REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE USTHB - BAB EZZOUAR – ALGER

FACULTE DE MATHEMATIQUES

DEPARTEMENT D'ALGEBRE ET THEORIE DES NOMBRES

Mémoire présenté pour l'obtention de
diplôme de PGS en cryptologie

Par

Abdel Halim ZAIDI

Ingénieur d'Etat en Informatique

Cryptanalyse des systèmes symétriques.

Cas d'étude l'AES.

Soutenu le 24/03/2004 devant le jury composé de :

Président :	
Mr A. AISSANI	Professeur / USTHB
Examinateurs :	
Mr S. KELLALI	Maître de conférences / EMP
Mr A. MERMOUL	Docteur / ENC
Promotrice :	
Mme H. DRIAS	Professeur / INI

Remerciements

Je suis très reconnaissant à Mme H. DRIAS, Directrice générale de l'INI d'avoir accepté de diriger ce travail. Je lui serais toujours reconnaissant pour l'ensemble des orientations et des conseils apportés pour mener à bien ce travail,,,

Je remercie les membres du jury d'avoir accepté de juger ce travail,,,

Mes remerciements vont aussi à tous ceux qui ont participé de près ou de loin dans l'élaboration de ce modeste travail,,,

A
tous ceux qui travaillent
sans trop parler…

Sommaire

Introduction générale

Partie I. Principes de base et terminologie de la cryptologie

1. Introduction générale 1

2. Cryptographie 1
2.1.Chiffrement et déchiffrement 2
2.2.Algorithmes de cryptage 2

3. Cryptanalyse 3
3.1. Définition 3
3.2. Cryptanalyse classique 5
3.2.1. Cryptanalyse des chiffres à substitution 5
3.2.2. Cryptanalyse des chiffres permutatifs 13
3.3. Cryptanalyse moderne 15
3.3.1. Attaques des fonctions de cryptage 15
a. Classement des attaques 15
b. Attaques sur les algorithmes symétriques 16
c. Attaques sur les algorithmes asymétriques 18
3.3.2. Attaques des protocoles cryptographiques 18

4. Conclusion 20

Partie II. Le standard de cryptage AES

1. Introduction 21

2. Faiblesses du standard DES 21

3. Le cahier des charges pour l'AES 23

4. Les propositions pour l'AES 23

5. La proposition retenue : RIJNDAEL 24
5.1. Présentation générale 24
5.2. Préliminaires mathématiques 26
5.3. Spécification technique 27
5.3.1. L'état, la clef et le nombre de tours 27
5.3.2. Les transformations élémentaires 28
5.3.2.1. La substitution 28
5.3.2.2. Décalage de ligne 29
5.3.2.3. Mixage de colonnes 29
5.3.2.4. Addition de clefs 30
5.3.3. Cadencement des clefs 30
5.3.4. Mode de fonctionnement du cryptosystème 31

5.4. Caractéristiques et points forts de l'AES 32

6. Conclusion 33

Partie III. Cryptanalyse de L’AES

1. Attaque basée sur les codes de répétition des textes clairs dépendants 34
2. Attaque du timing 41
3. Attaque algébrique (carré) 43
4. Etude comparative 44
5. Conclusion 48

Conclusion générale et perspectives 49

Références bibliographiques 50

Introduction générale

Devant le développement fulgurant des technologies de communication et la compétence accrue qu'exigent celles-ci, les processus traditionnels de communication et d'échange d'informations tendent à devenir obsolètes et inefficaces, c'est pourquoi l'introduction des procédés cryptographiques pour la protection des données est devenue plus qu'une nécessité. Mais en parallèle aux procédés cryptographiques, il y a des méthodes d'attaque, qui cherchent à contourner les protections proposées par la cryptographies ; c'est le rôle principal de la cryptanalyse qui devient elle aussi un élément très important, soit par le danger qu'elle présente aux processus cryptographiques ou bien par son intérêt dans la détection des failles et l'amélioration des cryptosystèmes.

