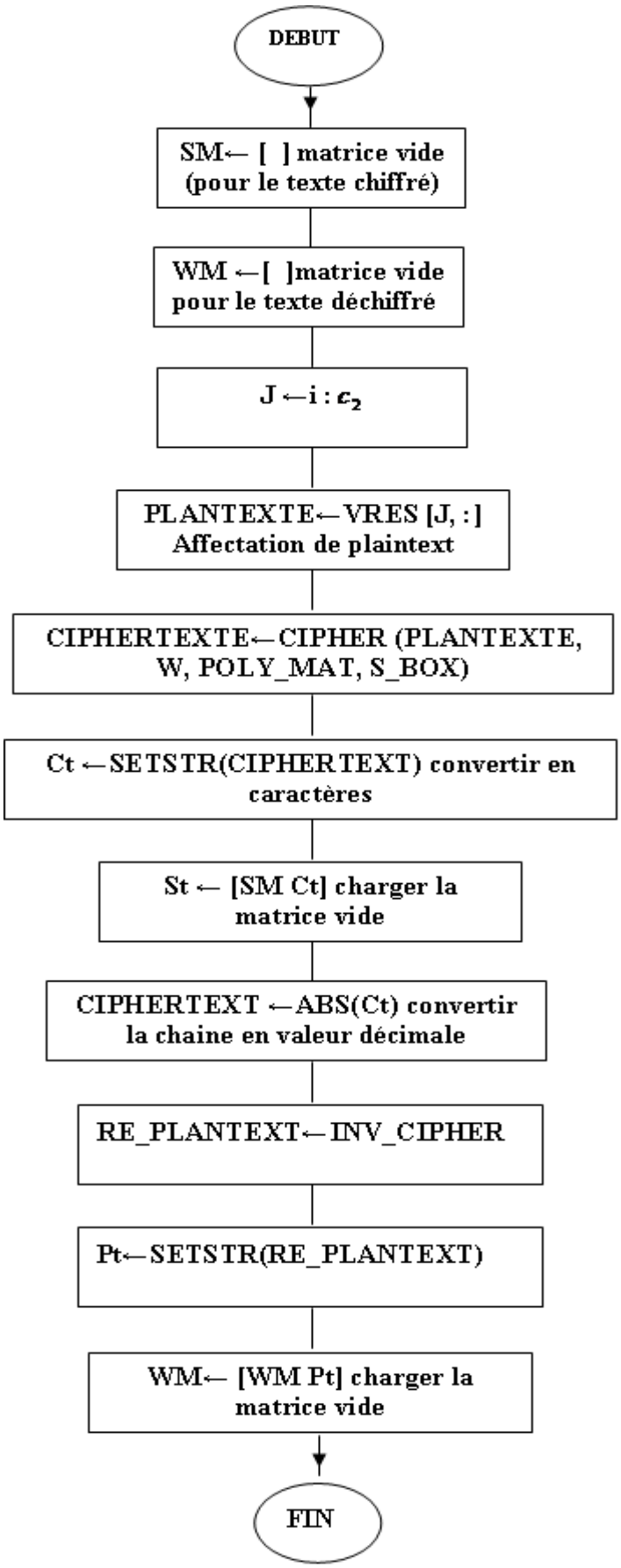


Organigramme de la fonction Mix_columns.


DEBUT
TEXTE CLAIR
d ← DOUBLE TEXTE
C1← SIZE(d)
t← MOD (L, 16)
t=0
Oui
BLOC ← d
DIF ← (16-t)
ZEROS ← ZER (L : DIF) ajouter les bits de bourrage
BLOC ← [d ZER]
[C1 L1] ← SIZE (BLOC)
nb←0
1
2



Organigramme de la fonction de décomposition du texte.



Organigramme de la fonction de démo.

Organigramme de la fonction de démo.

IV.1 APPLICATION :

Dans cette partie on va présenter les résultats donnés par l'implémentation de l'algorithme de cryptage et de décryptage AES sous Matlab.

Pour vérifier le bon fonctionnement de notre algorithme, on l'applique à trois exemples :

1*er* Exemple :

- On a pris les textes clairs du test défini par le standard AES, et on les a cryptés par

l'algorithme qu'on a implémenté puis on les a comparés avec les textes décryptés donnés par le standard.

Entrée du standard = 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34.
Clé de chiffrement = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c.
Sortie du standard = 39 25 84 1d 02 dc 09 fb dc 11 85 97 19 6a 0b 32.

Le résultat obtenu avec notre implémentation est:

<< Texte chiffré

39 25 84 1d 02 dc 09 fb dc 11 85 97 19 6a 0b 32.

En comparant ce qui a été donné par le standard et ce que nous avons obtenue comme résultat, on remarque que notre algorithme fonctionne correctement et suit bien ce qui à été spécifié par le standard.

