N^0 d'ordre : 01 /2004 - M /MT

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie
Houari Boumediene

Faculté des Sciences Mathématiques

Mémoire

Présentée pour l'obtention du Grade de

Magister en Mathématiques

Spécialité : Algèbre et Théorie des Nombres

Par

M elle : HAMZAOUI SAIDA

Thème

Techniques de Cryptographie

Soutenue publiquement le: 05 / 01 / 2004. Devant le Jury composé de:

Mr- M. ZITOUNI, Professeur, USTHB.	Président
Mr- K. BETINA, Professeur, USTHB.	Directeur de thèse
Mr- M.S. HACHAICHI, Maître de conférences, USTHB.	Examinateur
Mr- M. ABID, Maître de conférences, USTHB.	Examinateur

Dédicace

Je dédie ce modeste travail

A ma très chère mère qui a tant sacrifiée pour que je réussisse mes études, à mon unique frère Abdelkrim et à mon oncle Mohamed qui m'ont beaucoup encouragé dans ce parcours.

A mon grand père et ma grande mère.

A mes oncles, mes tantes et mes cousins.

A mes trois petit chatons Abdelmalek, Bilel et Amel.

Remerciements

Je tiens à remercier le bon Dieu, le tout puissant, de m'avoir permis de réussir mes études.

J'exprime mes vifs remerciements à Monsieur K.Betina. , mon promoteur de m'avoir guidé et conseillé dans mon travail.

Je remercie Monsieur M.Zitouni, d'avoir accepter de présider le jury.

Mes remerciements vont également à Messieurs M.S.Hachaichi et M.Abid de l'honneur qu'ils m'ont fait faire partie de mon jury.

Mes remerciements vont également à mes enseignants et tous ceux qui m'ont aidé de prés ou de loin, qu'ils trouvent ici ma profonde gratitude

Sommaire

Introduction générale

Le mot cryptographie vient du grec kruptos (caché) et graphein (écrire). Réservé autre fois à l'usage diplomatique et militaire. La cryptographie est aujourd'hui en plein développement. Elle entre de plein pied dans le développement économique des pays.

Le développement des réseaux de communication informatique permet des « contacts » plus faciles et peu chers du point de vu « sécurité » (afin d'éviter tout piratage). La télécommunication remplace le courrier et les voyages, ce qui entraîne un gain de temps et d'argent.

La cryptographie fait appel à plusieurs branches de mathématiques et en particulier à l'algèbre et théorie des nombres.

On essaie dans ce modeste travail d'étudier quelques techniques de cryptographie basées sur l'algèbre et théorie des nombres (les nombres premiers, logarithme discret et courbes elliptiques) pour protéger les communications à travers le réseau Internet ou d'autres canaux de transmissions.

Dans le chapitre 1 on introduit la cryptographie classique.

Le chapitre 2 est consacré à la cryptographie à clé secrète.

Dans le chapitre 3 on explique la cryptographie à clé publique.

Et enfin pour le dernier chapitre on étudie les procédés de signature.

1.1 Introduction

Avant l'avènement des ordinateurs, la cryptographie traitait des systèmes cryptographiques basés sur les lettres (ou caractères). Les différents algorithmes cryptographiques remplaçaient des caractères par d'autres ou transposaient les caractères. Les meilleurs systèmes faisaient les deux opérations plusieurs fois.

Définition 1-1

La cryptographie est l'étude des principes, méthodes et techniques mathématiques reliés aux aspects de la sécurité de l'information telles la confidentialité, l'intégrité des données, l'authentification d'entité, et l'authentification de l'originalité des données. C'est un ensemble de technique qui fournit la sécurité de l'information.

La cryptographie nous permet de stocker des informations sensibles ou de les transmettre à travers des réseaux non sûrs (comme Internet) de telle sorte qu'elles ne peuvent être lues par personne à l'exception du destinataire convenu.

Définition 1-2

Un cryptosystème est constitué d'un algorithme cryptographique ainsi que tous les clés possibles et tous les protocoles qui le font fonctionner.

Définition 1-.3

Le cryptographe est la personne qui utilise la cryptographie pour fournir des outils pour éliminer les risques, afin de rendre les échanges d'information confidentiels, infalsifiables, authentiques et inaltérables.

L'information est chaque jour échange d'un point à un autre et se trouve susceptible d'être lue, copiée, supprimée, altérée ou falsifiée.

Définition 1-4

Le message clair est le message d'origine.

Définition 1-5

Le chiffrement est la transformation effectuée sur le texte clair.

Définition 1-6

Le texte chiffré ou cryptogramme est le message transformé par un algorithme de chiffrement.

Définition 1-7

Le déchiffrement est la transformation de reconstitution sur un texte.

Définition 1-.8

Un algorithme cryptographique, est une procédure utilisée pour le chiffrement et le déchiffrement.

Définition 1-9

Le niveau de sécurité est la quantité de travail maximal pour chiffrer un message.

Définition 1-10

Les fonctionnalités sont déterminées par les propriétés de base des outils cryptographiques.

Définition 1-11

La performance consiste en l'efficacité de la puissance de calcul par rapport au temps, par exemple compter le nombre de bits chiffrés par seconde.

Définition 1-12

La facilité d'implémentation est définie par la complexité selon l'environnement (logiciel et matériel).

1.2 Objectifs de la cryptographie

Les principaux objectifs de la cryptographie sont les suivants :

1.2.1 La confidentialité

Assurer que les données concernées ne sont connues que par les personnes autorisées.

1.2.2 L'intégrité

Assurer la non altération des données pendant leur transmission.

1.2.3 L'authentification

Prouver l'identité d'une personne ou l'origine d'une donnée.

1.2.4 La signature

Elle garantit par un contrat le suivit des engagements.

1.3 Communication à l'aide de la cryptographie

L'émetteur veut envoyer un message à son destinataire, mais il ne veut pas que quelqu'un d'autre prenne connaissance de ce message. Pour cela l'émetteur utilise un système cryptographique qui cache le sens du message. Seul le destinataire du message sera capable de retrouver le contenu du message, grâce à une information supplémentaire (la clé) qu'il aura convenu à l'avance avec l'émetteur.

L'information que l'émetteur veut transmettre à son destinataire, que l'on appelle texte clair, peut prendre de nombreuses formes : un texte, une donnée numérique, etc. L'émetteur transforme le texte clair par un procédé de chiffrement en utilisant la clé, et envoie le texte chiffré au destinataire qui déchiffre ce texte pour retrouver le texte clair.

Définition 1-14

Un système cryptographique est un quintuplé ($\wp$, *C, K, E, D)*

1. $\wp$ est un ensemble fini de blocs de textes clairs possibles.
2. *C* est un ensemble fini de blocs de textes chiffrés possibles.
3. *K* est un ensemble fini de clés possibles.
4. *E* est l'ensemble des règles de chiffrements possibles.
5. *D* est l'ensemble des règles de déchiffrements possibles.

Satisfaisant aux conditions suivantes :

Pour tout $k \in K$, il y a une règle de chiffrement $e_k \in E$ et une règle de déchiffrement correspondante $d_k \in D$.

$e_k : \wp \to C$ et $d_k : C \to \wp$ sont des fonctions telles que $d_k(e_k(x)) = x$ pour tout texte clair $x \in \wp$.

1.4 La cryptanalyse

C'est l'étude des systèmes cryptographiques, en particulier de leurs faiblesses, dans le but de déchiffrer les messages dont on n'est pas destinataire.

On fait généralement l'hypothèse du principe de Kerckhoff, l'attaquant connaît le système cryptographique utilisé.

Dans la pratique, le principe de Kerckhoff n'est pas toujours vérifié.On souhaite baser la sécurité de la transmission uniquement sur la protection de la clé.

La cryptanalyse d'un système cryptographique peut-être :

Une cryptanalyse partielle :

L'attaquant découvre le texte clair correspondant à un ou plusieurs messages chiffrés interceptés.

Une cryptanalyse totale :

L'attaquant découvre un moyen pour déchiffrer tous les messages, aussi bien ceux qu'il a interceptés que ceux à venir, par exemple en découvrant la clé utilisée.

Selon les moyens dont dispose l'attaquant, on distingue plusieurs types d'attaques. Par ordre de moyens croissants ces types sont:

- **Attaque à messages chiffrés connus :**

  L'attaquant a seulement la possibilité d'intercepter un ou plusieurs messages chiffrés.

- **Attaque à messages clairs connus :**

  L'attaquant dispose d'un ou plusieurs messages clairs avec les messages chiffrés correspondants.

- **Attaque à messages clairs choisis :**

  L'attaquant a la possibilité d'obtenir la version chiffrée de messages clairs de son choix.

- **Attaque à messages chiffrés choisis :**

  L'attaquant a temporairement l'opportunité de déchiffrer les messages de son choix (en ayant accès par exemple à une machine déchiffrante). Il tente alors d'en profiter pour obtenir des informations lui permettant de déchiffrer ensuite d'autres messages par ses propres moyens.

1.5 Quelques systèmes cryptographiques classiques

Les systèmes cryptographiques classiques utilisent la correspondance entre les lettres alphabétiques et l'anneau $\mathbf{Z}_{26}=Z/26Z$, A correspond à 0, B à 1,…, et Z à 25.

La table de correspondance complète est représenté comme suit :

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

1.5.1 Chiffrement par décalage (cf [18])

Ce chiffrement consiste à décaler les lettres de l'alphabet de quelques crans vers la droite ou la gauche. Il est basé sur l'arithmétique modulaire.

Définition 1-15

Soient a et b deux entiers relatifs, si $a-b$ est multiple de $n \in$IN on dit que a est congru à b modulo n et on écrit $a \equiv b \bmod n$.

Le chiffrement par décalage est représenté comme suit :

$$\wp = \mathrm{C} = K = \mathbf{Z}_{26} \quad \text{pour} \quad 0 \le k \le 25.$$

Ce chiffrement se définit dans $\mathbf{Z}_{26}$. On utilise les 26 lettres de l'alphabet. Mais on pourrait le définir dans n'importe quel anneau $\mathbf{Z}_{m}$.

Cas particuliers:

1) Pour $k = 3$ le système cryptographique est appelé chiffrement de César. C'est le plus naïf de tous les systèmes cryptographiques. Historiquement c'est le plus ancien. César s'en est servi pour envoyer des messages au Sénat romain pendant ses campagnes.

2) Pour k =13 le système cryptographique est appelé Rot 13 (Rot vient de rotation).

Si on applique Rot 13 deux fois à un message, on obtient le message originel.

Sécurité du chiffrement par décalage

Le système de chiffrement par décalage n'est pas sûr, car il peut être cryptanalysé par la méthode de recherche exhaustive.

Comme il n'y a que 26 clés possibles, les cryptanalystes peuvent essayer le déchiffrement avec toutes les clés jusqu'à ce qu'ils trouvent un texte clair compréhensible et aisé.

Exemple 1-1

Supposons que la clé du chiffrement par décalage soit $k = 5$ et que le texte clair soit : Cryptographie.

Tout d'abord, convertissons ce texte en une suite d'entiers en utilisant la table de correspondance, on obtient :

2 17 24 15 19 14 6 17 0 15 7 8 4

Ensuite, ajoutons 5 à chaque valeur et réduisons modulo 26 :

7 22 3 20 24 19 11 22 5 20 12 13 9

Enfin, convertissons cette suite d'entiers en caractère alphabétiques. On obtient le texte chiffré : hwduytlwfumnj

Pour déchiffrer ce texte, le destinataire doit d'abord convertir les lettres du texte en nombres, puis soustraire 5 à chaque valeur en réduisant modulo 26, et enfin convertir les nombres en caractères alphabétiques.

1.5.2 Chiffrement affine (cf [18])

Ce chiffrement est une amélioration du chiffrement par décalage. Il est basé sur des fonctions affines.

Définition1-16

Le p gcd de deux entiers naturels a et b est le plus grand élément de l'ensemble de leurs diviseurs communs, on le note $p\gcd(a,b)$.

Le chiffrement affine est représenté comme suit :

Soit $\wp = \mathrm{C} = \mathbf{Z}_{26}$ et $K = \{(a,b) \in \mathbf{Z}_{26} \times \mathbf{Z}_{26} : p\gcd(a,26)=1\}$

Pour $k=(a,b) \in K$, on définit la fonction de chiffrement e_k et la fonction de déchiffrement d_k par :

$$e_k(x) = ax + b \bmod 26 \quad \text{et} \quad d_k(y) = a^{-1}(y-b) \bmod 26$$

Définition1-17

L'indicatrice d'Euler est l'application ϕ: IN* $\rightarrow$ IN de valeur ϕ(n) égale au nombre des entiers positifs inférieurs à n et premiers avec n.

Sécurité du chiffrement affine

Le nombre de clés dans le chiffrement affine sur $\mathbf{Z}_{\mathrm{m}}$ est égal à mϕ(m).Donc la méthode de recherche exhaustive sur $\mathbf{Z}_{\mathrm{m}}$, pour m assez grand , n'est pas rentable.

Le chiffrement affine est attaquable par l'analyse des fréquences d'apparition des lettres. Dans un texte français ou anglais , le symbole qui apparaît le plus fréquemment, remplace probablement le e (voir la table 1).

Par exemple si dans le texte chiffré le nombre d'occurrences de a est 8, c est 5, f est 3 et k est 1. On suppose tout d'abord que a correspond à e et que c correspond à t car t et e sont respectivement les plus fréquentes donc : $e_k(4) = 0$ et $e_k(19) = 2$. Ce qui nous donne le système suivant :

$$\begin{cases} 4a + b = 0 \\ et \\ 19a + b = 2 \end{cases} \quad \dots (1)$$

Le système (1) n'admet pas de solution dans $\mathbf{Z}_{26} \times \mathbf{Z}_{26}$.