Parmi les cryptosystèmes utilisés, on trouve le standard AES (Advanced Encryption Standard) mis en service en 2002. C'est un cryptosystème symétrique et itératif, composé d'opérations algébriques simples sur les octets ; cet aspect le rend très rapide, soit pour les architectures des processeurs des cartes à puces ou bien pour les implémentations logicielles, mais cette même structure algébrique simple le rend vulnérable à des attaques connues, et d'autres proposées spécialement contre l'AES.

Ce mémoire est organisé en trois grandes parties. Dans la première, nous présenterons les concepts de base et la terminologie de la cryptologie. Dans la seconde partie un tour d'horizon est fait sur le standard AES où on va présenter les différentes composantes et les étapes de cryptage. Dans la troisième partie, nous présenterons trois cryptanalyses proposées contre l'AES. Enfin, une étude comparative entre les trois cryptanalyses est faite, et un environnement d'attaque parallèle est étudié et proposé pour l'implémentation.

Partie I. Principes de base et terminologie de la cryptologie

1.1. Introduction

Lorsque deux individus, un émetteur (expéditeur) et un récepteur (destinataire), souhaitent communiquer entre eux, ils sont amenés à échanger des messages de longueur finie par un canal de transmission (téléphone, télex, réseau informatique…). L'expéditeur veut être sûr qu'aucun intermédiaire ne puisse agir sur le message de n'importe quelle façon que ce soit: spécifiquement, l'intermédiaire ne peut pas intercepter et lire le message, ni intercepter et modifier le message, ni fabriquer un message factice pouvant être considéré comme valide par le destinataire légitime.

Donc, si l'émetteur du message souhaite que celui ci ne soit accessible en clair qu'à son destinataire, il utilise alors un procédé cryptographique pour rendre le message inintelligible pour toute tierce partie branchée sur le canal de transmission.

L'accroissement considérable des moyens de communication et leur utilisation massive, ont rendu les informations de plus en plus vulnérables à l'intrusion des individus à la recherche d'informations tenues secrètes pour toutes sortes de raison. On fait de la protection du secret de la communication un problème majeur. C'est pour cette raison que le cryptage, est devenu depuis les années 1970, un domaine de recherche publique très actif. Et pour faciliter la lecture de ce mémoire, on commence par un ensemble de définitions nécessaires pour la compréhension.

En effet, la cryptographie est l'art de garder le secret des messages, elle est pratiquée par des cryptographes, tandis que la cryptanalyse est l'art de décrypter des messages chiffrés sans avoir le droit de le faire, pratiquée par les cryptanalystes. La branche des maths qui traite la cryptographie et la cryptanalyse s'appelle la cryptologie et ses pratiquants sont les cryptologues, la figure 1 présente l'arborescence de la cryptologie et ses branches **[2]**.

1.2. Cryptographie

C'est la science (discipline) qui garde le secret des messages, incluant les principes, les moyens et les méthodes de transformation des messages dans le but de masquer leur

contenu, empêcher leur modification ou une utilisation illégale, ainsi que les opérations inverses pour rendre le message de nouveau intelligible.

La cryptographie à toujours été une nécessité militaire : de tous les temps les armées ont dû transmettre des messages confidentiels qui, s'ils étaient interceptés par l'ennemi, ne devaient pas pouvoir être compris. Mais avec l'avènement des réseaux, et tout particulièrement Internet, la cryptographie prend maintenant une nouvelle dimension, économique cette fois. C'est en effet toute la sécurité du commerce électronique qui dépend maintenant de l'inviolabilité des codes cryptés : lorsque vous disposez d'un site sur Internet, vous avez en effet intérêt à ce que votre code soit crypté, sinon un pirate mal intentionné pourrait bien s'en emparer et s'en servir **[1].**

1.2.2. Chiffrement et déchiffrement

Un message est appelé **texte en clair.** Le processus de transformation d'un message de telle manière à le rendre incompréhensible est appelé **chiffrement.** Le résultat du chiffrement est appelé **texte chiffré** ou **cryptogramme.** Le processus de reconstruction du texte en clair à partir du texte chiffré est appelé **déchiffrement [2].**