Pour vérifier maintenant le fonctionnement de l'algorithme de décryptage on va prendre cette séquence et on va lui appliquer l'algorithme de décryptage.

<< Texte déchiffré

32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

Ce résultat montre bien le bon fonctionnement de décryptage puisque le texte déchiffré correspond bien au texte chiffré.

❖ 2eme **Exemple :**

texte clair : 00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff.

clé de chiffrement : 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f.

<< Texte chiffré

69 C4 E0 D8 CA 7B 04 30 D8 CD B7 80 70 B4 C5 5A

<< Texte déchiffré

00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff.

On constate que le résultat de déchiffrement correspond bien au texte clair.

❖ 3eme **Exemple :**

Pour une chaine de caractères on a :

<<Texte clair :

« La cryptographie est la branche de la cryptologie qui conçoit les procédés empêchant la divulgation d'une information à un individu qui n'est pas autorisé à en prendre connaissance. En opposition, on trouve la cryptanalyse dont l'objectif est d'acquérir une information par des moyens non légitimes. La cryptographie cherche à garantir la sécurité de ces informations ».

<<Texte chiffré :

«]Ù 3/š ÷È à †;¶Ìv_ç ,-7»Q*@gTö«:¾^,êJWS™§0æ°P <õ)w½r© =ãÉ ã% ¿Å µÔ¬ßæŠ$x öZÙ“ÙM}ƒF Û •jƒ€úƒi Œ| ìt<~ü° ÛÔ4aòä{+™µéêùzŸ½ i¢‡ BZg : €4 i¶h*©ÇååxQÐû .JžŠbYÝŽ,Ã Ô[’Ãl%õvcTÒK½•Â†DÁÎ”2Ü4œŽ‘.<mFßFRtƒwŒ‰/ Üù* ûÇœÑŠ(h On'0?+âÌ=a˜² È!ô >¨ o[?4øÂ¸Ç°P,Z •¦ùª Ž¹Ù±Õn\Ñ0â#0-ÂÓ2KOYÆ ;«÷ÑœÛ! $} ò9ŠÚ"Ì|ÿ`¾÷~ i´;ÖShIqÐs :ÞH(<´ÃkíÞt €ìaumz¶úk„• ˆê§ÅrhBÔÃž.B ÓxO„Ye µž ».

<< Texte déchiffré :

« La cryptographie est la branche de la cryptologie qui conçoit les procédés empêchant la divulgation d'une information à un individu qui n'est pas autorisé à en prendre connaissance.

En opposition, on trouve la cryptanalyse dont l'objectif est d'acquérir une information par des moyens non légitimes. La cryptographie cherche à garantir la sécurité de ces informations ».

Le texte obtenu au déchiffrement correspond au texte clair chiffré, ce qui implique le bon fonctionnement de l'algorithme.

IV.2 Caractéristiques et points forts de l'AES:

L'AES est un algorithme de cryptage qui répond très bien aux besoins en terme de sécurité et aux contraintes (temps réel, taille des données et ressources limitées). En effet, l'AES offre un niveau de sécurité suffisant pour tout type d'application commerciale ; il est caractérisé par sa facilité de calcul et sa rapidité de traitement, ses besoins faibles en ressources et en mémoire, sa flexibilité d'implémentation, sa simplicité et la possibilité de son implémentation aussi bien sous forme logicielle que matérielle.

- **Sécurité :**

L'AES n'a pas d'attaques de sécurité connues. Il utilise des boîtes S_Box comme composants non-linéaires. L'AES apparaît avoir une marge de sécurité suffisante. En effet, il résiste aux attaques par cryptanalyse linéaire, aux attaques par cryptanalyse différentielle et aux attaques par dictionnaires. Il a aussi démontré de bons résultats pour résister aux timing attack et power attack sur les environnements ayant peu de mémoires comme les cartes à puce. Ces attaques permettent d'extraire d'un tel composant la clé secrète en effectuant des mesures de temps et de puissances dégagés par la puce.

- **Flexibilité :**

L'AES a été conçu pour être flexible en terme de taille des clés et du bloc de chiffrement. En effet, l'AES peut être modifié en utilisant différents nombres de tours bien que ces caractéristiques n'aient pas encore été étudiées quant aux conséquences sur la sécurité de l'algorithme.

L'AES spécifie trois tailles de clé: 128, 192 où 256 bits. Ceci représente respectivement environ $3{,}4 \times 10^{38}$ clés de 128 bits possibles ; $6{,}2 \times 10^{57}$ clés de 192 bits et

$1{,}1 \times 10^{77}$ clés de 256 bits. Ce qui rend impossible la construction de machines comme le « DES Cracker » qui effectuait une recherche exhaustive sur une clé DES de 56 bits ($7{,}2 \times 10^{16}$ clés). Si on fait l'hypothèse qu'on peut construire une machine permettant de trouver une clé DES en une seconde (qui essaie 255 clés par seconde), alors il lui faudrait 149 mille milliards d'années pour casser une clé de 128 bits.