L'hypothèse est donc fausse. On peut supposer que a correspond à e et que f correspond à t. Si ça marche c'est bon, si non on fait correspondre a à e et k à t. Et ainsi de suite jusqu'à ce qu'on trouve une solution $(a,b) \in \mathbf{Z}_{26} \times \mathbf{Z}_{26}$ tel que : $p \gcd(a,b) = 1$.Nous obtenons le texte clair convenable.

Exemple 1-2

Supposons $k = (11,15)$. La fonction de chiffrement est $e_k(x) = 11x + 15 \bmod 26$.

Donc la fonction de déchiffrement est

$$d_k(y) = 11^{-1}(y - 15) \bmod 26.$$

Le message clair : **nombre premier** correspond à la suite :

13 14 12 1 17 4 15 17 4 12 8 4 17

En utilisant la fonction de chiffrement, on obtient la suite :

2 13 17 0 20 7 24 20 7 17 25 7 20

qui correspond au message chiffré : **cnrauhyuhrzhu** .

En utilisant la fonction de déchiffrement, on obtient la suite :

13 14 12 1 17 4 15 17 4 12 8 4 17

qui correspond au message déchiffré : **nombre premier** qui est bien le message clair.

	Anglais	Français		Anglais	Français
a	8.17	8.25	n	6.75	7.25
b	1.49	1.25	o	7.51	5.75
c	2.78	3.25	p	1.93	3.75
d	4.25	3.75	q	0.10	1.25
e	12.70	17.75	r	5.99	7.25
f	2.23	1.25	s	6.33	8.25
g	2.02	1.25	t	9.06	7.25
h	6.09	1.25	u	2.76	6.25
i	6.97	7.25	v	0.98	1.75
j	0.15	0.75	w	2.36	0.00
k	0.77	0.00	x	0.15	0.00
l	4.03	5.75	y	1.97	0.75
m	2.41	3.25	z	0.07	0.00

Table 1 : Fréquences des lettres en anglais et en français (en %)

1.5.3 Chiffrement par substitution (cf [18]).

Le chiffrement par substitution à été utilisé pendant des centaines d'années. Il consiste à remplacer chaque caractère du texte en clair par un autre caractère dans le texte chiffré.

Le destinataire applique la substitution inverse au texte chiffré pour retrouver le texte en clair.

Ce chiffrement est représenté comme suit :

Soit $\wp = C = \mathbf{Z}_{26}$. K est l'ensemble des permutations de 26 nombres sur l'ensemble des 26 nombres 0, 1, …, 25.

Pour chaque permutation $\pi \in K$, on définit la fonction de chiffrement e_k et la fonction de chiffrement d_k par :

$$e_\pi(x) = \pi(x)$$

et

$$d_\pi(x) = \pi^{-1}(x)$$

où π^{-1} est la permutation réciproque de π.

Sécurité du chiffrement par substitution

Le chiffrement par substitution est attaquable par l'analyse des fréquences des lettres. Mais cette méthode de cryptanalyse dans ce cas est plus compliquée parce que la correspondance concerne toutes les lettres possibles du texte chiffré.

1.5.4 Chiffrement de Vigenère (cf [18])

Dans les chiffrements précédents , dés qu'une clé est fixée, chaque caractère alphabétique est transformé en un unique caractère alphabétique. Pour cette raison , le procédé est appelé monoalphabétique . Plus tard en 1586, Blaise de Vigenère (diplomate français) propose un système basé sur la substitution polyalphabétique, qui traite un caractère alphabétique à la fois.

Le chiffrement de Vigenère est représenté comme suit :

Soit m un entier strictement positif. Soit $\wp = C = K = (\mathbf{Z}_{26})^m$.

Pour $k = (k_1, k_2, ..., k_m)$, on définit la fonction de chiffrement e_k et la fonction de chiffrement d_k par :

$$e_k(x_1, x_2, ..., x_m) = (x_1 + k_1, x_2 + k_2, ..., x_m + k_m)$$

et

$$d_k(y_1, y_2, ..., y_m) = (y_1 - k_1, y_2 - k_2, ..., y_m - k_m)$$

où les opérations sont effectuées dans $\mathbf{Z}_{26}$.

Sécurité du chiffrement de Vigenère

La grande force du chiffrement de Vigenère est que la même lettre est chiffrée de différentes manières, ce qui rend inutilisable la méthode de l'analyse des fréquences.

Le chiffrement de Vigenère a résisté trois siècles aux cryptanalyses ; il est pourtant relativement facile à casser, grâce à la méthode de kasiski (utilisation de l'indice de coïncidence).

1.5.5 Chiffrement de Hill (cf [18]).

Le chiffrement de Hill est un autre système cryptographique polyalphabétique, inventé en 1929 par Lester S. Hill. Il s'agit de transformer m caractères d'un bloc du texte chiffré par des combinaisons linéaires de ces caractères.

Ce chiffrement est présenté ainsi :

Soit m un entier strictement positif.

Soit $\wp$ = C = $(\mathbf{Z}_{26})^m$ et K= {matrices m × m inversibles dans $\mathbf{Z}_{26}$}.

Pour toute clé k , on définit la fonction de chiffrement e_k et la fonction de chiffrement d_k par :

: et $e_k(x) = xk$ $d_k(x) = yk^{-1}$

où toutes les opérations sont faites dans l'anneau $\mathbf{Z}_{26}$.

Sécurité du chiffrement de Hill

Le chiffrement de Hill est très difficile à casser par une attaque à texte chiffré connu, alors qu'il est facile à casser avec une attaque à texte clair choisi.

1.5.6 Chiffrement par permutation (cf [18]).

Les systèmes cryptographiques précédents reposent sur une substitution : chaque caractère du texte clair est remplacé par un autre dans le texte chiffré alors que le chiffrement par permutation conserve les mêmes caractères entre le chiffrement par permutation. Le chiffrement par substitution fut énoncé pour la première fois en 1563 par Giovanni Porta.

Le chiffrement à été utilisée pendant des centaines d'années. Nous représentons le chiffrement par permutation comme suit :

Soit un entier strictement positif m. Soit $\wp$ = C = $\{0, 1, \ldots, 25\}^m$ et K l'ensemble des permutations de $\{1, \ldots, m\}$. Pour toute clé π (c'est-à-dire pour toute permutation), on définit la fonction de chiffrement e_k et la fonction de chiffrement d_k par :

et $e_\pi(x_1,\ldots,x_m) = (x_{\pi(1)},\ldots,x_{\pi(m)})$ $d_\pi(y_1,\ldots,y_m) = (y_{\pi^{-1}(1)},\ldots,y_{\pi^{-1}(m)})$

où π^{-1} est la permutation inverse de π .

1.6 Conclusion

La cryptographie classique utilisait des cryptosystèmes basés sur les lettres .
De nos jours on manipule des bits, au lieu des caractères .On passe de 26 élément à deux éléments.

On va voir par la suite les deux types de cryptographie moderne : la cryptographie à clé secrète et la cryptographie à clé publique.

2.1 Introduction

La cryptographie à clé secrète, appelée aussi à clé unique ou symétrique, est utilisée depuis déjà plusieurs siècles. C'est l'approche la plus authentique du chiffrement de donnée et mathématiquement la moins problématique. Son principe fondamental est le choix pour chaque couple d'utilisateurs (émetteur, destinataire) d'une clé secrète utilisée à la fois pour le chiffrement et le déchiffrement. Cette clé doit être échangée par un canal sûr.

2.2 Chiffrement par bloc

Dans ce type de chiffrement, il y a une séparation du texte clair en blocs d'une longueur fixe. Un algorithme chiffre un bloc à la fois.

Les systèmes de chiffrement par blocs nécessitent une recherche approfondie du choix de la clé. Les clés très longues sont plus coûteuses en travail à cause de leur génération et de leur transmission. La taille des blocs a un impact sur la sécurité et sur la complexité : les blocs de grande taille sont plus sécuritaires mais sont plus lourds à implémenter

2.3 Le chiffrement DES (cf [1]).

Le système cryptographique le plus utilisé dans le monde est le Data Encryption Standard (DES) publié en 1975 et adopté par le Bureau National de Standard en 1977

Le DES est largement utilisé dans le monde des affaires aux Etats Unis :conversations téléphoniques, transactions bancaires et différents types de données chiffrées.

Le DES chiffre un bloc de texte clair de 64 bits en utilisant une clé secrète k de 56 bits, pour obtenir un bloc de texte chiffré de 64 bits.

La clé de 56 bits est contenue dans un bloc – clé de 64 bits.

Les 8 bits de parité (ou bits de détection d'erreur) sont : 8, 16, 24, 32, 48 ,56 et 64 du bloc- clé. Ces 8 bits sont tels que chaque octet contient un nombre impair de 1.

2.3.1 Structure globale du DES

L'algorithme se déroule en trois étapes. (Voir figure2.1)

1) L'application d'une permutation initiale IP d'un bloc de 64 bits de l'entrée à chiffrer.
2) 16 tours d'une certaine fonction sont effectuées en utilisant 16 clés partielles calculées à partir de la clé secrète K.
3) L'application d'une permutation finale IP^{-1} qui est l'inverse de la permutation initiale.

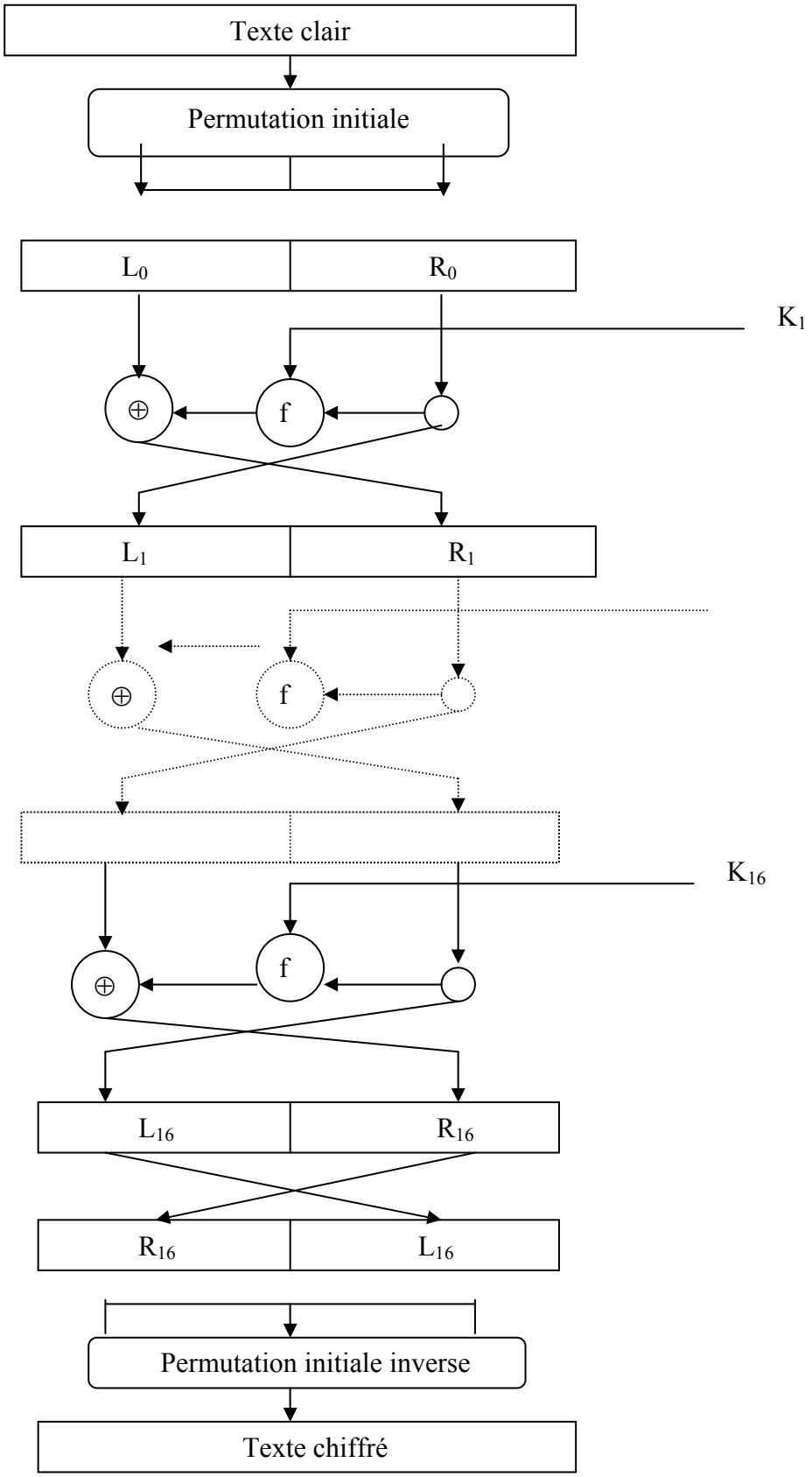


Figure2.1. Structure globale de DES

2.3.2 Les permutations IP et IP^{-1}

La permutation initiale IP et son inverse IP^{-1} sont décrites par la figure 2.2.

Les tableaux se lisent de gauche à droite et de haut en bas, le n - iéme nombre est la position avant permutation du bit qui se retrouve en n- ième position après permutation.

Permutation initiale

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

Permutation initiale inverse

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

Figure 2.2. La permutation initiale et son inverse

2.3.3 Les 16 Tours

Après la permutation initiale, le message est séparé en deux moitiés de 32 bits, désignées par L_0 et R_0 . Pour chaque tour de l'algorithme , On détermine deux blocs de 32 bits L_i et R_i en fonction de L_{i-1} e t R_{i-1} obtenus précédemment.

Pour cela, on utilise une clé intermédiaire K_i de 48 bits, calculée à partir de la clé secrète K et on applique les formules suivantes :

$$\mathbf{L_i = R_{i-1}} \quad \textbf{et} \quad \mathbf{R_i = L_{i-1} \oplus f(R_{i-1}, K_i).}$$

où $\oplus$ est une addition bit à bit modulo 2 (ou exclusif) et f est décrite par la suite.