1.2.3 Algorithmes de cryptage

Un algorithme cryptographique ou Cryptosystème est une fonction mathématique utilisée pour le chiffrement et le déchiffrement. Pour chiffrer un message on applique un algorithme de chiffrement au texte de ce message. Pour déchiffrer un texte chiffré on applique un algorithme de déchiffrement au texte chiffré. Il existe deux types d'algorithmes de cryptage:

a: Les Algorithmes restreints

La sécurité est basée sur le fait que l'algorithme est tenu secret (pas de clef). De tels algorithmes ne présentent plus qu'un intérêt historique car de nos jours ils sont inadéquats pour les besoins de sécurité.

b: Les Algorithmes à clef

Pour une vraie sécurité, les algorithmes modernes de chiffrement utilisent une **clef.** Cette clef peut prendre une valeur parmi un grand nombre de valeurs possibles. L'ensemble des valeurs possibles d'une clef est appelé **espace des clefs.** En effet, il y a deux types principaux d'algorithmes à clef : à clef secrète ou à clef publique :

- ***Les algorithmes à clef secrète (symétrique)***

C'est des algorithmes où la clef de chiffrement peut être calculée à partir de la clef de déchiffrement ou vice versa. Dans la plupart des cas, la clef de chiffrement et la clef de déchiffrement sont identiques. Pour de tels algorithmes l'émetteur et le récepteur doivent se mettre d'accord sur une clef à utiliser avant d'échanger des messages.

Les algorithmes à clef secrète peuvent être classés en deux catégories. Certains opèrent sur le message en clair un bit à la fois, ceux-ci sont appelés *algorithmes de chiffrement continu***.** D'autres opèrent sur le message en clair par groupes de bits, ceux-ci sont appelés *algorithmes de chiffrement par blocs.*

Les algorithmes à clef secrète posent un problème particulier : l'échange de la clef entre l'émetteur et le récepteur, mais avec l'apparition des algorithmes à clef publique (paragraphe suivant), le problème est résolu ; en effet la clef secrète est crypté par un algorithme à clef publique avant la transmission.

- ***Les algorithmes à clef publique (asymétrique) :***

Ils sont conçus de telle façon à ce que la clef de chiffrement soit différente de la clef de déchiffrement. De plus, la clef de déchiffrement ne peut pas être calculée a partir de la clef de chiffrement ; de tels algorithmes sont appelés à clef publique parce que la clef de chiffrement peut être rendue publique : n'importe qui peut utiliser la clef publique pour chiffrer un message mais seul celui qui possède la clef secrète peut déchiffrer le message chiffré. Donc, dans ce cas, on a une clef de chiffrement appelée **clef publique** et une clef de déchiffrement appelée **clef privée**.

L'apparition de la cryptographie à clef publique a été un très grand pas dans l'histoire de la cryptographie, et peut-être considérée comme la seule vraie révolution dans l'histoire entière de la cryptographie **[1].**

1.3. Cryptanalyse et attaque

1.3.1. Définition

La cryptanalyse est la science de la reconstitution du texte en clair sans connaître la clef de chiffrement. Une cryptanalyse réussie peut fournir, soit le texte en clair, soit la clef. La cryptanalyse peut également mettre en évidence les faiblesses d'un cryptosystème qui peuvent éventuellement faciliter les attaques contre celui-ci. Une tentative de cryptanalyse

est appelée **attaque.** Une attaque est basée sur l'hypothèse que le cryptanalyste connaît les détails de l'algorithme.

Il y a cinq types d'attaques cryptanalytiques contre les fonctions de chiffrement ; ces attaques sont classées selon les données dont dispose le cryptanalyste. Toutes ces méthodes reposent sur l'hypothèse que le cryptanlyste dispose de la connaissance complète de l'algorithme de chiffrement :

1- Les attaques à texte chiffré connu;
2- Les attaques à texte clair connu;
3- Les attaques à texte clair choisi;
4- Les attaques à texte clair choisi adaptatif;
5- Les attaques à texte chiffré;

Dans le domaine de la cryptanalyse, on peut citer deux grande, catégories d'attaques : celles utilisées contre les méthodes de cryptage classique, qui traitent la substitution et la transposition simple ; d'autre part il y a les méthodes modernes qui s'attaquent aux algorithmes de cryptage moderne ou bien aux protocoles cryptographiques.