- **Besoins en ressources et mémoire très faibles :**

L'AES s'insère très bien dans les systèmes embarqués. En effet, il ne demande que peu de mémoire c'est pourquoi il est utilisé avec les cartes à puce.

CONCLUSION GENERALE

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.

Dans notre travail on a mis l'accent sur l'importance de la cryptographie dans la sécurité des réseaux et l'apport des algorithmes de cryptage en particulier l'AES qui résiste bien aux différentes attaques des cryptanalistes jusqu'à maintenant.

L'objectif principal assigné à notre travail est l'implémentation de l'algorithme AES sous matlab qui permet de chiffrer et déchiffrer des messages avec une clé secrète qui répond bien à nos ambitions.

Ce travail nous a permis d'améliorer nos connaissances dans le domaine de la sécurité des réseaux notamment la cryptographie ainsi que la programmation.

Le domaine de la securité des réseaux reste ouvert aux développements avec l'évolution des réseaux et les systèmes d'information et aux contraintes liées à leurs faiblesses.

BIBLIOGRAPHIE

[1] *HASSINA CHAOUCHE et DJAMILA KADOUCHE* : *«sécurité des réseaux» thèse d'ingéniorat, UMMTO 2003.*

[2] *BELAL SMAIL et TOUBAL RAFIK* : *«sécurité des réseaux de troisième génération algorithme Kasumi» thèse d'ingéniorat UMMTO 2006.*

[3] *BOUTIOUTA ABOUBAKR* : *«sécurité et géstion de la mobilité dans le réseau GSM» thèse d'ingéniorat Institue de télécommunication d'ORAN 2006.*

[4] *Simon Guillem-Lessard «Tutoriel de la cryptographie», Projet de fin d'étude 2001-2002 Département des mathématiques et de l'informatique Université du Québec à Trois-Rivières.*

[5] *HAJER DJELOUL* : *«développement et intégration d'un module logiciel embarque pour la sécurisation de la transmission des données de suivi par GPS» thèse d'ingéniorat en télécommunication Ecole supérieure des Télécommunication de Tunis 2005.*

[6] *Guy Pujolle: «les réseaux édition 1998» édition DUNOD.*

[7] *«sécurité informatique et réseaux» édition DUNOD 2006.*

[8] *«cryptographie et sécurité des systèmes et réseaux», édition 2006 sous la direction de TOURADJ EBRAHIMI, FRANCK LEPREVOST et BERTRAND WARUSFEL.*

[9] *«systèmes cryptographiques», ABDALLAH M'HAMED. Département réseaux et services, Institut National des télécommunications.*

[10] *National Institute of Standards and Technology: Specification for the Advance Encryption Standard (AES). http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf, (2001).*

[11] *National Institute of Standards and Technology: Data Encryption Standard (DES). http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, (2001).*

[12] *http://www.urec.cnrs.fr/IMG/GER/-060124-les algorithmes de cryptographie.pdf.*

[13] *http://www.pgp.org/introduction à la cryptographie/chapitre2.pdf.*

[14] *http://www.uqtr.ca/DMI/.*

[15] *http://fr.wikipedia.org/.*

[16] *http://www.commentcamarche.net.*

[17] *http://www.securiteinfo.com.*

- ***Confusion, diffusion :***

Les systèmes de chiffrement sont des systèmes de chiffrements par blocs, c'est à dire que l'on découpe d'abord le texte clair en blocs de même taille, et le chiffrement est obtenu par l'application d'une fonction de chiffrement E_K à chacun des blocs. Dans le monde moderne, les messages sont des suites de bits (par exemple obtenus à partir de l'alphabet courant par le code ASCII). Si la taille des blocs est ℓ , supposons pour simplifier que la fonction de chiffrement E_K définisse une bijection $E_K : F_2^\ell \to F_2^\ell$.

Le premier exemple montre que, si ℓ est trop petit, la répartition des blocs de taille ℓ dans les messages clairs ne suivant pas une loi uniforme, le cryptosystéme est sensible à une analyse statistique. Si ℓ devient grand, et si E_K est vraiment quelconque, le problème est de calculer E_K (m). Si l'on doit passer par le stockage en mémoire des valeurs prises par E_K (m) cela limite la taille de ℓ (justement, on peut prendre comme définition de la notion de permutation quelconque la propriété suivant laquelle il n'y a pas de méthode plus simple par exemple en terme de longueur de programme pour la définir que de donner toutes ses images). Si à l'opposé on choisit une fonction extrêmement facile à stocker et à calculer, on tombe sur un système du type linéaire comme dans le deuxième exemple.