2.3.4 La fonction f

Le calcul de la fonction f (voir figure 2.3) se fait comme suit :

Tout d'abord le bloc de 32 bits à gauche est expansé en appliquant une fonction d'expansion E.

(Voir figure 2.4). Ensuite, on calcule le ou exclusif du bloc obtenu avec la clé partielle K_i.

Le résultat est un bloc de $48 = 8 \times 6$ bits. Il est transformé en un autre bloc de $32 = 8 \times 4$ bits en utilisant des S – boites. Enfin, on applique la permutation IP_{32} décrite par la figure 2.4 à ces 32 bits pour obtenir la valeur de f.

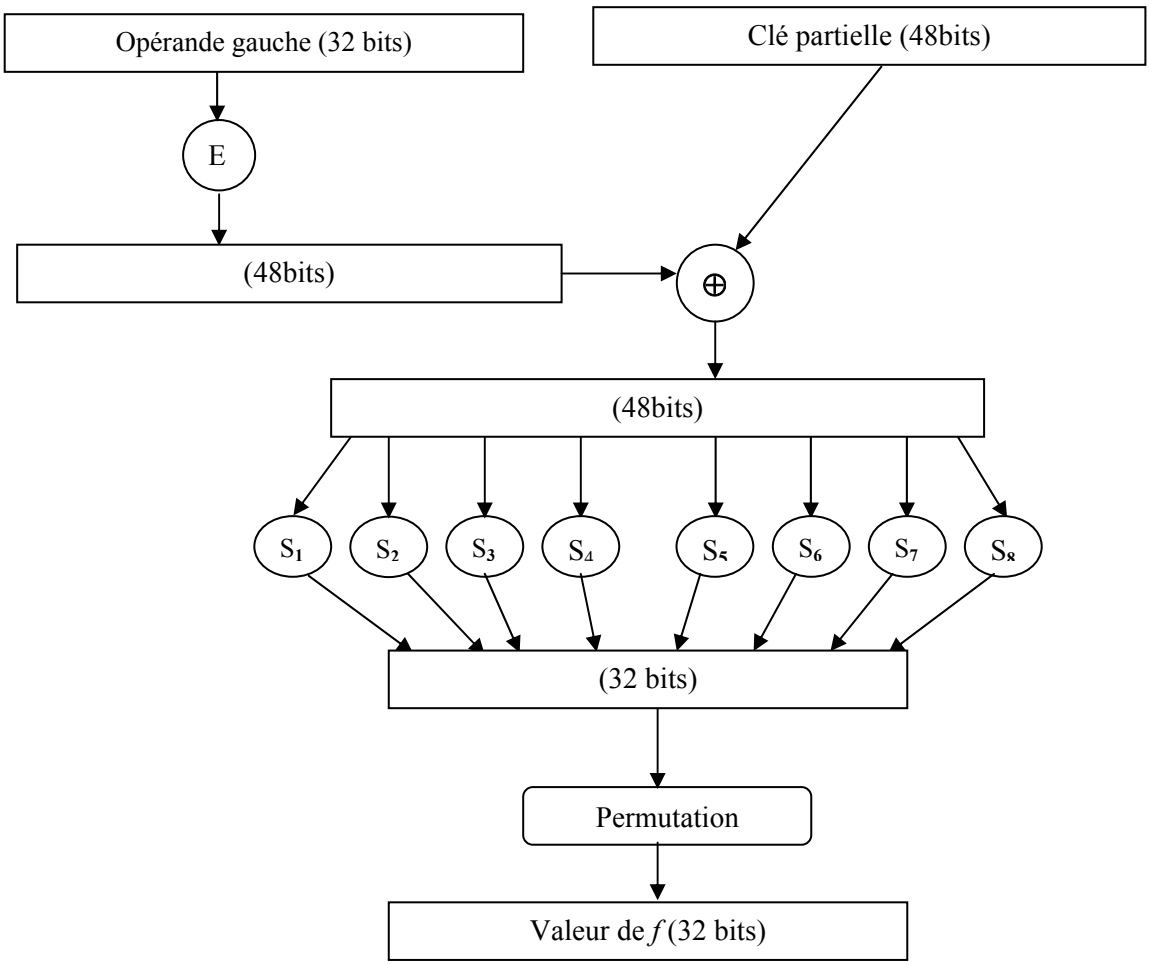


Figure 2.3. Schéma de la fonction *f*

Fonction E d'expansion

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

Permutation IP_{32} finale

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

Figure 2.4 L'expansion de premier argument et la permutation finale.

2.3.5 Les S- boite

Il y a huit S- boite différentes (voir figure 2.5) qui calculent un bloc de 4 bits à partir d'un bloc de 6 bits.

On les représente par des tableaux à 2 lignes et 16 colonnes, les premiers et derniers bits de l'entrée déterminent une ligne de tableau , les autre bits déterminent une colonne, La valeur numérique trouvée à cet endroit indique la valeur des quatre bits de sortie.

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

S_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Figure 2.5. Les S – boites

2.3.6 Calcul des clés partielles

Les 16 clés partielles sont calculées de la manière suivante :

On applique d'abord une permutation PC_1 à K, puis, à chacune des 16 étapes, on applique à chaque moitié du bloc de 56 bits obtenu une rotation à gauche, d'un cran aux étapes 1, 2, 9, 16, et de deux cran aux autre étapes. A chacune de ces étapes, on obtient une clé partielle de 48 bits en appliquant la règle d'extraction PC_2 .

PC_1 et PC_2 sont décrites par la figure 2.6. Les 56 bits de K sont numérotés de 1 à 64 en évitant les bits de parité.

Permutation PC_1

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

Règle d'extraction PC_2

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

Figure 2.6. La permutation PC_1 et la règle PC_2

2.3.7 Le déchiffrement du DES

Pour le déchiffrement du DES, on applique le même algorithme de chiffrement en générant les clés partielles du chiffrement dans l'ordre inverse.

Exemple 2-1 (de chiffrement par DES)

Considérons le bloc de texte clair (en hexadécimal) suivant :

0123456789ABCDEF

et la clé (hexadécimal)

133457799BBCDFF1

En binaire, la clé sans les bits de parité est

0001001001101001010110111100100110110111101101111111000

En appliquant IP, on obtient L_0 et R_0 en binaire :

L_0 =11001100000000001100110011111111

$L_1=R_0$ =11110000101010101111000010101010

Les seize tours se déroulent ainsi :

$E(R_0)$ = 011110100001010101010101011110100001010101010101
K_1 = 000110110000001011101111111111000111000001110010
$E(R_0)+K_1$ = 011000010001011110111010100001100110010100100111
Sorties de S- boites 01011100100000101011010110010111
$f(R_0, K_1)$ = 00100011010010101010100110111011
$L_2 = R_1$ = 11101111010010100110010101000100

$E(R_1)$ = 011101011110101001010100001100001010101000001001
K_2 = 011110011010111011011001110110111100100111100101
$E(R_1)+K_2$ = 000011000100010010001101111010110110001111101100
Sorties de S- boites 11111000110100000011101010101110
$f(R_0, K_1)$ = 00111100101010111000011110100011
$L_3 = R_2$ = 11001100000000010111011100001001

$E(R_2)$ = 111001011000000000000010101110101110100001010011
K_3 = 010101011111110010001010010000101100111110011001
$E(R_2)+K_3$ = 101100000111110010001000111110000010011111001010
Sorties de S- boites 00100111000100001110000101101111
$f(R_2, K_3)$ = 01001101000101100110111010110000
$L_4 = R_3$ = 10100010010111000000101111110100

$E(R_3)$	= 01010000010000101111100000000101011111111010101001
K_4	= 01110010101011011101011011011011001101010001110 1
$E(R_3)+K_4$	= 00100010111011110010111011011110010010101011010 0
Sorties de S- boites	00100001111011011001111100111010
$f(R_3 , K_4)$	= 10111011001000110111011101001100
$L_5 = R_4$	= 01110111001000100000000001000101

$E(R_4)$	= 10111010111010010000010000000000000000001000001010
K_5	= 01111100111011000000011111101011010100111010100 0
$E(R_4)+K_5$	= 11000110000001010000001111101011010100011010001 0
Sorties de S- boites	01010000110010000011000111101011
$f(R_4, K_5)$	= 00101000000100111010110111000011
$L_6= R_5$	= 10001010010011111010011000110111

$E(R_5)$	= 11000101010000100101111111010000110000011010111 1
K_6	= 01100011101001010011111001010000011110110010111 1
$E(R_5)+K_6$	= 10100110111001110110000110000000101110101000000 0
Sorties de S- boites	01000001111100110100110000111101
$f(R_5 , K_6)$	= 10011110010001011100110100101100
$L_7 = R_6$	= 11101001011001111100110101101001

$E(R_6)$	= 11110101001010110000111111100101101010110101001 1
K_7	= 11101100100001001011011111110110000110001011110 0
$E(R_6)+K_7$	= 00011001101011111011100000010011101100111110111 1
Sorties de S- boites	00010000011101010100000010101101
$f(R_6 , K_7)$	= 10001100000001010001110000100111
$L_8 = R_7$	= 00000110010010101011101000010000

$E(R_7)$	= 00000000110000100101010101011111010000001010000 0
K_8	= 11110111100010100011101011000001001110111111101 1
$E(R_7)+K_8$	= 11110111010010000110111110011110011110110101101 1
Sorties de S- boites	01101100000110000111110010101110
$f(R_7, K_8)$	= 00111100000011101000011011111001
$L_9= R_8$	=11010101011010010100101110010000

$E(R_8)$	= 01101010101010110101001010100101011111001010000 1
K_9	= 11100000110110111110101111101101111001111000000 1
$E(R_8)+K_9$	= 10001010011100001011100101001000100110110010000 0
Sorties de S- boites	00010001000011000101011101110111
$f(R_8, K_9)$	= 00100010001101100111110001101010
$L_{10}= R_9$	= 00100100011111001100011001111010

$E(R_9)$ = 000100001000001111111001011000001100001111110100
K_{10} = 101100011111001101000111101110100100011001001111
$E(R_9)+K_{10}$ = 101000010111000010111110110110101000010110111011
Sorties de S-boites 11011010000001000101001001110101
$f(R_9 , K_{10})$ = 011000101011110010011100001000010
$L_{11} = R_{10}$ = 101101111101010111010111110110010

$E(R_{10})$ = 010110101111111010101011111010101111110110100101
K_{11} = 001000010101111111010011110111101101001110000110
$E(R_{10})+K_{11}$ = 011110111010000101111000001101000010111000100011
Sorties de S-boites 01110011000001011101000100000001
$f(R_{10} , K_{11})$ = 111000010000010011111101000000010
$L_{12} = R_{11}$ = 110001010111100000111110001111000

$E(R_{11})$ = 011000001010101111110000000111111000001111110001
K_{12} = 011101010111000111110101100101000110011111101001
$E(R_{11})+K_{12}$ = 000101011101101000000101100010111110010000011000
Sorties de S-boites 01111011100010110010011000110101
$f(R_{11} , K_{12})$ = 110000100110100011001111111101010
$L_{13} = R_{12}$ = 011101011011110100011000010110000

$E(R_{12})$ = 001110101011110111111010100011110000001011110000
K_{13} = 100101111100010111010001111110101011101001000001
$E(R_{12})+K_{13}$ = 101011010111100000101011011101011011100010110001
Sorties de S-boites 10011010110100011000101101001111
$f(R_{12} , K_{13})$ = 110111011011101100101001001000010
$L_{14} = R_{13}$ = 000110001100001100010101011010

$E(R_{13})$ = 000011110001011000000110100010101010101011110100
K_{14} = 010111110100001110110111111100101110011100111010
$E(R_{13})+K_{14}$ = 010100000101010110110001011110000100110111001110
Sorties de S-boites 01100100011110011001101011110001
$f(R_{13} , K_{14})$ = 101101110011000110001110010101010
$L_{15} = R_{14}$ = 110000101000110010010110000001101

$E(R_{14})$ = 111000000101010001011001010010101100000001011011
K_{15} = 101111111001000110001101001111010011111100001010
$E(R_{14})+K_{15}$ = 010111111100010111010100011101111111111101010001
Sorties de S-boites 10110010111010001000110100111100
$f(R_{14} , K_{15})$ = 010110111000000100100111011011110
$L_{16} = R_{15}$ = 010000110100001000110010001101000

$E(R_{15})$ = 00100000011010100000010000011010010000011010100
K_{16} = 11001011001111011000101100001110000101111111010 1
$(R_{15})+K_{16}$ = 11101011010101111000111100010100010101100101 1101
Sorties de S-boites 10100111100000110010010000101001
(R_{15}, K_{16}) = 11001000110000000100111110011000
R_{16} = 00001010010011001101100110010101

Enfin, en appliquant IP^{-1} à R_{16} et L_{16}, on obtient le bloc de texte chiffré (en hexadécimal) 85 E813540F0AB405.

2.3.9. Sécurité de DES

Comme tout système cryptographique, le niveau de sécurité de DES dépend de l'une de ses cryptanalyses.

Depuis 1990, de nouvelles et puissantes techniques de cryptanalyse des systèmes de type DES sont en effet apparus, telles celles basées sur la cryptanalyse différentielle ou linéaire.

La cryptanalyse différentielle a été introduite par Biham et Shamir. C'est une attaque à messages clairs choisis. Elle permet de casser facilement un DES à 8 ou à 10 tours. Le DES complet à 16 tours est resté hors de portée de cette attaque.

La cryptanalyse linéaire a été introduite par H. Gilbert et Matsui. C'est une attaque à messages clairs connus et utilisable pour un DES restreint à quelques tours.

Le DES complet à 16 tours n'est pas menacé par cette attaque.

La première technique de cryptanalyse du DES plus efficace que les deux cryptanalyses précédentes est la technique de recherche exhaustive : recherche parmi les 2^{56} clés possibles.

En Janvier 1997, les laboratoires RSA ont lancé un défi consistant à déchiffrer par recherche exhaustive un message chiffré par DES . Ils ont réussi à trouver la bonne clé le 17 Juin 1997 après avoir exploré environ un quart de l'espace des clés moyennant des milliers d'ordinateurs.