Dans la suite de cette première partie on va présenter les deux catégories : classique et moderne avec des exemples.

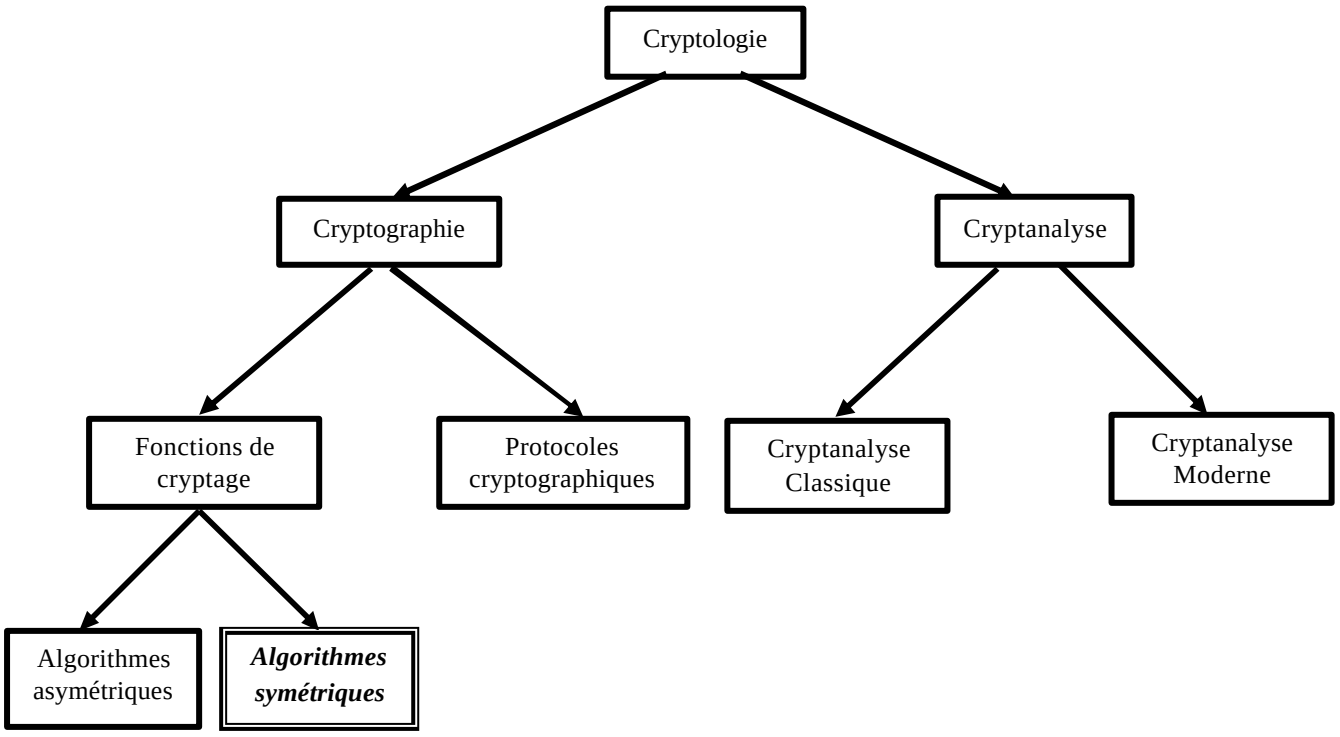


Figure 1.1. La cryptologie et ses branches

1.3.2. Cryptanalyse classique: Substitution et transposition

1.3.2.1. Chiffre à substitution

Dans ce type de chiffrement, chaque caractère du texte en clair est remplacé par un autre caractère de l'alphabet. Il y a deux types de substitution: *monoalphabétique et polyalphabétique.*

a. Chiffrements monoalphabétiques

C'est un chiffre dans lequel chaque caractère du texte en clair est remplacé par un seul caractère de l'alphabet.

- **Chiffre "Jules César"**

Soit n un entier, et Pi un caractère du texte en clair alors Ci le caractère chiffré correspondant est défini par un décalage de n positions :

Ci = E(Pi) = (Pi + n) modulo la taille de l'alphabet utilisé.