Très schématiquement on dit qu'un système avec ℓ petit et E_K quelconque apporte de la confusion tandis qu'un système linéaire où ℓ est choisit grand apporte de la diffusion (i.e. un bit donné du message clair influence un grand nombre de bits du message chiffré). Les systèmes par blocs modernes, alternent les séquences de confusion et de diffusion.

- ***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* la reste.
- Remplacer *a* par *b* et *b* par *r* (on a donc $0 \leq r < b$ 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
$0 \leq r > b$. on appelle q le quotient de a par b et r le reste de la division de a par b.

- ***Théorème d'Euclide étendu:***

Il existe une version étendu de l'algorithme Euclide appelé « Euclide étendu ». Qui permet de calculer les coefficients de Bézout.
Soit $a, b \in Z$ dont au moins un est nul et *d=pgcd (a, b).* Alors il existe un couple d'entiers
(u , v) tel que :
$a * u = b * v = d$
Les entiers u et v sont appelés coefficient de Bézout.

- ***Fonction d'indicatrice d'Euler :***

Soit a et n deux entiers avec $n \geq 2$.les affirmations suivantes sont équivalentes :
a est un générateur du groupe additif $(Z / nZ, +)$.
a est un élément inversible de l'anneau Z / nZ .
Pgcd (a, n) =1.
On note $(Z / nZ)^{\times}$ l'ensemble des éléments inversible de Z / nZ et pour $\varphi(n)$
Son cardinal. La fonction φ est appelé fonction indicatrice d'Euler. Elle put être calculée par la relation :

$$\varphi(n) = n \prod_{ppremier} (1 - 1 / p)$$

Ainsi $(Z/nZ)^{\times}$ est un groupe multiplicatif de cardinale $\varphi(n)$. Rappelons que tout élément g d'un groupe fini (G, X) , on a $g^{card(G)} = 1$.On obtient ainsi la proposition suivante.

Soit a et n deux entiers tel que n ≥ 2et pgcd de (a, n) = 1.

Alors :

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

Ce qui s'écrit $\bar{a}^{\varphi(n)} = \bar{1}$ dans Z/nZ .

- ***Notion de nombre premier :***

Soit *p* un entier positif, $p \geq 2$. Ont dit que p est premier si ses seul 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.*

- ***Préliminaires mathématiques sur le corps F_{256} ou GF (2^8) :***

- ***Rappels: construction d'un corps fini à P^m éléments :***

On considère d'abord le corps fini à *P* éléments F_P (ou *P* est premier).

Comme P est premier, $F_P = Z/PZ$. Soit m un entier positif. On souhaite construire le corps F_{P^m} possédant P^m éléments. Pour cela, on considère ***g*** un polynôme unitaire irréductible de F_P de degré m. Alors on peut montrer que F_{P^m} est isomorphe à $F_P[X]/g(X)$. Autrement dit, tout élément de F_{P^m} peut être vu comme un polynôme de $F_P[x]$ (i.e. dont les coefficients sont dans $F_P = Z/PZ$ de degré inférieur à m:

$$\mathbb{F}_{p^m} \simeq \left\{ \sum_{i=0}^{m-1} a_i X^i, \quad a_i \in \mathbb{F}_p \right\} = \left\{ \sum_{i \geq 0} a_i X^i \mod g(X), \quad a_i \in \mathbb{F}_p \right\}$$

- ***Applications à la construction de*** F_{256} ***:***

Dans ce cas, *P*=2 et *m=8*. Il convient donc de choisir un polynôme unitaire irréductible de degré 8. Plusieurs choix sont possibles.

Ainsi, les encodeurs/décodeurs CIRC utilisés pour les CD audio utilisent le polynôme $g(X) = X^8 + X^4 + X^3 + X + 1$.

Pour AES, le polynôme utilisé est : $g(X) = X^8 + X^4 + X^3 + X + 1$.

On aura donc :

$$\mathbb{F}_{256} \simeq \left\{ \sum_{i=0}^{7} a_i X^i, \quad a_i \in \{0,1\} \right\} = \left\{ \sum_{i \geq 0} a_i X^i \mod g(X), \quad a_i \in \{0,1\} \right\}$$

En pratique, on représentera le polynôme $a_7 X^7 + ... + a_1 X + 1$ par l'octet $a_7 a_6 ... a_1 a_0$.

On dira qu'on utilise une notation polynomiale des éléments de F_{256}.

- ***Addition dans*** F_{256} ***:***

L'addition de deux éléments de $F_P = Z/2Z$ correspond à l'opération XOR. Ainsi, l'addition de deux éléments a et b de F_{256} correspond à l'addition de deux polynômes à coefficients dans $F_P = Z/2Z$:

$$a(x) = a_7 x^7 + \ldots + a_1 x + a_0$$

$$b(x) = b_7 x^7 + \ldots + b_1 x + a_0$$

$$\Rightarrow a(x) = b(x) = (a_7 \oplus b_7) x^7 + ... + (a_1 \oplus b_1) x + (a_0 \oplus b_0)$$

Exemple :

$(x^6 + x^4 + x^2 + x + 1) \oplus (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$ (notation polynomiale).