Grâce à l'amélioration des performances des ordinateurs ces vingt dernières années, la taille de la clé secrète de 56 bits, rend le DES aujourd'hui vulnérable.aux attaques exhaustives. Il existe bien une amélioration du DES, triple DES, qui double la taille de clé, mais il n'est pas assez rapide.

2.4 Le chiffrement Rijndael (AES) (cf [4], [21]).

L'AES (Advanced Encryption Standard) est comme son nom l'indique, un standard de chiffrement symétrique destiné à remplacer le DES qui est devenu trop faible au regard des attaques actuelles.

Le développement de l'AES à été institué par le NIST (National Institut of Standards and Technology) le 2 janvier 1997 ; l'algorithme à été choisi il y a peut de temps : il s'agit de l'algorithme Rijndael qui a été conçu par Joan Daemen et Vincent Rijmen, deux chercheurs Belges.

2.4.1 Rijndael

Rijndael est un algorithme de chiffrement par blocs, à plusieurs tours, avec une taille de blocs et de clés variables, choisis entre 128, 192 et 256 bits.

Définition 2-1

On appelle état un résultat intermédiaire de l'algorithme.

L'état est représenté par un tableau d'octets (un octet représente 8 bits) rectangulaire de 4 lignes, et d'un nombre de colonnes N_b égal à $\frac{taille\ du\ bloc}{32}$.On représente la clé de la même façon. Le nombre de colonnes N_k est égal à $\frac{longueur\ de\ clé}{32}$.

Au début de l'algorithme, on remplit l'état avec le bloc à chiffrer dans l'ordre $a_{0,0}$ $a_{1,0}$ $a_{2,0}$ $a_{3,0}$ $a_{0,1}$ $a_{1,1}$…..et on procède de la même façon pour la clé.

Exemple 2-2 (exemple d'état avec des blocs de 192 bits $N_b = 6$ et de clé de longueur 128 bits $N_k = 4$)

Le bloc à chiffrer

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$

la clé de chiffrement

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

2.4.2 Les itérations

Le nombre de tours de Rijndael dépend de la longueur du bloc et de la clé selon le tableau suivant :

N_r	N_b = 4(128 bits)	N_b = 6(192 bits)	N_b = 8(256 bits)
N_k = 4(128 bits)	10	12	14
N_k = 6(192 bits)	12	12	14
N_k = 8(256 bits)	14	14	14

Chaque tour utilise une sous – clé K_i différente (générée à partir de la clé principale) et est composé de quatre transformations qui agissent sur l'état actuel K_i

Byte Sub (Etat) ; ShiftRow (Etat) ; Mixcolumn (Etat) ; AddRound Key (Etat K_i)

2.4.3 Le corps GF (2^8)

Un octet b , composé des 8 bits b_7 , b_6 , b_5 , b_4 , b_3, b_2 , b_1 , b_0 ,peut être considéré comme un polynôme de degré inférieur ou égal à 7 avec des coefficient dans { 0 ,1 }

L'addition de deux de ces polynômes est l'addition usuelle modulo 2. Cette addition correspond au XOR au niveau des bits.

Pour la multiplication de deux polynômes, c'est la multiplication usuelle (double distributivité), suivie d'une réduction modulo un polynôme binaire irréductible de degré 8.

Dans Rijndael ce polynôme est $m(x) = x^8 + x^4 + x^3 + x + 1$

Le résultat est un polynôme de degré inférieur ou égal à 7.

Pour tout polynôme binaire de degré inférieur à 8 l'algorithme d'Euclide étendu permet de calculer b(x) tel que $a(x)\ b(x) = 1 \bmod m(x)$, autrement dit, de calculer l'inverse $a^{-1}(x)$ de $a(x)$.

L'ensemble des 256 bits possibles, avec l'addition ci – dessus a une la structure de corps fini GF (2^8).

2.4.4 La transformation Bytesub

Pour chacun des octets $a_{i,j}$ de l'état, on applique les deux transformations suivantes

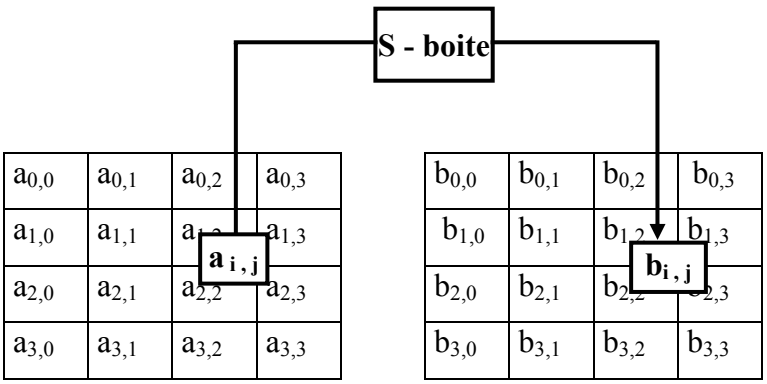
1) On considère $a_{i,j}$ comme polynôme dans GF(2^8), et on prend son inverse $a_{i,j}^{-1}$:

2) On calcule l'image du résultat par la fonction y = f (*x*) suivante :

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Ces deux transformations mises à la suite forment la S – boite, qui est appliquée à chacun des octets de l'état : $b_{i,j}$ = S – boite ($a_{i,j}$).

La S – boite peut être implémentée efficacement par une table qui contient l'image de chacune des 256 valeurs d'entrée possible :



.

La S- boîte est représentée en hexadécimal par la table suivante :

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

Où ByteSub ({ ij }) est déterminé par l'intersection de la ligne i avec la colonne j (i et j sont des quartez).

Par exemple ByteSub ({56}) = {b 1}.

2.4.5 La transformation ShiftRow

La transformation ShiftRow effectue un décalage cyclique vers la gauche des lignes de l'état.

La ligne 0 n'est pas décalée, la ligne 1 l'est de C_1 octets, la 2 de C_2 octets, et la ligne3 de C_3 octets.

Les valeurs de C_1, C_2 et C_3 dépendent de la longueur du bloc, selon la table suivante :

N_b	C_1	C_2	C_3
4	1	2	3
6	1	2	3
8	1	3	4

Exemple 2-3

La transformation ShiftRow d'un bloc de 128 bits ($N_b = 4$)

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	→	Pas de décalage	→	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	→	Décalage de 1	→	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,0}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	→	Décalage de 2	→	$a_{2,2}$	$a_{2,3}$	$a_{2,0}$	$a_{2,1}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	→	Décalage de 3	→	$a_{3,3}$	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$

2.4.6. La transformation MixColumn

La transformation MixColumn consiste à prendre chaque colonne de l'état et à la multiplier par une matrice M dont les coefficients sont soit 1, soit 2, soit 3, codés en binaire :

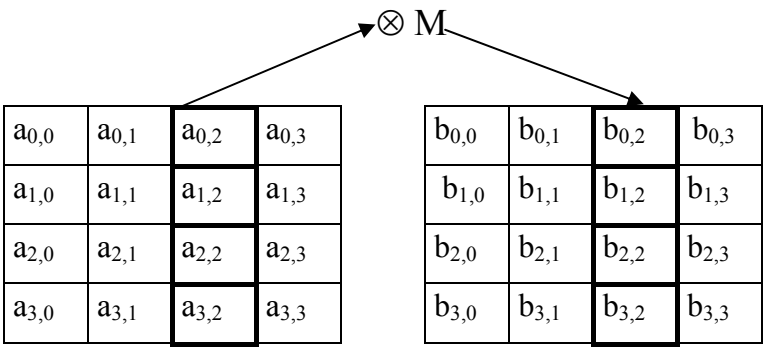
00000001, 00000010, 00000011.

Les opérations de cette matrice ne sont pas des multiplications et des additions classiques, mais sont effectuées dans le corps GF(2^8). Le choix des coefficients de la matrice à été déterminé par la facilité d'implémentation.

$$M = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

Exemple 2-4

La transformation MixColumn d'un bloc de 128 bits ($N_b = 4$).



2.4.7 La transformation AddroundKey

La transformation AddroundKey consiste simplement à faire un XOR de chaque octet de l'état avec l'octet correspond de la sous – clé du tour $a_{i,j} \oplus k_{i,j} = b_{i,j}$

Exemple 2-5

La transformation AddroundKey d'un bloc de 128 bits (N_b =4) avec une sous – clé de 128 bits (N_k=4).

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

=

$d_{0,0}$	$d_{0,0}$	$d_{0,0}$	$d_{0,0}$
$d_{1,0}$	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$
$d_{2,0}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$
$d_{3,0}$	$d_{3,1}$	$d_{3,2}$	$d_{3,3}$

2.4.8. La clé étendue

La clé étendue est un vecteur W composé de mots de 4 octets W[i] = (a, b, c, d), de longueur N_b (Nr+1).

Les N_k premiers mots W[0] …W [N_k -1] représentent la clé secrète et les mots suivants sont calculés en faisant un XOR du mot précédent (W[i - 1]) et du mot situé N_k positions avant (W[i - N_k]).

Pour les mots situés sur une position qui est un multiple de N_k une fonction g est appliquée à W [i-1] avant le XOR.

2.4.9. Description de g

La fonction g(voir figure 2.7.) comprend dans l'ordre les deux transformations suivantes :

Une permutation cyclique d'un cran vers la gauche (Rot word). Elle transforme le mot (a, b, c, d) en le mot (b, c, d, a).

L'application de la S-boite séparément sur chaque octet du mot et d'un XOR avec RCon [i], où RCon [i] est un vecteur qui dépend du tour i.

RCon[i] = ({02} i-1, {00}, {00}, {00}) et i ≥ 1.

00 et 02 sont deux octets si la clé de 256 bits, et que I – 4 est un multiple de 4, on appliqué encore la S-boite séparément sur chaque octet du mot W [i - 1] avant de faire le XOR final.

Au fur et à mesure des tours, on prend les clés partielles nécessaires les unes à la suite des autres, pour le tour i on prend les mots W[j] avec $i.N_b \leq j \leq (i + 1).\ N_b$.

2.4.10 Structure globale de Rijndael (voir figure 2.8)

1) On calcule la clé étendue.
2) On effectue un AddroundKey initial (tour 0)
3) On effectue $N_r - 1$ tours :
   Chaque tour i comprend les 4 transformations :
   3.1 ByteSub (état).
   3.2 ShiftRow (état).
   3.3 MixColumn (état).
   3.4 AddroundKey (état, k_i).
4) On effectue un tour final :
   4.1 ByteSub (état).
   4.2 ShiftRow (état).
   4.3 AddroundKey (état, k_i).

2.4.11 Le Déchiffrement de Rijndael

Le déchiffrement de Rijndael se fait de la même façon que le chiffrement, mais en utilisant les inverses des différentes informations et en générant les clés partielles de manière inversée.

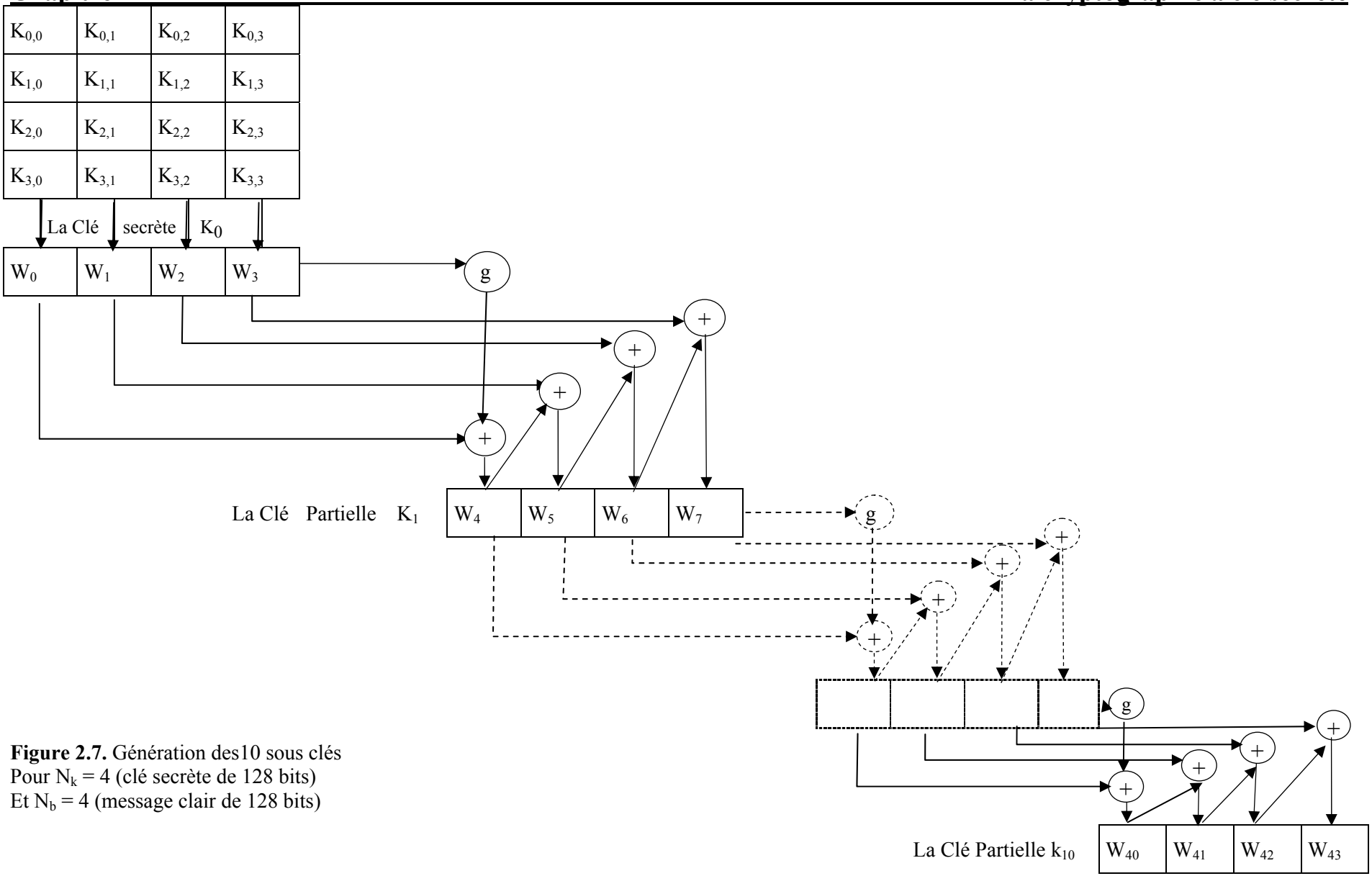


Figure 2.7. Génération des10 sous clés
Pour $N_k = 4$ (clé secrète de 128 bits)
Et $N_b = 4$ (message clair de 128 bits)