Exemple : n = 3 et l'alphabet c'est l'ensemble des lettres majuscules et minuscules (52 lettres) :

Message en clair : ***Ce message est bien lisible***

Message chiffré : ***Fh phvvdjh hvw elhq olvleoh***

Le principe de cet algorithme est très simple et rapide, même la clef utilisée est simple, mais son inconvénient est qu'il est très facile à casser en utilisant des indices linguistiques tirés du texte chiffré, même pour un message court où on a un minimum d'informations.

- **Cryptanalyse "Jules César"**

Soit le message crypté suivant : ***lo hvw idfloh gh eulvhu fh phvvdjh*** **[10].**

On a les indices suivants :

Trois mots de deux lettres:

- ***lo*** : la deuxième lettre *"**o**"* est 3 fois plus loin que la 1^{ere} *"**l**"* : deux possibilités: ***or*** ou ***il***.
- ***gh*** : la deuxième lettre suit directement la première donc deux possibilités: ***de*** ou ***tu***.
- ***fh*** : la deuxième lettre est 2 fois plus loin que la 1^{ere} donc deux possibilités: ***ce*** ou ***su***.

Deux indices suggèrent que ***h*** devient ***e*** ou bien ***u***, donc n = 3 ou n = 16;

Donc, il est facile de trouver le message en clair :

Si n = 16 alors on trouve une phrase incompréhensible;

Si n = 3 alors on trouve la phrase: ***il est facile de briser ce message.***

En général si le message est plus compliqué, on peut essayer toutes les clefs possibles par ordinateur, où le nombre de clefs possibles = taille de l'alphabet, donc c'est un algorithme polynomial.

- **Permutation aléatoire**

Dans ce cas, on utilise une table de substitution des caractères où on fait la substitution de chaque caractère du message par le caractère correspondant de la table.
Soit par exemple la table de substitution suivante :

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

Exemple :

Texte clair : **IL EST DIFFICILE DE BRISER CE MESSAGE**

Texte chiffré : **OS TLZ ROYYOEOST RT WKOLTK ET MTLLQUT**

Cette méthode est plus sûre par rapport à "Jules César", car une approche "force brute" exige (T)! essais pour trouver le texte clair, où T est la taille de l'alphabet ; mais son principal inconvénient d'une part, la difficulté de se souvenir de la table de substitution et d'autre part elle peut être brisée par des outils tels que l'analyse fréquentielle.

- **Fréquences des caractères**

En français, comme en d'autres langues, certaines lettres apparaissent plus souvent que les autres dans les mots de la langue, la table suivante donne les fréquences des lettres de la langue française :

a	8.39%	g	0.95%	m	2.94%	s	8.01%	y	0.32%
b	0.77%	h	0.80%	n	7.56%	t	7.48%	z	0.12%
c	3.33%	i	8.18%	o	5.37%	u	5.98%		
d	4.07%	j	0.64%	p	3.21%	v	1.58%		
e	14.50%	k	0.64%	q	1.26%	w	0.01%		
f	1.21%	l	5.84%	r	7.02%	x	0.41%		