$\{01010111\} \oplus \{10000011\} = \{110101\}$ (Notation binaire).

$\{57\} \oplus \{82\} = \{da\}$ (Natation hexadécimale).

En notation hexadécimale, le polynôme g(x) s'écrit : `0x11B`.

- ***Multiplication dans*** F_{256} ***:***

En notation polynômiale, la multiplication dans F_{256} correspond à la multiplication de deux polynômes suivie d'une réduction modulo *m(X)*.

Exemple:

$$(x^6 + x^4 + x^2 + x + 1) \oplus (x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^7 +$$

$$x^7 + x^5 + x^3 + x^2 + x +$$

$$x^6 + x^4 + x^2 + x + 1$$

$$= x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1$$

Et :

$$(x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1) \bmod (x^8 + x^4 + x^3 + x + 1) = x^7 + x^6 + 1$$

La réduction modulo *m(x)* assure que le résultat reste un polynôme binaire de degré inférieur à 8 qui peut donc être représenté par un octet. Dans l'exemple précédent, on a obtenu: $\{57\} \times \{83\} = \{c1\}$

En pratique, pour réaliser cette multiplication, on dispose de deux méthodes, expliquées dans les sections suivantes.

- ***Méthode "lente" : multiplication par x^i :***

Cette méthode consiste à réaliser effectivement la multiplication polynomiale. Soit $a \in F_{256}$. En notation polynomiale, $a(x) = a_7 x^7 + ... + a_1 x + a_0$

Ainsi, $xa(x) = a_7 x^8 + ... + a_1 x^2 + a_0 x$. Il reste à effectuer la réduction modulo *m(x)*. Deux cas sont possibles:

- Soit $a_7 = 0$, on obtient directement une expression réduite est donc :

$$x.a(x) = a_6 x^7 + \ldots + a_1 x^2 + a_0 x$$

.

- Soit $a_7 = 1$ et d: $xa(x) = a_7 x^8 + ... + a_1 x^2 + a_0 x$ En outre $g(X) = X^8 + X^4 + X^3 + X + 1 = 0 \bmod(mx) \Rightarrow x^8 = x^4 + x^3 + x + 1 \bmod m(x)$

  Donc :

$$x^8 + a_6 x^7 + \ldots + a_1 x^2 + a_0 x = (a_6 x^7 + \ldots + a_1 x^2 + a_0 x) \oplus (x^4 + x^3 + x + 1)$$

En notation polynomiale, cette opération consiste donc à un décalage à gauche suivi éventuellement d'un XOR bits-à-bits avec {1B}.

Soit x_time la fonction qui réalise l'opération $x \times a(x) \bmod m(x)$. x_time peut être utilisée pour définir la fonction x_time qui réalise l'opération $x^i \times a(x) \bmod m(x)$. Enfin, à partir de cette dernière fonction, il est possible de réaliser la multiplication complète de deux éléments de F_{256} .

- ***Méthode "rapide" à partir d'une représentation exponentielle :***

Soit $w(x)$ un générateur de $F_P[X]/m(X)$: dans ce cas, tout élément non nul de F_{256} se représente de manière unique par $w(x^i) \times a(x) \bmod m(x)$,0≤ i < 255.

On obtient ainsi une autre écriture pour F_{256} :

$$\mathbf{F}_{256} \simeq \{0\} \cup \{w(x)^i \mod g(x)\}_{0 \le i < 255}$$

Avec cette représentation (dite cyclique ou exponentielle), la multiplication de deux éléments a et b non nuls est aisée.

$$a(x) = w(x^i)$$

$$b(x) = w(x^j)$$

$$a(x) \times b(x) = w(x)^{i+j \bmod 255}$$

L'idée consiste alors a passer par cette représentation pour effectuer la multiplication. On utilise pour cela des tables de correspondance entre les éléments a(x) de F_{256} et l'exposant i correspondant à sa représentation exponentielle.

A noter que la valeur de *w(x)* dépend de *m(x)*.