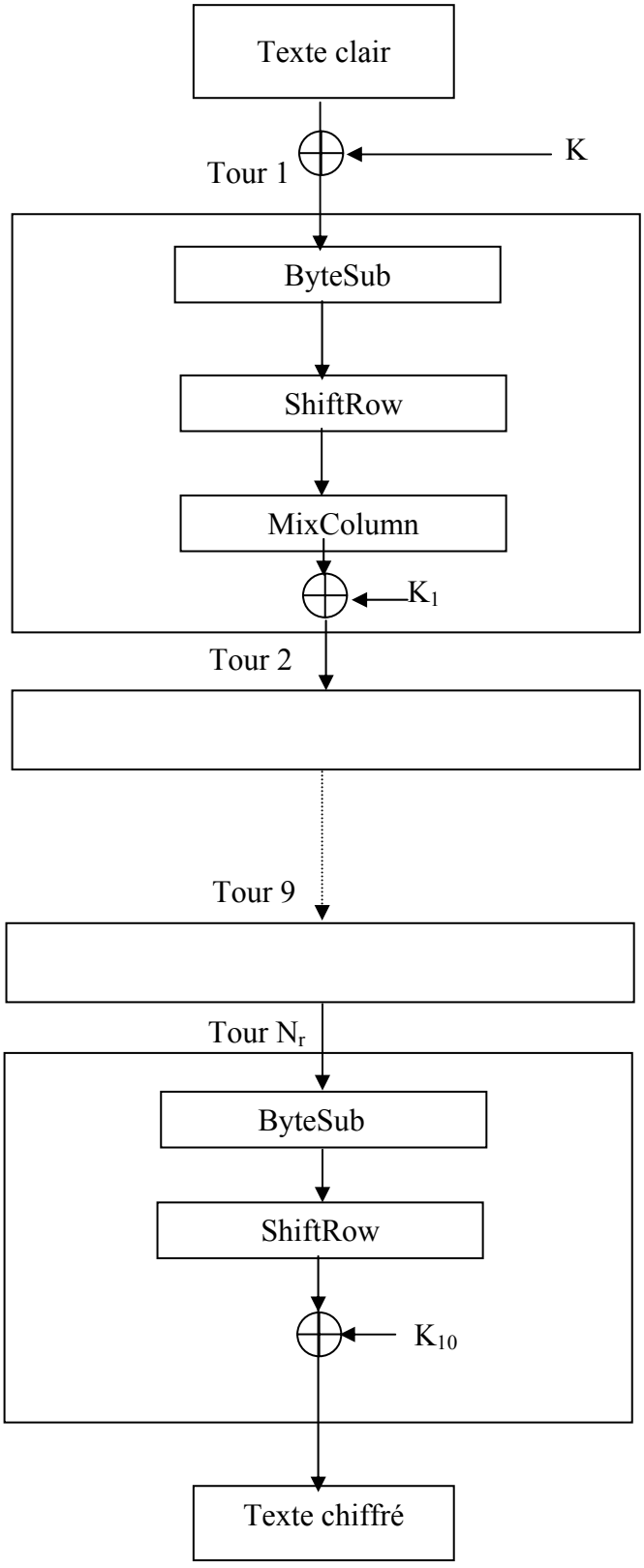


Figure 2.8 : Structure globale de Rijndael

2.4.12. Sécurité de Rijndael

Contre Rijndael, on ne connaît que l'attaque exhaustive de nos jours, car même pour une petite longueur de clé (128 bits) aucun ordinateur au monde n'aura une puissance de calcul suffisante avant 70 ans pour trouver la bonne clé. Si le nombre de tours est inférieur à 10, des attaques sont possibles : il en existe une pour sept tours.

Remarque 2-1

Dans les deux exemples suivants, on représente tous les messages en hexadécimal.

Exemple 2-6 (de la Clé étendue)

Soit la clé secrète k_0 (clé de chiffrement) de 128 bits ($N_k = 4$) suivante :

k_0 = 2 b 7 e 15 16 28 ae d2 a6 ab f7 15 88 09 cf af 3c (en hexadécimal)

W_0 = 2b7e1516 W_1 = 28aed2a6 W_2 = abf71588 W_3 = 09cf4f3c

Le tour i	W[i-1]	Après Rot word	Après Sub word	R Con [i]	Après XOR Avec R Con [i]	W [I - N_k]	W[i] = W[i-1] XOR W [I- N_k]
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					575c006a	b6630ca6

La table précédente nous donne les dix clés partielles suivantes:

k_1 = a0fafe17 88542cb1 23a33939 2a6c7605

k_2 = f2c295f2 7a96b943 5935807a 7359f67f

k_3 = 3d80477d 4716fe3e 1e237e44 6d7a883b

k_4 = ef44a541 a8525b7f b671253b db0bad00

k_5 = d4d1c6f8 7c839d87 caf2b8bc 11f915bc

k_6 = 6d88a37a 110b3afd dbf98641 ca0093fd

k_7 = 4e54f70e 5f5fc9f3 84a64fb2 4ea6dc4f

k_8 = ead27321 b58dbad2 312bf560 7f8d292f

k_9 = ac7766f3 19fadc21 28d12941 575c006e

k_{10}= d014f918 c9ee2589 e13f0cc8 b6630ca6

Exemple 2-7 (du Chiffrement Rijndael)

Soient le texte clair de 128 bits (N_k = 4) et la clé secrète k_0 de 128 bits (N_k = 4).

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

k_0 = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

Dans cet exemple, on utilise les clés partielles de l'exemple (1) , car on a utilisé la même clé secrète.

		Après SubBytes	Après Shiftrows	Après MixColum		Les clés Partielles	
Etat Initial (le texte clair)	32 88 31 a0 43 5a 31 37 f6 30 98 07 a8 8d a2 34				⊕	2b 28 ab 09 7e ac f7 cf 15 d2 15 4f 16 a6 88 3c	=
Etat 1	19 a0 9a a9 3d f4 c6 f8 a3 a2 8d 48 ba 2b 2a 08	d4 e0 b8 1e 27 bf b4 41 11 96 5d 52 ae f1 e5 30	d4 e0 b8 1e bf b4 41 27 5d 52 11 98 30 ae f1 a5	04 e0 48 28 66 cb f8 06 81 19 d3 26 e5 9a 7a 4c	⊕	a0 88 23 2a fa 54 a3 6c fe 2c 39 05 17 b1 39 05	=
Etat 2	a4 68 6b 02 9c 9f 5b 6a 7f 35 ea 50 f2 2b 43 49	49 45 7f 77 de db 39 02 d2 96 87 53 89 f1 1a 3b	49 45 7f 77 db 39 02 de 87 53 d2 96 3b 89 f1 1a	58 1b db 1b 4d 4b e7 6b ca 5a ca b0 f1 ac a8 a5	⊕	f2 7a 59 73 c2 96 35 59 95 b9 80 f6 f2 43 7a 7f	=
Etat 3	aa 61 82 68 8f dd d2 32 5f e3 4a 46 03 ef d2 9a	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a 7b df b5 b8	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	75 20 53 bb ec 0b c0 25 09 63 cf d0 93 33 7c dc	⊕	3d 47 1e 6d 80 16 23 7a 47 fe 7e 88 7d 3e 44 3b	=
Etat 4	48 67 4d d6 6c 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	52 85 e3 f6 50 a4 11 cf 2f 5e c8 6a 28 d7 07 94	52 85 e3 f6 a4 11 cf 50 c8 6a 2f 5e 94 28 d7 07	0f 60 6f 5e d6 31 c0 b3 da 38 10 13 a9 bf 6b 01	⊕	ef a8 b6 db 44 52 71 0b a5 5b 25 ad 41 7f 3b 00	=
Etat 5	e0 c8 d9 85 92 63 b1 b8 7f 63 35 be e8 c0 50 01	e1 e8 35 97 4f fb c8 6c d2 fb 96 ae 9b ba 53 7c	e1 e8 35 97 fb c8 6c 4f 96 ae d2 fb 7c 9b ba 53	25 bd b6 4c d1 11 3a 4c a9 d1 33 c0 ad 68 8e b0	⊕	d4 7c ca 11 d1 83 f2 f9 c6 9d b8 15 f8 87 bc bc	=

Etat 6

f1	c1	7c	5d
00	92	c8	b5
6f	4c	8b	d5
55	ef	32	0c

a1	78	10	4c
63	4f	e8	d5
a8	29	3d	03
fe	df	23	fe

a1	78	10	4c
4f	e8	d5	63
3d	03	a8	29
fa	fc	df	23

4b	2c	33	4b
86	4a	9d	d2
8d	89	f4	18
6d	80	e8	d8

⊕

4e	5f	84	4e
88	0b	f9	00
a3	3e	86	93
7a	fd	41	fd

=

Etat 7

26	3d	e8	fd
0e	41	64	d2
2e	b7	72	8b
17	7d	a9	25

f7	27	9b	54
ab	83	43	b5
31	a9	40	3d
f0	ff	d3	3f

f7	27	9b	54
83	43	b5	ab
40	3d	31	a9
3f	f0	ff	d3

14	46	27	34
15	16	46	2a
b5	15	56	d8
bf	ec	d7	43

⊕

4e	5f	84	4e
54	5f	a6	a6
f7	c9	4f	dc
0e	f3	b2	4f

=

Etat 8

5a	19	a3	7a
41	49	e0	8c
42	dc	19	04
b1	1f	65	0c

be	d4	0a	da
83	3b	e1	64
2c	86	d4	f2
c8	c0	4d	fe

be	d4	0a	da
3b	e1	64	83
d4	f2	2c	86
fe	c8	c0	4d

00	b1	54	fa
51	c8	76	1b
2f	89	6d	99
d1	ff	cd	ea

⊕

ea	b5	31	7f
d2	8d	2b	8d
73	ba	f5	29
21	d2	60	2f

=

Etat 9

ea	04	65	85
83	45	5d	96
5c	33	98	b0
f0	2d	ad	c5

87	f2	4d	97
ec	6e	4c	90
4a	c3	46	e7
8c	d8	95	a6

87	f2	4d	97
6e	4c	90	ec
46	e7	4a	c3
a6	8c	d8	95

47	40	a3	4c
37	d4	70	9f
94	e4	3a	42
ed	a5	a6	bc

⊕

ac	19	28	57
77	fa	d1	5c
66	dc	29	00
f3	21	41	6e

=

Etat 10

eb	59	8b	1b
40	2e	a1	c3
f2	38	13	42
1e	84	e7	d2

e9	cb	3d	af
09	31	32	2e
89	07	7d	2c
72	5f	94	b5

e9	cb	3d	af
31	32	2e	09
7d	2c	89	07
b5	72	5f	94

⊕

d0	c9	e1	b6
14	ee	3f	63
f9	25	0c	0c
a8	89	c8	a6

=

Etat finale (Le texte chiffré)

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

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

2.5 Conclusion

Dans les systèmes cryptographiques à clés secrètes la fonction de déchiffrement se déduit facilement de la fonction de chiffrement et avant toute utilisation d'un tel système, le couple (émetteur, destinateur) doit se mettre d'accord sur une clé, ce qui pose le problème de distribution des clés.

Pour la résolution de ce problème les chercheurs ont crée la cryptographie à clé publique qu'on va étudier dans le chapitre 3.

3.1 Introduction

Le concept de cryptographie à clé publique fut introduit en 1976 par Witfield Diffie et Martin Helman. Son principe fondamental est le choix pour chaque utilisateur de deux clés associées, l'une est publique utilisée pour le chiffrement, et l'autre est secrète utilisée pour le déchiffrement.

La puissance de ce procédé consiste dans l'impossibilité de déduire la clé secrète en connaissant la clé publique et la non nécessité de créer autant de clés qu'il y a de communications possibles, donc un nouvel arrivant n'a qu'une paire de clés à générer.

3.2 Le cryptosystème RSA (cf [13]).

Le premier système cryptographique à clé publique est le système RSA, qui a été crée en 1978 par Ron Rivest, Adi Shamir et Leonard Adleman. Ce système est basé sur la factorisation d'entiers en produit de deux grands facteurs premiers.

3.2.1 Description de RSA

Le destinataire crée une clé publique qu'il diffuse à ses correspondants et une clé secrète. Pour cela , il choisit au hasard deux grands nombres premiers distincts p et q . Leur produit n a pour indicatrice d'Euler $\phi(n) = (p-1)(q-1)$. Il choisit un entier e tel que : $0 \langle e \langle n$ et e premier à $\phi(n)$.

D'autre part, il calcule l'entier d tel que : $0 \langle d \langle n$ et $ed \equiv 1 \bmod \phi(n)$.

Il diffuse les entiers n et e , tout en gardant secrètes p, q et d .

Lorsque l'émetteur veut envoyer un message confidentiel au destinataire :

1. Il représente le message par un nombre m compris entre 0 et $n-1$, si besoin il découpe auparavant le message en blocs.