Tableau 1 . Fréquence d'apparition des lettres

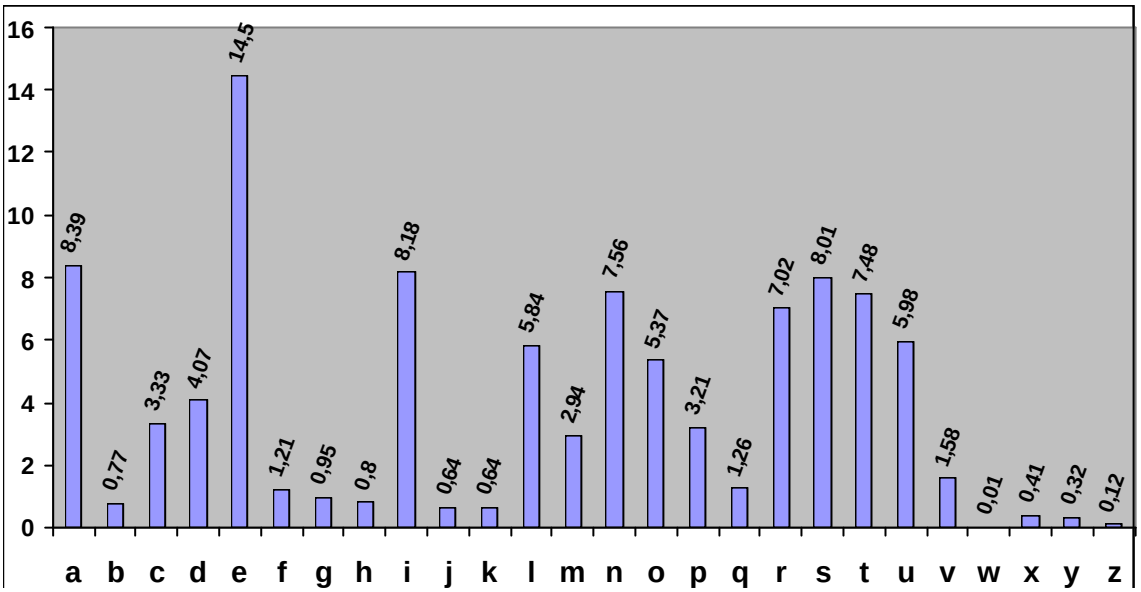


Figure 1.2. Fréquence des lettres

- **Cryptanalyse du chiffre à permutation aléatoire**

Soit le message chiffré suivant :

g cdbowaumzc ciu yd fswcd v miiyoco vci bsffydkbmuksdi icbyokumkoci iyo vci bmdmye dsd icbyokumkoci amo g yimzc vc g cdbowaumzc sd vczykic gc fciimzc asyo pyc bcgyk bk dc aykiic ami cuoc octcgc fcfc ik gm uomdifkiiksd m cuc kducobcaucc **[10].**

L'analyse fréquentielle donne :

a	2.98%	g	2.98%	m	6.38%	s	2.98%	y	5.53%
b	4.26%	h	0.00%	n	0.00%	t	0.43%	z	2.13%
c	**15.74%**	i	**9.36%**	o	6.38%	u	4.68%		
d	5.96%	j	0.00%	p	0.43%	v	2.13%		
e	0.43%	k	5.96%	q	0.00%	w	1.28%		
f	2.98%	l	0.00%	r	0.00%	x	0.00%		

1. Essayons c ≡ e (fréquence max):

   g edbowaumze eiu yd fswed v miiyoeo vei bsffydkbmuksdi iebyokumkoei
   iyo vei bmdmye dsd iebyokumkoei amo g yimze ve g edbowaumze sd
   vezykie ge feiimze asyo pye begyk bk de aykiie ami euoe oetege fefe ik gm
   uomdifkiiksd m eue kdueobeauee

2. Un triplet **eue**

   On a trois possibilités: ère, été, Ève.

   Dans le texte on a : t ≡ 7.47%; r ≡ 7.02%; v ≡ 1.57%.

   Essayons u ≡ t :

   g edbowatmze eit yd fswed v miiyoeo vei bsffydkbmtksdi
   iebyoktmkoei iyo vei bmdmye dsd iebyoktmkoei amo g yimze ve g
   edbowatmze sd vezykie ge feiimze asyo pye begyk bk de aykiie
   ami etoe oetege fefe ik gm tomdifkiiksd m ete kdteobeatee

3. Trois mots de deux lettres finissant par e : ve, ge, de.

   Possibilités: ce, de, hé, le, me, ne

   Dans le texte on a : d ≡ 5.96%; g ≡ 2.98%; v ≡ 2.13%;

   En français on a : n ≡ 7.55%; l ≡ 5.84%; d ≡ 4.07%; c ≡ 3.32%; m ≡ 2.93%; h ≡ 0.79%;

   Essayons d ≡ n; g ≡ l; v ≡ d :

   l enbowatmze eit yn fswen d miiyoeo dei bsffynkbmtksni iebyoktmkoei
   iyo dei bmnmye nsn iebyoktmkoei amo l yimze de l enbowatmze sn
   dezykie le feiimze asyo pye belyk bk ne aykiie ami etoe oetel e fefe ik l m
   tomnifkiiksd m ete knteobeatee