- ***Les polynômes à coefficients dans*** F_{256} ***:***

Tout comme un octet peut être représenté par un polynôme de degré 7, un mot de 32 bits (4 octets) peut l'être par un polynôme de degré 3 à coefficients dans $\frac{Z_2(x)}{m(x)Z_2(x)}$, chaque coefficient représentant un octet du mot.

$$
\begin{array}{ll}
 & \texttt{0x5b 0x99 0xb3 0xe7} \\
\Leftrightarrow & 01011011_2\ 10011001_2\ 10110011_2\ 11100111_2 \\
\Rightarrow & (01011011_2)X^3 + (10011001_2)X^2 + (10110011_2)X + (11100111_2) \\
\Leftrightarrow & a_1X^3 + a_2X^2 + a_1X + a_0 = a(X)
\end{array}
$$

L'addition de deux mots est alors égale à l'addition des polynômes les représentants, soit un ou-exclusif entre les coefficients de même degré.

$$a(X) + b(X) = (a_3 \oplus b_3)X^3 + (a_2 \oplus b_2)X^2 + (a_1 \oplus b_1)X + (a_0 \oplus b_0)$$

La multiplication de deux mots donne un polynôme de degré 3+3 dont on peut calculer les coefficients par la définition précédente et la définition générale du produit de polynômes.

$$a(X) \times b(X) = c(X) = c_6X^6 + c_5X^5 + c_4X^4 + c_3X^3 + c_2X^2 + c_1X + c_0$$

$$
\begin{array}{lcl}
c_0 & = & a_0 \otimes b_0 \\
c_1 & = & a_1 \otimes b_0 \oplus a_0 \otimes b_1 \\
c_2 & = & a_2 \otimes b_0 \oplus a_1 \otimes b_1 \oplus a_0 \otimes b_2 \\
c_3 & = & a_3 \otimes b_0 \oplus a_2 \otimes b_1 \oplus a_1 \otimes b_2 \oplus a_0 \otimes b_3 \\
c_4 & = & a_3 \otimes b_1 \oplus a_2 \otimes b_2 \oplus a_1 \otimes b_3 \\
c_5 & = & a_3 \otimes b_2 \oplus a_2 \otimes b_3 \\
c_6 & = & a_3 \otimes b_3
\end{array}
$$

Afin de rester dans l'espace des mots de 32 bits, on considère les polynômes *C(x)* $modulo$ un polynôme de degré 4 qui vaut pour l'AES $X^4 + 1$, formant un groupe que l'on pourrait écrire :

$$\mathcal{A} = \frac{GF(2^8)[X]}{(X^4 + 1)GF(2^8)[X]}$$

On obtient par calcul les coefficients d_i :

$$
\begin{array}{lcl}
d_0 & = & a_0 \otimes b_0 \oplus a_3 \otimes b_1 \oplus a_2 \otimes b_2 \oplus a_1 \otimes b_3 \\
d_1 & = & a_1 \otimes b_0 \oplus a_0 \otimes b_1 \oplus a_3 \otimes b_2 \oplus a_2 \otimes b_3 \\
d_2 & = & a_2 \otimes b_0 \oplus a_1 \otimes b_1 \oplus a_0 \otimes b_2 \oplus a_3 \otimes b_3 \\
d_3 & = & a_3 \otimes b_0 \oplus a_2 \otimes b_1 \oplus a_1 \otimes b_2 \oplus a_0 \otimes b_3
\end{array}
$$

Ces opérations peuvent être mises sous forme matricielle :

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Comme X^4+1n'est pas un polynôme irréductible, $\mathcal{A}$ n'est pas un corps et ses éléments ne sont pas forcément inversibles, l'AES utilise alors pour ses calculs sur les mots le polynôme *a(x)* inversible dans $\mathcal{A}$.

$$\begin{aligned} a(X) &= (0\text{x}03)X^3 + (0\text{x}01)X^2 + (0\text{x}02)X + (0\text{x}02) \\ a^{-1}(X) &= (0\text{x}0\text{b})X^3 + (0\text{x}0\text{d})X^2 + (0\text{x}09)X + (0\text{x}0\text{e}) \end{aligned}$$

- ***Détail de la boite S_Box et son inverse :***

		Y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**E**	**f**
	0	**63**	**7c**	**77**	**7b**	**f2**	**6b**	**6f**	**c5**	**30**	**01**	**67**	**2b**	**fe**	**d7**	**Ab**	**76**
	1	**ca**	**82**	**c9**	**7d**	**fa**	**59**	**47**	**f0**	**Ad**	**d4**	**a2**	**af**	**9c**	**a4**	**72**	**c0**
	2	**b7**	**fd**	**93**	**26**	**36**	**3f**	**f7**	**cc**	**34**	**a5**	**e5**	**f1**	**71**	**d8**	**31**	**15**
	3	**04**	**c7**	**23**	**c3**	**18**	**96**	**05**	**9a**	**07**	**12**	**80**	**e2**	**eb**	**27**	**b2**	**75**
	4	**09**	**83**	**2c**	**1a**	**1b**	**6e**	**5a**	**a0**	**52**	**3b**	**d6**	**b3**	**29**	**e3**	**2f**	**84**
	5	**53**	**d1**	**00**	**ed**	**20**	**fc**	**b1**	**5b**	**6a**	**cb**	**be**	**39**	**4a**	**4c**	**58**	**cf**
	6	**d0**	**ef**	**aa**	**fb**	**43**	**4d**	**33**	**85**	**45**	**f9**	**02**	**7f**	**50**	**3c**	**9f**	**a8**

x																
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

La table S_Box.