2. Il doit s'assurer que la clé publique (n, e) est bien celle de son destinataire.

3. Il calcule $c = m^e \bmod n$ qui est le texte chiffré.

4. Il transmet c au destinataire.

Lorsque le destinataire reçoit c il calcule le texte clair en utilisant sa clé secrète d

$m = c^d \bmod n.$

3.2.2 Fonctionnement de RSA

Pour le fonctionnement de RSA on a besoin du théorème suivant :

Théorème3-1 (petit théorème de Fermat)

Soit p un nombre premier et a un entier naturel premier à p ; alors

$$a^{p-1} \equiv 1 \bmod p .$$

Comme $ed \equiv 1 \bmod \phi(n)$, alors il existe un entier k tel que $ed = 1 + k\phi(n)$.

Si m est premier avec p, d'après le petit théorème de Fermat $m^{p-1} \equiv 1 \bmod p$.

On élève les deux membres à la puissance $k(q-1)$ on obtient :

$$m^{k(p-1)(q-1)} \equiv 1 \bmod p .$$

En multipliant les deux membres par m on obtient :

$$m^{ed} \equiv m^{1+k(p-1)(q-1)} \equiv m \bmod p \qquad (1)$$

Si m n'est pas premier avec p alors m est un multiple de p et la congruence précédente reste encore vraie car les deux membres sont congrus à $0 \bmod p$.

On montre de même que : $m^{ed} \equiv m \bmod q$ (2)

Les deux formules (1) et (2) et l'hypothèse p et q sont premiers et distincts impliquent :

$$m^{ed} \equiv m \bmod n .$$

Exemple 3-1

Supposons

$p = 954210358745120365879874110236587410236598701246 07$

$q = 136589752102356987452814569874512035689741209851202 5699$

Alors :

$n = pq = 130335356354497119703101241277213156185113188180675435947778999387 3$

$3690539151737629712250983428711627529 3$

et

$\phi(n) = (p-1)(q-1) = 130335356354497119703101241277213156185113188180674069954836 9399429$

$503406578312201531068713985287341249 88$

Choisissons e tel que : $0 \langle e \langle n-1$ et e premier à $\phi(n)$.

En prenant

$e =$ 27317950420471397490562913974902407137948241970240515 57

Donc :

$d = e^{-1} \bmod \phi(n) =$ 41590897809510320212183021601282393207526331671542450354632851073
3812525830
13019306393802514666505625309

Le message clair

$m =$ 501235478541201245230213434356786786454121010124245454546587899785 4
542000000001212424245455787554 57

Le message chiffré

$c = m^e \bmod n =$ 67702675284986641463728580254014344887659143535685651
00803445376359427562708815689976360831871420223423 8

Le message déchiffré

$m = c^d \bmod n =$ 501235478541201245230213434356786786454121010124245454546587899785 4
542000000001212424245455787554 57

3.2.3 Sécurité de RSA

La sécurité du cryptosystème RSA repose sur le problème de factorisation des grands nombres premiers, en effet, si on arrive à factoriser n on peut calculer $\phi(n) = (p-1)(q-1)$ et calculer l'exposant de déchiffrement d en utilisant l'algorithme d'Euclide étendue et ensuite découvrire le message clair.

Il est possible d'attaquer RSA en essayant de deviner la valeur de $\phi(n)$, cette attaque n'est pas plus facile que la précédente comme le montre le théorème suivant :

Théorème 3-2

La connaissance de $\phi(n)$ est équivalente à la connaissance de la factorisation de n.

Démonstration :

Si la factorisation $n = pq$ est connue alors $\phi(n)$ est calculable, (il suffit de remplacer la valeur de p et q dans $\phi(n) = (p-1)(q-1)$.

Réciproquement supposons que $\phi(n)$ soit connu. En substituant n/p à q dans la relation : $\phi(n)=(p-1)(q-1)$.

On obtient l'équation du second degré :

$$p^2 - (n + 1 - \phi(n))\,p + n = 0 .$$

Les solutions de cette équation sont p et q.

Il existe d'autres possibilités d'attaques contre RSA comme l'attaque à module commun :

En effet, la diffusion d'un même message m à deux utilisateurs en employant le même module n et deux exposant de chiffrement premiers entre eux sert à trouver le message clair. On a :

$$C_1 = m^{e_1} \quad \text{et} \quad C_2 = m^{e_2}.$$

Si e_1 et e_2 sont premiers entre eux on peut trouver m sans connaître les clés secrètes d_1 ,d_2 et sans factoriser n ; on calcule par l'algorithme d'Euclide étendue, u et v tels que :

$ue_1 + ve_2 = 1$ et on trouve $C_1^u C_2^v \equiv m \bmod n.$

3.3 Le cryptosystème de Diffe-Hellman (cf [13]).

Ce système est inventé par Diffe-Hellman il permet à deux personnes qui n'ont jamais communiqué ensemble auparavant d'engendrer une clé secrète en utilisant le logarithme discret.

3.3.1 Description de Diffe-Hellman

L'émetteur et le destinataire se mettent d'accord sur un nombre premier p et un élément g de IF_P^* qu'ils rendent publiques, ensuite :

1) L'émetteur choisit un nombre $x \in IF_q$ et le destinataire un nombre $y \in IF_q$

les nombres x et y sont respectivement les clés secrètes de l'émetteur et du destinataire .

2) L'émetteur envoie $X = g^x \bmod p$ au destinataire et le destinataire envoie

$Y = g^y \bmod p$ à l'émetteur.

3) L'émetteur calcule la clé $K = Y^x \bmod p$ et le destinataire calcule

$K' = X^y \bmod p = X^y \bmod p.$

3.3.2 Fonctionnement de Diffe-Hellman

L'émetteur et le destinataire ont bien la même clé car :

$$K = Y^x \bmod p = (g^y)^x \bmod p = g^{xy} \bmod p = (g^x)^y \bmod p = X^y \bmod p = K'.$$

3.4 Le cryptosystème ElGamal (cf [13]).

Le cryptosystème ElGamal est publié en 1985. Il peut être utilisé à la fois par les systèmes numériques et pour le chiffrement..

Le cryptosystème ElGamal est non déterministe car l'opération de chiffrement dépend du message clair m et d'une valeur aléatoire k choisie par l'émetteur, il y a donc plusieurs textes chiffrés qui correspondant à un même message clair.

.

Définition 3-1

Soit (G,.) un groupe noté multiplicativement, a et b deux éléments de G, un logarithme discret de b dans la base a est un élément $x \in IN$ tel que $a^x = b$.

3.4.1 Le problème du logarithme discret

Soient p un grand nombre premier et g une racine primitive modulo p , le problème du logarithme discret est celui de trouver, étant donné $A \in \boldsymbol{Z}$ tel que p ne divise pas A , l'entier a vérifiant :

$$g^a \equiv A \bmod p$$

3.4.2 Description d'ELGamal

Le destinataire choisit au hasard un grand nombre premier p et un générateur α du groupe multiplicatif $\mathbf{Z}_p^*$. Il choisit ensuite au hasard un entier a compris entre 1 et $p-2$ et il calcule $A = \alpha^a \bmod p$.

Il diffuse le triplet (p, α, A) qui est sa clé publique et garde a , sa clé secrète.

Lorsque l'émetteur veut envoyer un message confidentiel au destinataire :

1) Il représente le message par un entier m compris entre 0 et $p-1$.

2) Il doit s'assurer que la clé publique (p, α, A) est bien celle de son destinataire.

3) Il choisit au hasard un entier k tel que : $1 \le k \le p-2$ et calcule $y = \alpha^k \bmod p$ et $\delta = mA^k \bmod p$.

4) Il envoie le texte chiffré (y, δ) à son destinataire.

Lorsque le destinataire reçoit le couple (y, δ) il utilise sa clé secrète a pour calculer le texte clair $y^{p-1-a}\delta \bmod p$.

3.4.3 Fonctionnement d'ElGamal

L'ordre du groupe multiplicatif $\mathbf{Z}_p^*$ est égal à $p-1$ donc :

$$y^{p-1} \equiv 1 \bmod p$$

et $$y^{p-1-a} \equiv (\alpha^k)^{-a} m(\alpha^a)^k \equiv m \bmod p.$$

3.4.4 Sécurité d'AlGamal

La sécurité du cryptosystème AlGamal repose sur le problème du logarithme discret, actuellement on sait calculer le logarithmes avec un module de 300 bits environ, pour une sécurité durable il faut choisir un module p de 1024 bits au moins et il faut que $p-1$ ait un grand facteur premier.

Exemple 3-2

Supposons

$p = 952103254897901$

$\alpha = 2$

$a = 8564521358745$

$A = \alpha^a \bmod p = 494531055194754.$

Si l'émetteur veut envoyer le message $m = 2013652108745$ à son destinataire, il choisit au hasard l'entier k, disons : $k = 7452102584$ et calcule $y = \alpha^k \bmod p = 891609355258005$

Puis $\delta = mA^k \bmod p = 690053504698822$

Lorsque le destinataire reçoit le texte chiffré (y, δ), il calcule

$y^{p-1-a} \bmod p = 2013652108745$ qui est bien le texte clair.

3.5 Cryptosystème de Messay-Omura (cf [17]).

Ce système à été inventé par Messay et Omura. Il est basé sur une idée de Shamir ; avec ce système l'émetteur et le destinataire n'ont pas besoin d'utiliser des clés publiques. Ils s'arrangent pour communiquer de la manière suivante :

3.5.1 Description de Messay-Omura

Lorsque l'émetteur veut envoyer un message confidentiel au destinataire :

1) L'émetteur choisit au hasard une clé de chiffrement e_A qui est premier à $p-1$, et le destinataire choisit au *hasard* une clé de chiffrement e_B qui est premier à $p-1$.

2) L'émetteur calcule $d_A \equiv e_A^{-1} \bmod p-1$ et le destinataire calcule $d_A \equiv e_B^{-1} \bmod p-1$

3) L'émetteur code le message clair en un nombre M compris entre 1 et p et envoie la *réduction* $M^{e_A} \bmod p$ au destinataire.

4) Le destinataire renvoie $(M^{e_A})^{e_B} \equiv M^{e_A e_B} \bmod p$ à l'émetteur.

5) L'émetteur envoie $(M^{e_A e_B})^{d_A} \equiv M^{e_B} \bmod p$ au destinataire.

6) Le destinataire calcule $(M^{e_B})^{d_B}$.

3.5.2 Fonctionnement de Messay-Omura

Le destinataire a bien reçu le message clair $(M^{e_B})^{d_B} \equiv M \bmod p$.

3.5.3 Sécurité de Messay-Omura

L'avantage de ce système est qu'il n'a besoin ni d'annuaire, ni de nouvelles clés à chaque fois ; avec ELGamal il est possible d'arriver au calcul de la clé privée dans un laps de temps assez long alors qu'avec ce système il faut moins de temps.

Le désavantage de ce système est qu'il demande trois transmissions, ce qui entraîne un risque du point de vue sécurité du message.

Définition 3-2

Soit K un corps commutatif, une courbe elliptique sur K est une cubique irréductible non singulière d'équation particulière (dite de Weiestrass).

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6. \qquad (1)$$

Où les coefficients a_i sont des éléments du corps K et les racines (x, y) de l'équation (1) sont dans K_{alg}^2 (K_{alg} étant la clôture algébrique du corps K).

3.6 Cryptosystèmes utilisant les courbes elliptiques (cf [8]).

Les courbes elliptiques permettent d'implémenter des systèmes cryptographiques semblables à celui proposé par ElGamal.

La sécurité de ces systèmes repose sur la difficulté de calculer un logarithme discret, mais contrairement à IF_q , il n'existe aucun algorithme sous-exponentiel pour résoudre le problème du logarithme discret dans le groupe $E(IF_q)$ des points de la courbe elliptique E définie sur le corps fini IF_q.

Cela permet, pour une sécurité équivalente, d'utiliser des clés plus petites. Par ailleurs, sur un corps choisi, il est possible de construire une multitude de courbes elliptiques. En particulier dans le groupe $E(IF_q)$ les calculs se font très rapidement. Les avantages des courbes elliptiques sont donc multiples.

3.6.1 Le problème du logarithme discret d'une courbe elliptique

Soit E une courbe elliptique définie sur un corps fini IF_q et P, Q deux points de la courbe $E(IF_q)$.

Le problème du logarithme discret sur $E(IF_q)$ est de trouver l'entier rationnel a tel que $P = aQ$

3.6.2 Cryptosystème de Diffie Hellman-elliptique :

Ce cryptosystème est identique au système de Diffie-Hellman sauf qu'il est basé sur le problème du logarithme discret d'une courbe elliptique.

L'émetteur et le destinataire rendent publique une courbe elliptique E définie sur IF_q et un point p de $E(IF_q)$.

1) l'émetteur choisit un nombre $x \in IF_q^*$ et le destinataire un nombre $y \in IF_q^*$.

2) l'émetteur envoie $X = xp$ au destinataire et le destinataire envoie $Y = yp$ à l'émetteur.

3) l'émetteur calcule la clé $K = xy$ et le destinataire calcule la clé $K' = yX$

3.6.3 Fonctionnement de Diffie Hellman - elliptique

L'émetteur et le destinataire ont bien la même clé :

$K = xY = x(yp) = xyp = y(xp) = yX = K'$.

Exemple 3-3

Soit $p = 211,\ E : y^2 = x^3 - 4$ et $G = (2,2) \in E(IF_{211})$.

p , E et G sont publiques.

L'émetteur choisit une clé privée $a_A = 121$ et envoie $a_A G = 121(2,2) = (115,48)$ et le destinataire choisit une clé privée $a_B = 203$ et envoie $a_B G = 203(2,2) = (130,203)$.

Ensuite, l'émetteur calcule $a_A(a_B G) = 121(130,203) = (161,69)$. Le destinataire calcule $a_B(a_A G) = 203(115,48) = (161,69)$.

3.6.4 Cryptosystéme d'ElGamal elliptique

Ce système est identique au système d'ElGamal sauf qu'il est basé sur le problème du logarithme discret d'une courbe elliptique.