4. Considérons le triplet **e**i**t** :

   Le seul mot possible c'est "est", donc i ≡ s

Maintenant le texte devient :

l enbowatmze est yn fswen d mssyoeo des bsffynkbmtksns sebyoktmkoes
syo des bmnmye nsn sebyoktmkoes amo l ysmze de l enbowatmze sn
dezykse ge fessmze asyo pye begyk bk ne ayksse ams etoe oetel e fefe sk lm
tomnsfkssksd m ete knteobeatee

Parmi les lettres chiffrées restantes du message, on a celles à haute fréquence :

m ≡ o ≡ 6.38%, k ≡ 5.96%, y ≡ 5.53%, a ≡ f ≡ s ≡ 2.98%

Parmi les lettres non identifiées, celles à haute fréquence sont :

a ≡ 8.39%, i ≡ 8.18%, r ≡ 7.02%, u ≡ 5.98%, o ≡ 5.37%, p ≡ 3.21%, m ≡ 2.94%

Alors "o" pourrait donner "i" ou bien "r"; le mot "*etoe*" ne peut être que "*etre*" donc o ≡ r

De même l'hypothèse y ≡ u transforme *yn* en *un* et *syr* en *sur*;

D'autre part "k" pourrait donner "i", "o" donc "*sk*" ne peut être que "*si*" car "*so*" n'est pas un mots.

Pour la lettre unique "m": la seule possibilité est "*a*".

Essayons avec m ≡ a, o ≡ r, k ≡ i et y ≡ u :

l enbrwataze est un fswen d assurer des bsffunibatisns seburitaires sur des banaue nsn seburitaires aao l usaze de l enbrwataze sn dezuise le fessaze asur pue belui bi ne auisse aas etre retele fefe si la transfissisd a ete interbeatee

Il nous reste un doublon "ff" avec f ≡ 2.98%.

Si on retire les lettres déjà identifiées, il reste o ≡ 5.37%, p ≡ 3.21%, m ≡ 2.94%, alors les seules possibilités sont f ≡ m ou bien f ≡ p :

Donc le mot "*fefe*" donnerait soit "*meme*" ou "*pepe*" (même ou pépé).

Essayons avec f ≡ m :

l enbrwataze est un mswen d assurer des bsmmunibatisns seburitaires sur des banaue nsn seburitaires aao l usaze de l enbrwataze sn dezuise le messaze asur pue belui bi ne auisse aas etre retele meme si la transmissisd a ete interbeatee

Une fois qu'on a un nombre suffisant de lettres identifiées, les autres peuvent l'être par une simple inspection, donc :

b ≡ c, w ≡ y, a ≡ p, s ≡ o, e ≡ x, z ≡ g, p ≡ q, t ≡ v.

On aura donc le texte en clair :

l encryptage est un moyen d assurer des communications securitaires sur des canaux non securitaires par l usage de l encryptage on deguise le message pour que celui ci ne puisse pas etre revele meme si la transmission a ete interceptee

En fin la table de substitution est la suivante :

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
P	C	E	N	X	M	L		S		I		A		R	Q			O	V	T	D	Y		U	G

b. Chiffres polyalphabétiques

Le but principal de cette méthode est de rendre le chiffre plus difficile à casser par l'analyse fréquentielle. Donc chaque caractère peut être substitué par plusieurs caractères en utilisant des décalages ou une substitution différente.

- **Chiffres de Vigenère**

On utilise le tableau suivant :

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

Tableau 2. Tableau de Vigenère

➢ **Utilisation du tableau de Vigenère**

- On choisit une clef alphabétique, on la répète autant de fois pour l'avoir aussi longue que le message.
- Placer le message et la clef en correspondance.
- Pour chaque paire (caractère de la clef, caractère du message), le caractère de la clef donne la rangée et le caractère du message donne la colonne.