		Y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	E	f
	0	52	09	6a	d5	30	36	a5	38	Bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	Ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5^{e}	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73

x	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	Df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	Be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	Ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

La table Inv_S_Box.

Poly_mat =

$$\begin{matrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{matrix}$$

inv_poly_mat =

$$\begin{matrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{matrix}$$

- ***Expansion de la clé:***

Cette section contient l'expansion de la clé à partir de la clé de chiffrement initiale**.**

Clé de chiffrement = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c.

Pour Nk=4:

$w0 = \mathbf{2b7e1516}$ $w1 = \mathbf{28aed2a6}$ $w2 = \mathbf{abf71588}$ $w3 = \mathbf{09cf4f3c}$

i (dec)	temp	After RotWord()	After SubWord()	Rcon[i/Nk]	After XOR with Rcon	W[i-Nk]	w[i]= temp XOR w[i-Nk]
4	09cf4f3c	cf4f3c09	8a84eb01	01000000	8b84eb01	2b7e1516	a0fafe17
5	a0fafe17					28aed2a6	88542cb1
6	88542cb1					abf71588	23a33939
7	23a33939					09cf4f3c	2a6c7605
8	2a6c7605	6c76052a	50386be5	02000000	52386be5	a0fafe17	f2c295f2
9	f2c295f2					88542cb1	7a96b943
10	7a96b943					23a33939	5935807a
11	5935807a					2a6c7605	7359f67f
12	7359f67f	59f67f73	cb42d28f	04000000	cf42d28f	f2c295f2	3d80477d
13	3d80477d					7a96b943	4716fe3e
14	4716fe3e					5935807a	1e237e44
15	1e237e44					7359f67f	6d7a883b
16	6d7a883b	7a883b6d	dac4e23c	08000000	d2c4e23c	3d80477d	ef44a541
17	ef44a541					4716fe3e	a8525b7f
18	a8525b7f					1e237e44	b671253b
19	b671253b					6d7a883b	db0bad00
20	db0bad00	0bad00db	2b9563b9	10000000	3b9563b9	ef44a541	d4d1c6f8
21	d4d1c6f8					a8525b7f	7c839d87
22	7c839d87					b671253b	caf2b8bc
23	caf2b8bc					db0bad00	11f915bc
24	11f915bc	f915bc11	99596582	20000000	b9596582	d4d1c6f8	6d88a37a
25	6d88a37a					7c839d87	110b3efd
26	110b3efd					caf2b8bc	dbf98641
27	dbf98641					11f915bc	ca0093fd
28	ca0093fd	0093fdca	63dc5474	40000000	23dc5474	6d88a37a	4e54f70e
29	4e54f70e					110b3efd	5f5fc9f3
30	5f5fc9f3					dbf98641	84a64fb2
31	84a64fb2					ca0093fd	4ea6dc4f
32	4ea6dc4f	a6dc4f4e	2486842f	80000000	a486842f	4e54f70e	ead27321
33	ead27321					5f5fc9f3	b58dbad2
34	b58dbad2					84a64fb2	312bf560
35	312bf560					4ea6dc4f	7f8d292f
36	7f8d292f	8d292f7f	5da515d2	1b000000	46a515d2	ead27321	ac7766f3

37	ac7766f3					b58dbad2	19fadc21
38	19fadc21					312bf560	28d12941
39	28d12941					7f8d292f	575c006e
40	575c006e	5c006e57	4a639f5b	36000000	7c639f5b	ac7766f3	d014f9a8
41	d014f9a8					19fadc21	c9ee2589
42	c9ee2589					28d12941	e13f0cc8
43	e13f0cc8					575c006e	b6630ca6

- ***AES*** (Advanced Encryption Standard) [Standard de chiffrement avancé] Standard approuvé par le NIST, en général valable pour les 20 ou 30 prochaines années.

- ***Algorithm (encryption)*** [Algorithme de déchiffrement]
  Ensemble de règles mathématiques (logiques) utilisées pour le chiffrement et le déchiffrement.

- ***Algorithm (hash)*** *[*Algorithme de hachage]
  Ensemble de règles mathématiques (logiques) utilisées dans le processus de création d'empreinte de message et la génération de clés / de signatures.

- ***Asymetric Key*** [Clés asymétriques]
  Paire de clés séparées mais unifiées, composée d'une clé publique et d'une clé privée. Chaque clé est à sens unique, ce qui signifie que la clé utilisée pour chiffrer des informations ne peut pas être utilisée pour déchiffrer les mêmes données.

- ***Block cipher*** [Chiffre par blocs]
  Chiffre symétrique opérant sur des blocs de texte clair et de texte chiffré, habituellement de 64 bits.