Il n'existe aucune méthode déterministe pour représenter un message néanmoins, il est possible de fixer la probabilité d'échec.

Pour fixer les idées Notons :

$E : y^2 = x^3 + ax + b \cup O$ la courbe elliptique sur IF_q . Soit k un entier positif tel que la probabilité qu'on ne puisse pas représenter le message m est 2^{-k}

Supposons que :

$$m \langle M \quad \text{et} \quad q \rangle Mk .$$

On représente le message m comme un élément de IF_q par : $x' = mk + j \quad 1 \le j \le k.$

Remarquons que :

$$x' \le (M-1)k + k = Mk < q .$$

On choisit alors y' tel que : $y' = x'^3 + ax' + b$ soit un carré en essayant $j = 1,2,...,k$ pour retrouver la partie entière de $\frac{x'-1}{k}$ qu'on note par $\left\lfloor \frac{x'-1}{k} \right\rfloor = m$.

Comme il y a approximativement une chance sur deux pour que y' soit un carré, la probabilité d'échec est de 2^{-k}.

3.6.5 Description d'AlGamal elliptique

Si l'émetteur veut envoyer un message au destinateur alors :

1) L'émetteur choisit une courbe elliptique E définie sur un corps fini IF_q Il représente son message m comme un point Q de la courbe $E(IF_q)$. Il choisit un entier k et un point G de la courbe $E(IF_q)$ et p au destinataire.

2) Le destinataire choisit un entier a_B et envoie $a_B G$ à l'émetteur.

3) L'émetteur calcule kG et $Q + ka_B G$ et envoie $(kG, Q + ka_B G)$ au destinataire.

4) Le destinateur calcule $-a_B kG$, puis la somme $(Q + ka_B G) + (-a_B kG)$.

3.6.6 Fonctionnement d'AlGamal elliptique

$$(Q + ka_B G) + (-a_B kG) = Q + ((ka_B G) + (-a_B G)) = Q$$

Le destinataire trouve bien le point Q qui correspond au message clair.

.

3.7 Cryptographie et courbes hyperelliptiques (cf [22])

Définition 3-3

Une courbe hyperelliptique sur un corps de caractéristique différente de 2 est une courbe qui admet une équation de la forme :

$$y^2 = f(x)$$

où $f(x)$ est un polynôme de degré $2g+1$. Les racines de $f(x)$ sont distincts. .L'entier $g \geq 1$ est appelé le genre de la courbe.

Il n'y a pas de loi de groupe définie directement sur l'ensemble des points des courbes hyperelliptiques .C'est pourquoi on étudie la jacobienne de la courbe qui, elle, peut être munie d'une loi d'addition.

On va se restreindre au cas où le genre est 2.

Toutes les définitions peuvent se généraliser au genre supérieur.

La jacobienne d'une courbe C de genre 2 est la réunion de l'ensemble suivant :

1) L'élément neutre que l'on note O, et qui est le point à l'infini de la courbe.

2) L'ensemble des points de la courbe.

3) L'ensemble des paires de points de la courbe.

Un élément de la jacobienne est appelé un diviseur. Il est commode de noter $D = P_1+P_2$ un diviseur de type 3. En effet, si P_1, P_2 sont deux points de le courbe considérés comme des diviseurs, leur somme sera précisément le diviseur du type 3formé par ces deux points.

Pour faire la somme de diviseurs $D_1 = P_1+ P_2$ et $D_2 = Q_1+ Q_2$. On trace la courbe $y^2 = f(x)$ passant par les quatre points P_1, P_2, Q_1, Q_2, où *f(x)* est un polynôme de degré 3. Cette courbe coupe *C* en deux autres points R_1 , R_2 et on prend leur symétriques S_1 ,S_2. on a alors $D_1 +D_2= D_3 =R_1 +R_1$.

La connaissance du cardinal exact d'un groupe est nécessaire si l'on veut construire un cryptosystème. En effet, afin de garantir une bonne sécurité, le cardinal doit être premier. Il existe un algorithme théorique dû à Pila qui effectue cette tâche en temps polynomial. Cet algorithme s'inspire de l'algorithme de Schoof pour les courbes elliptiques.

En 1994, Adleman, Demaviers et Huang ont présenté un algorithme résolvant le problème de logarithme discret pour une courbe hyperelliptique de genre « grand » la complexité de cet algorithme est heuristiquement sous- exponentielle.

Un premier algorithme sous- exponentielle prouvé à été donné par Enge, les courbe hyperellitique de genre supérieure à 4 étaient moins sûr que les autres courbes.

3.8 Combinaison de la cryptographie à clé publique et à clé secrète

En pratique, les algorithmes à clé publique ne se substituent pas aux algorithmes à clé secrète. On les utilise pour chiffrer des clés et non des messages par ce que :

1) Les algorithmes à clé publique sont lents et les algorithmes à clé secrètes sont généralement au moins 1000 fois plus rapide que les algorithmes à clé publique.

2) Les cryptosystèmes à clé publique sont vulnérables aux attaques à texte clair choisi, alors que les cryptosystèmes à clé secrètes ne présentent pas ce défaut. Un cryptanalyste ne peut pas faire des déchiffrements avec une clé inconnue.

Dans la plupart des applications pratiques, la cryptographie à clé publique est utilisée pour protéger et distribuer les clés et ces clés sont utilisées dans des algorithmes à clé secrète pour protéger les messages transmis. Cela est appelé un cryptosystème hybride.

3.9 Conclusion

La cryptographie a clé publique résout le problème de la mise en accord de clés. Ceci est très important lorsque l'on ne connaît pas son correspondant, par exemple, pour transmettre un numéro de carte de crédit à un serveur commercial sur Internet.

Deux problèmes se posent cependant :

1) D'un point de vue pratique, tout d'abord, la cryptographie à clé publique nécessite des calculs bien plus complexes que ceux requis par la cryptographie à clé unique. On peut contourner ce problème en ne l'utilisant que pour se mettre d'accord sur une clé symétrique qui sera ensuite utilisée pour chiffrer la communication de manière conventionnelle.

2) Un second problème, bien plus difficile à résoudre, est celui de la certification des clés publiques. En effet, il est très facile pour quiconque de générer des couples de clés publiques et secrétés associées. Comment peut-on alors s'assurer que la clé publique que l'on utilise pour chiffrer un message à l'intention d'un correspondant lui appartient effectivement et n'est pas celle d'un pirate ?Une solution consiste à faire signer la clé publique par une autorité certifiant son appartenance à un individu. Ceci nécessite par conséquent, de disposer d'un équivalent numérique à la signature manuscrite qu'on va étudier dans le chapitre 4.

4.1 Introduction

Un procédé de signature permet de signer un document sous une forme électronique. Ainsi, une signature peut être transmise par un réseau informatique.

Dans ce chapitre, on étudie plusieurs procédés de signature. On discute tout d'abord des différences entre une signature manuscrite et une signature électronique.

Un premier problème est la notion de signature d'un document. Une signature conventionnelle est physiquement attachée au document signé. Mais une signature électronique ne peut pas l'être de la même manière. Ainsi, le procédé de signature doit être en quelque sorte « coller » la signature au message.

Un second problème est celui de la vérification .Une signature conventionnelle est authentifiée par comparaison à une autre qui a été certifiée.

Un procédé de signature est composé d'un algorithme de vérification.

Pour tout couple (x, y), l'algorithme de vérification fournit une réponse « vrai » ou « faux » suivant que y est une authentique signature de x ou non.

Définition 4-1

Un procédé de signature est un quintuplé (P, S, K, A, γ) vérifiant :

1) P est un ensemble fini de messages.

2) S est un ensemble fini de signatures.

3) K est un ensemble fini de clés.

4) A est un ensemble de signatures.

5) γ est un ensemble de vérifications.

Pour chaque $k \in$ K, il y a une fonction de signature $sig_k \in$ A et une fonction de vérification $ver_k \in$ γ correspondante.

Les fonctions $sig_k : P \longrightarrow S$ et $ver : P \times S \longrightarrow \{vraie, faux\}$

Vérifient chaque message $m \in P$ et chaque signature $s \in S$

$$Ver\ (m, s) = \begin{cases} vrai & si: \quad S = sig_k(m) \\ faux & si: \quad S \neq sig_k(m) \end{cases}$$

Remarque

Pour chaque $k \in$ K, les fonctions sig_k et ver_k doivent être calculables en temps polynomial.

Le premier exemple de procédé de signature est le système à clé publique RSA en mode de signature.

4.2 Signature RSA (cf [18]).

Le cryptosystéme à clé publique RSA peut être utilisé comme procédé de signature, d'ailleurs on peut très bien se servir de la même clé secrète pour déchiffrer des messages reçus que pour envoyer des messages signés.

Le procédé de signature RSA est représenté comme suit :

Soit $n = pq$, où p et q sont premiers.

Soit P = A =$\mathbf{Z}_n$, et K= $\{(n, p, q, a, b) : n = pq, p, q \text{ premiers}, ab \equiv 1 \bmod \phi(n)\}$.

n et b sont publiques, et p, q et a sont secrets.

Pour $(n,p,q,a,b) \in$K, on définit la fonction de signature sig_k et la fonction de vérification ver_k correspondante par :

$$\text{et} \qquad \begin{aligned} sig_k(x) &= x^a \bmod n \\ ver_k(x,y) &= vrai \Leftrightarrow x \equiv y^b \bmod n \end{aligned}$$

où $x, y \in \mathbf{Z}$

4.3 Signature ElGamal (cf [18]).

Le procédé de signature ElGamal a été spécialement conçu pour réaliser des signatures électroniques contrairement à RSA, qui sert, à la fois, de système cryptographique à clé publique et de procédé de signature

Le procédé de signature ElGamal n'est pas déterministe, tout comme le cryptosystème ElGamal. Cela implique que pour un message donné, il existe plusieurs signatures valables.

Ce procédé est représenté ainsi :

Soit un nombre premier p tel que le problème du logarithme discret dans $\mathbf{Z}_p$ soit difficile, et soit $\alpha \in \mathbf{Z}_p^*$ une racine primitive. Soit $\wp \in \mathbf{Z}_p^*$, $\mathbf{Z}_p^* \times \mathbf{Z}_p^*$ et

$$\text{K}=\{(p,\alpha,a,\beta) : \beta = \alpha^a \bmod p\}.$$

Les valeurs p, α et β sont publiques, et α est secret.

Pour $\text{K} = (p,\alpha,a,\beta)$ et pour un $k \in \mathbf{Z}_{p-1}^*$ (secret), on définit $sig_{\text{K}}(x,k) = (y,\delta)$ où :

et
$$y = \alpha^k \bmod p$$
$$\delta = (x - ay)k^{-1} \bmod (p-1)$$

Pour $x, y \in \mathbf{Z}^*_{p-1}$ et $\delta \in \mathbf{Z}_{p-1}$, on définit :

$$ver(x,y,\delta) = vrai \Leftrightarrow \beta^y y^\delta \equiv \alpha^x \bmod p.$$

Exemple 4-1

Supposons que

$p = 86541202314587451253$

$\alpha = 6$

$a = 5120344877$

On a :

$\beta = \alpha^a \bmod p = 34647667113535234844.$

Pour signer le message $x = 30214568795$ on choisit k premier à $p-1$.

Prenons $k = 12035421091.$

Donc : $k^{-1} \bmod(p-1) = 71195794235844666559.$

On a :

$y = \alpha^k \bmod p = 75559695747745449406$

et

$\delta = (x-ay)k^{-1} \bmod(p-1) = 53173612077282351967$

N'importe qui peut authentifier la signature en vérifiant :

$\beta^y y^\delta \bmod p = 84283814313951853737$

$a^x \bmod p = 84283814313951853737.$

La signature est donc valide.

4.4 Signatures et Fonctions de hachage

Le procédés de signature ne permettant de signer que des petits messages (128 ou160 bits). Que se passera-t-il lorsque on veut signer des messages beaucoup plus longs ? On peut découper le message en blocs, et signer chacun d'entre eux, cette approche est très lente, et elle ne garantit pas contre un ré-arrangement des blocs qui peut changer la signification du message.

Une solution consiste à compresser le message en utilisant une fonction de hachage.

Cette fonction doit être rapide à calculer, transforme un message de longueur arbitraire en une empreinte numérique de longueur fixe. Ensuite on signe l'empreinte pour authentifier le message.

Définition 4-2

Une fonction de hachage est une fonction H d'un ensemble infini (ensemble des messages) dans un ensemble fini (ensemble des empreintes) tel que H soit facilement calculable.

Définition 4-3

Une fonction de hachage H est à collisions faibles difficiles si, étant donné un message x, il est difficile d'obtenir un message $x' \neq x$ tel que $H(x) = H(x')$.

Définitions 4-4

Une fonction de hachage H est à collisions fortes difficiles s'il est difficile d'obtenir deux messages difficile d'obtenir deux messages différent x et x' tels que $\mathrm{H}(x') = \mathrm{H}(x)$.

Définitions 4-5

Une fonction de hachage H est à sens unique si, étant donné une empreinte numérique z, il est difficile de trouver un message x tel que $\mathrm{H}(x) = z$.

4.5 La fonction de hachage SHA (cf [2]).

4.5.1 Description de La fonction SHA

L'institut National des Standards et de la Technologie Américain ainsi que l'Agence Nationale de Sécurité Américaine ont conçu l'algorithme sûr de hachage SHA « Secure Hash Algorithme ».

Le SHA été expérimenté en 1993. Il est qualifié de sûr parce qu'il est impossible de retrouver deux messages différent qui ont la même empreinte.

4.5.2 Description de la fonction de hachage

La fonction de hachage SHA considère les messages comme des blocs de 512 bits. Elle fournit un haché de 160 bits et utilise une fonction de compression :

$$H : (GF(2)^{32})^5 \times (GF(2)^{512} \rightarrow (GF(2)^{32})^5$$

$$(AA, BB, CC, EE, m) \rightarrow H(AA,BB,CC,EE,m) = (AA',BB',CC,EE')$$

Pour hacher un message de longueur quelconque on le considère comme une chaîne de caractère codes sur des octets (8 bits) et on effectue la séquence suivante :

1) Ajouter à la fin du message à hacher un octet valant 1, suivi d'un nombre N de 0 et d'un entier codé sur 64 représentant la longueur initiale en octets du message. Le nombre N est le nombre minimal d'octets permettant à l'issu du processus d'avoir un message (y compris la taille rajoutée à la fin) codé sur un nombre entier *n* de blocs de 512 bits ($m_1,\ldots,m_n$).

2) Initialiser 5 registres de 32 bits notés *A.B.C.D* et *E* avec les constantes :

A=0x67452301

B=0xEFCDAB89

C=0x98BADCFE

D=0x10325476

E= 0Xc3d2e1f0

Tours i	Fonction $f^{(i)}$		Constant $K^{(i)}$
	Intitulé	Définition	
0-19	IF	$(X \wedge Y) \vee (X \wedge \neg Z)$	0x5A8227999
20-39	XOR	$(X \oplus Y \oplus Z)$	0x6ED9EBA1
40-59	MAJ	$(X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$	0x8F1BBCDC
60-79	XOR	$(X \oplus Y \oplus Z)$	0xCA62C1D6

Tableau 4.1 : définition de $f^{(i)}(X.Y.Z)$, et des constantes $K^{(i)}$

3) Pour chaque bloc m_i, on remplace *A.B.C.D* et *E* dans *AA, BB, CC, DD*, et *EE* est alors ajouté à *A,* appliquer la fonction de compression H à (AA, BB, CC, DD, EE, m_i).

Le résultat obtenu AA', BB', CC', DD' et EE' est alors ajouté à *A*, *B, C* et *E* (addition modulo 2^{32}).

4) Le résultat du hachage est obtenu par la concaténation des registres *A, B, C, D* et *E* obtenue après compression du dernier bloc m_n du message.

4.5.3 Description de la fonction de compression

Voyons maintenant le cœur de la fonction SHA, la fonction de compression.
On note tout d'abord ($W^{(0)}$,…, $W^{(15)}$) les 512 bits de messages entré dans H, représentés par 16 mots de 32 bits. La première étape consiste à effectuer une expansion de ces 512 bits :

$$W^{(i)} = W^{(i-3)} \oplus W^{(i-8)} \oplus W^{(i-14)} \oplus W^{(i-16)} \qquad \forall i,\ 16 \le i < 80 \quad (1)$$

Ces 80 mots de 32 bits sont utilisés pour altérer l'état interne de l'automate H constitué des cinq registres de 32 bits($A^{(i)},B^{(i)},C^{(i)},D^{(i)},E^{(i)}$) L'état initial ($A^{(0)},B^{(0)},C^{(0)},D^{(0)},E^{(0)}$) correspond à l'entrée (*AA,BB,CC,DD,EE*) de la fonction de compression.

La modification de l'états interne($A^{(i)},B^{(i)},C^{(i)},D^{(i)},E^{(i)}$) est effectuée de la façon suivante :

Pour i = 0 à 79

$A^{(i-1)}$ =ADD ($W^{(i)}$,ROL $_5(A^{(i)})$, $f^{(i)}$ ($B^{(i)},C^{(i)},D^{(i)}$) , $E^{(i)}$, $K^{(i)}$)

$B^{(i+1)} = A^{(i)}$

$C^{(i+1)}$ =ROL $_{30}$($B^{(i)}$)

$D^{(i+1)} = C^{(i)}$

$E^{(i+1)} = D^{(i)}$

La fonction $f^{(i)}$ et la constante $K^{(i)}$ sont définis par le tableau 4.1, et

ADD (U, V, W, X, Y) = $(U+V+W+X+Y) \bmod 2^{32}$ .Enfin ROL désigne la fonction qui sur un entier de 32 bits i effectue une rotation à gauche de i bits.

La sortie de la fonction de compression est donnée par les 160 bits obtenus à partir de l'état final

($A^{(80)}$, $B^{(80)}$, $C^{(80)}$, $D^{(80)}$, $E^{(80)}$).

Pour résumer, l'architecture de SHA peut être illustrée par la figure 4.1

Le standard est bien sûr complété par la présentation de vecteurs de tests qui permettent de contrôler la validité d'une implantation.

Elle consiste à remplacer dans le processus d'expansion d'écrit plus haut, l'équation de récurrence (1) par l'équation :

$$W^{(i)} = ROL_1 (W^{(i-3)} \oplus W^{(i-8)} \oplus W^{(i-14)} \oplus W^{(i-16)} \qquad \forall i,\ 16 \le i < 80 \quad (2)$$

La seule différence est donc cette rotation de un bit vers la gauche.

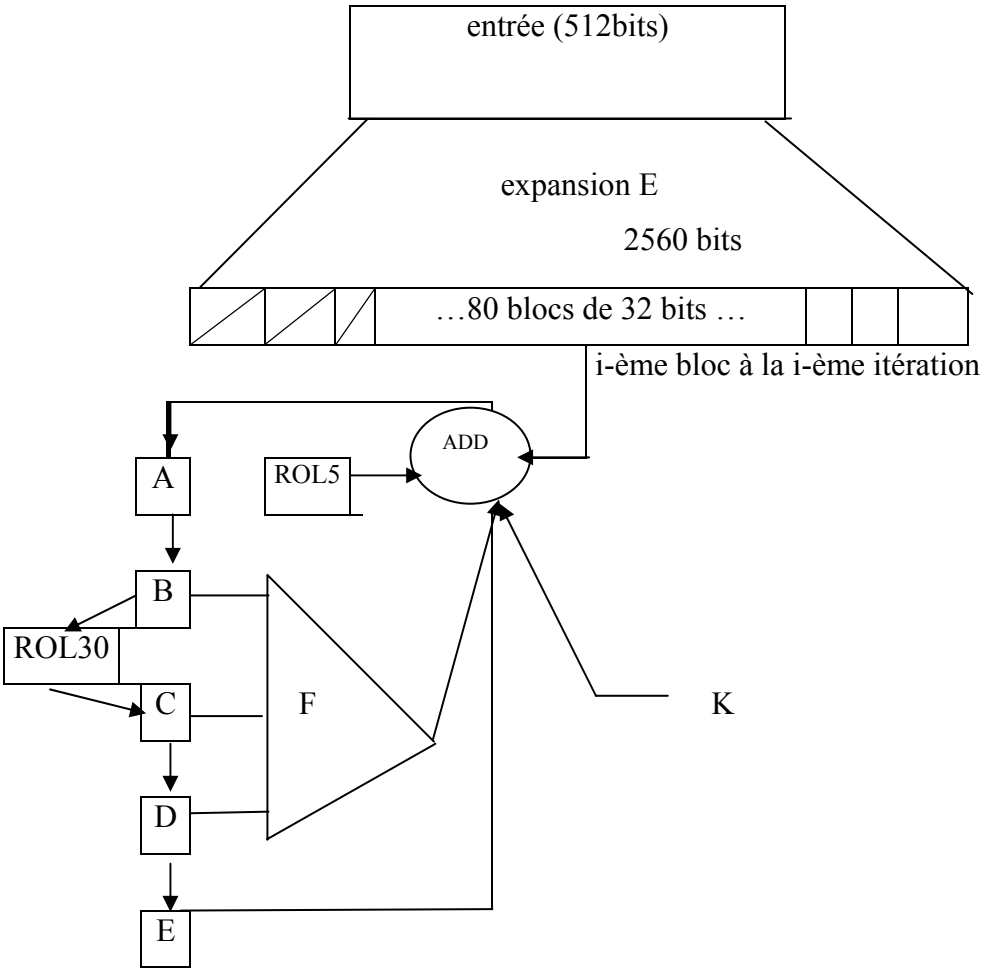


Figure 3.1 : Architecture de SHA

4.6 DSA : Digital signature Algorithme

Le DSA a été proposé en 1991 par le NIST comme standard : DSS (Digital signature standard).Sa sécurité est basée sur la difficulté du calcul du logarithme discret.

4.6.1 Génération des clés de DSA

1) Choisir un nombre premier p dont la longueur en bits, multiple de 64, est comprise entre 512 et 1024

2) Choisir un nombre premier q long de 160 bits qui divise $p-1$.

3) Choisir un générateur g du sous-groupe cyclique d'ordre q de $\mathbf{Z}_p^*$: on choisit pour cela $h \in \mathbf{Z}_p^*$ et on calcule $h^{(p-1)/q} \bmod p$ jusqu'à ce qu'on obtienne un résultat différent de 1.

4) Choisir x compris entre 1 et $q-1$.

5) Calculer $y = g^x \bmod p$.

p, q et g sont publics et peuvent être partagé par un groupe d'utilisateurs, x est la clé secrète de l'émetteur , y est sa clé publique.

4.6.2 Génération de signature de DSA

Pour signer le message m l'émetteur

1. choisit au hasard un nombre k compris entre 1 et $q-1$.
2. calcule $r = (g^k \bmod p) \bmod q$.
3. calcule $k^{-1} \bmod q$.
4. calcule $s = k^{-1}(H(m) + xr) \bmod q$, où H est la fonction de hachage du SHA.

Si r ou s est nul l'émetteur recommence (avec une autre valeur k).

La signature de l'émetteur pour le message m est le couple (r, s) .

4.6.3 Vérification de signature

Pour vérifier la signature de l'émetteur pour le message m le destinataire.

1. se procure les clés p, q, q et y .
2. calcule $w = s^{-1} \bmod q$.
3. calcule $u_1 = wH(m) \bmod q$.
4. calcule $u_2 = r\ w \bmod q$.
5. calcule $v = (g^{u_1} y^{u_2} \bmod p) \bmod q$.
6. accepté la signature si $v = r$.

4.6.4 Fonctionnement de DSA

D'après la définition de s on a :

$$H(m) = ks \bmod q .$$

Multipliant les deux membres par w , qui est l'inverse de s :

$wH(m) + xr\ w \equiv k$, c'est à dire $u_1 + xu_2 = k \bmod q$.

D'ou $g^{u_1 + x\ u_2} = g^{k + \lambda q}$ et, puisque $g^x \equiv y \bmod p$ et $g^q \equiv 1 \bmod q$.

Donc :

$$v = (g^{u_1} y^{u_2} \bmod p) \bmod q = (g^k \bmod p) \bmod q.$$

4.7 Conclusion

Les procédés de signature ont fait faire un grand pas à la cryptographie. Mais il reste des points faibles comme l'imitation des signatures.

Conclusion générale

Aujourd'hui nous vivons à un rythme accéléré .L'évolution rapide de la technologie engendrant l'utilisation de courrier électronique, téléphone mobile, mais dans tous ces échanges d'application (la confidentialité, l'intégrité des données et la sécurité des informations demeurent les points essentiels.

On a pu voir tout au long de ce développement sur les techniques de cryptographie , l'importance capitale tenue par l'algèbre et théorie des nombres ;en effet, la cryptographie tire le maximum de l'évolution de l'algèbre linéaire et de théorie des nombres algorithmique pour protéger au mieux des informations. Malgré ces progrès, la sécurité absolue des messages demeure une utopie.

Bibliographie

[1] François Arnault. *Théorie des nombres et cryptographie.* Université de Limoges Cours de D.E.A, *(*1999).

[2] Florent Chabaud. *Introduction à la cryptographie.* mastère « systèmes d'information » école supérieur d'Electricité.

[3] Rémi Ceillier. *La cryptographie*. Que sais-je ?. Edition presses universitaires de France. (1958).

[4] Joen Daeman et Vincent Rijmen. *The Rijndael Block Cipher* .(Septembre 1999).

[5] F.Diaz Y Diaz. *Algorithmique en théorie algébrique des nombres.*(DEA algorithmique 1995).Université de Bordeaux1, France.

[6] Jacques Stern. *La science du secret*. Edition Odile Jacob. (janvier 1998).

[7] Don B.Johnson et Alfred j.Menezes. *Elliptic curve DSA (ECDSA) :* An Enhanced DSA.

[8] Marc Joye. *Introduction élémentaire à la théorie des courbes elliptiques*. CG- 1995/1.

[9] J-M.Lamére,Y.Leroux et J.Tourly. *La sécurité des réseaux*. Méthodes et techniques. BORDAS , Paris, 1987.

[10] Antoine Mathys. *La cryptographie à clé publique et le système PGP*.

[11] Antoine Mathys. *Les clés logarithmiques ou l'algorithme de Diffie-Hellman.*

[12] A.Menezes, P.Van Oorschotet S.Vanstone. *Handbook of applied cryptography.* CRC Press, 1996.

[13] Jean-louis Poss. *Introduction à la cryptographie.* Ecole Nationale Supérieure d'Arts et Métiers, Aix.en-Provence, octobre 2000.

[14] R.L.Rivest, A.Shamir et L.Adleman. *A Methode for Obtaining Digital Signatures and Public-Key Cryptosystems.* Volume 21. Number 2. (February 1978).

[15] Bruce Schneir. *Cryptographie appliquée.* Algorithme, protocoles et codes source en C.(1995), traduction de Marc Vauclair, 1° Edition :international Thomson Publishing.

[16] Bruce Schneir. *Cryptographie appliquée.* Algorithme, protocoles et codes source en C. (1997), traduction de Marc Vauclair, 2° Edition :international Thomson Publishing.

[17] Ed Shaefer. *An introduction to cryptography.* Santa clara university.

[18] Douglas Stinson. *Cryptographie Théorie et pratique*. International Thomson publishing, traduit par S.Vaudenay 1996 de « cryptography : theory and pratique. », CRC, (1995).

[19] Jean Pierre Tual. *Cryptographie*. Doc. H 2 248-1.

[20] Gilles Zémor. *cours de cryptographie.*

[21] *Advanciced Encryption Standard (AES).* Fedéral Information Processing Standards Publication 197, (Novembre 2001).

[22] *http://ultralix. Polytechniques. fr/ gaudry/pres,html.*