- ***Bluetooth*** **:** C'est un protocole de communication sans-fil courtes distances qui a été largement employée depuis plusieurs années dans de nombreux types de périphériques (téléphones mobiles, ordinateurs portables, GPS, routeurs, imprimantes, appareils photos, etc.). Il fonctionne dans la gamme des 2.4GHz tout comme son homologue 802.11. Il existe 79 canaux Bluetooth, d'une largeur de bande d'1MHz chacun.

- ***Cipher text :*** [Texte chiffré (ou cryptogramme)]
  Résultat de la manipulation de caractères ou de bits via des substitutions, transpositions, ou les deux.

- ***Cryptanalyse* :** La cryptanalyse comprend l'ensemble des moyens qui permet d'analyser une information préalablement chiffrée, afin de la déchiffrer. Plus un système de chiffrement est robuste, plus sa cryptanalyse est difficile.

- ***FIPS*** (Federal Information Processing Standard) [Standard de traitement de données fédéral] Standard gouvernemental américain publié par le NIST.

- ***GNUPGP :*** (Gnu Privacy Guard) une implémentation du standard Open PGP appelée GPG libre de droit.

- ***3GPP:*** Third Génération Parthershipe Project.

- ***Initialization vector (IV)*** [Vecteur d'initialisation].
  Bloc de données arbitraire qui sert de point de départ pour les chiffres par blocs qui utilisent un mécanisme de chaînage ou de rétroaction.

- ***IPsec*** Couche TCP/ IP de chiffrement en cours d'examen par l'IETF.

- ***ISO*** (International Organization for Standardization) [Organisation de standardisation internationale]. Responsable d'une large gamme de standards, comme le modèle OSI et les relations internationales avec l'ANSI sur le X. 509.

- ***MISTY* :** Proposés en 1996 par Mitsuru Matsui, MISTY1 et MISTY2 sont des algorithmes de chiffrement symétriques itératifs par blocs. La taille des blocs est de 64 bits et la taille de la clé de 128 bits. Le nombre de tours, variable et multiple de 4, est au minimum de 8.

- ***NIST*** (National Institute for Standards and Technology) [institut national pour les standards et la technologie] .Division du département américain du commerce qui publie des standards ouverts d'interopérabilité appelés FIPS.

- ***PKI*** (Public Key Infrastructure) [Infrastructure à clé publique]
  Système de certificats largement disponible et accessible pour obtenir la clé publique d'une entité, avec une bonne probabilité que vous ayez la "bonne" clé et qu'elle n'ait pas été révoquée.

- ***Plaintext*** (or clear text) [Texte clair (ou libellé)]
  Données ou message lisible par un humain avant chiffrement.

- ***Session key Protocole* :** Un protocole est un Ensemble des règles qui régissent les échanges entre émetteur et récepteur, il est met en œuvre par un ensemble de règles matériel ou logiciel appeler procédure de transmission.

- ***[Clé de session]***
  Clé secrète (symétrique) utilisée pour chiffrer chaque jeu de données dans un système de transaction. Une clé de session différente est utilisée pour chaque session de communication.

- ***SSH*** (Secure Shell) [Shell sûr]
  Protocole proposé par l'IETF pour sécuriser la couche transport en fournissant chiffrement, authentification cryptographique de l'hôte et protection de l'intégrité.

- ***SSL*** (Secure Socket Layer) [Couche de sockets sûres]
  Développé par Netscape pour fournir sécurité et confidentialité sur Internet. Supporte l'authentification du serveur et du client et assure la sécurité et l'intégrité du canal de transmission. Opère au niveau de la couche de transport et imite la "bibliothèque des sockets", permettant d'être indépendant de l'application. Chiffre complètement le canal de communication et ne supporte pas les signatures numériques au niveau du message.

- ***XOR*** [Ou exclusif]
  Opération ou exclusif; une façon mathématique de représenter des différences. X. 509v3 Certificat numérique ITU- T qui est un document électronique reconnu à l'échelle internationale pour prouver l'identité et la propriété d'une clé publique sur un

réseau. Il contient le nom de celui qui l'émet, son information d'identification, sa signature numérique, ainsi que d'autres extensions possibles en version 3.

- ***Wifi*** La norme WiFi (Wireless Fidelity) est le nom commercial donné à la norme IEEE (Institute of Electrical and Electronics Engineers – l'organisme de certifications des normes réseaux) 802.11b et 802.11g par la WiFi Alliance. Ce standard est actuellement l'un des standards les plus utilisés au monde. Les débits théoriques du 802.11b sont de 11 Mb/s et 54 Mb/s pour le 802.11g.

- ***WEP*** (Wired Equivalent Privacy) est un protocole de sécurisation des réseaux WIFI. Il utilise un système symétrique dans lequel la même clé et le même algorithme sont utilisés pour le chiffrement et le déchiffrement des données.

Listes des figures