Exemple : soit ***GLADIATOR*** la clef alphabétique;
Et le message a chiffrer est : ***UTILISATION DU TABLEAU DE VIGENERE***

GLADIATORGL AD IATORGL AD IATORGLA
UTILISATION DU TABLEAU DE VIGENERE

On aura donc : G+U=A; L+T=E; A+I=I; …; L+R=C; A+E=E.
Apres un regroupement de cinq caractères, le message chiffré sera :

AEIOQ STHZU YDXBA UZVGF DHDIZ SEKCE

➢ **Cryptnalyse du chiffre polyalphabétique**

- Trouver la taille de la clef.
- Séparer les caractères en groupes reliés aux même caractères.
- Utiliser l'analyse fréquentielle.

1. Méthode de Kasiski :

Le but est de trouver la taille de la clef alphabétique utilisée. Cette méthode est fondée sur le fait qu'il y a des groupes de lettres qui se répètent en français comme en anglais ;

Par exemple, les fins de mots en : er, et, eau, ion, ent, ont, ant, ...

Description de la méthode

- Identifier des séquences de trois caractères ou plus qui se répètent dans le texte chiffré;
- Noter la position de départ de chaque séquence;
- Calculer les différences entre les points de départ des séquences successives;

- Calculer les facteurs de ces différences;
- S'il s'agit d'un chiffre polyalphabétique, la longueur de la clef est un de ces facteurs;

Exemple : Soit le texte suivant qui sera crypté avec la clef « gladiat » :

dans le sens de la théorie de l information la substitution provoquera la diffusion
glad ia tgla di at gladiat gl a diatgladiat gl adiatgladiat gladiatgla di atgladiat

le résultat du chiffrement est le suivant :
jlnv te lkys gm lt zserzix jp l lxfhxyawqog rl sxjsmoeuwqog vcoywqnkcq oi dblquv qog

La séquence **qog** se présentera aux positions 33, 47 et 68, soit des différences de 14=2x7 et 21=3x7.
On trouve des longueurs possibles de 2, 3 ou 7 ; donc 7 c'est le plus probable et c'est la longueur de la clef utilisée « gladiat ».

2. Indice de coïncidence

C'est un outil qui mesure jusqu'à quel point une distribution des caractères correspond à celle d'une langue donnée ; donc chaque groupe de lettres a un indice de coïncidence spécifique.
Soit un caractère a quelconque et Pa la probabilité d'apparition de ce caractère; on a alors:

- Pa + Pb +…… + Pz = ? Pa pour a = a…..z
- Si la distribution est uniforme, alors Pa vaut 1/26 = 0.0384
- La variance est donnée par (? Pa)^2 -1/26 ; Pour a = a…..z ;
- Si la distribution est uniforme, alors la variance = 0

L'indice de coïncidence est un moyen d'estimer la variance à partir de données observées :

IC = ? (Freq$_a$ * (Freq$_a$ - 1)) / (n * (n - 1) ?

1.3.2.2. Chiffre permutatif : transposition de colonnes

Un chiffre à transposition est un chiffre dans lequel les caractères du texte clair demeurent inchangés, mais les positions respectives sont modifiées. Pour la transposition en colonnes, on écrit le texte en clair horizontalement, et le texte chiffré est relevé verticalement.

Soit le texte en clair suivant **[10]**:

" DANS LE SENS DE LA THEORIE DE L INFORMATION LA SUBSTITUTION PROVOQUERA LA DIFFUSION "

On écrit le texte horizontalement par morceau de dix caractères :

DANSLESENS
DELATHEORI
EDELINFORM
ATIONLASUB
STITUTIONP
ROVOQUERAL
ADIFFUSION

Le texte chiffré est obtenu par lecture verticale :

DDEAS RAAED TTODN LEIIV ISALO TOFLT INUQF EHNLT UUSEF AIESE OOSOR INRRU NAOSI MBPLN

- **Cryptanalyse d'une transposition par colonnes**

Il semble facile de le casser, parce que les lettres du texte en clair demeurent en clair. En effet si l'analyse fréquentielle donne une répartition normale par rapport à la table des fréquences des caractères, on peut alors faire l'hypothèse d'une transposition par colonnes.

L'attaque est fondée principalement sur l’analyse exhaustive des paires de caractères et des triplets pour découvrir le nombre de colonnes utilisées.

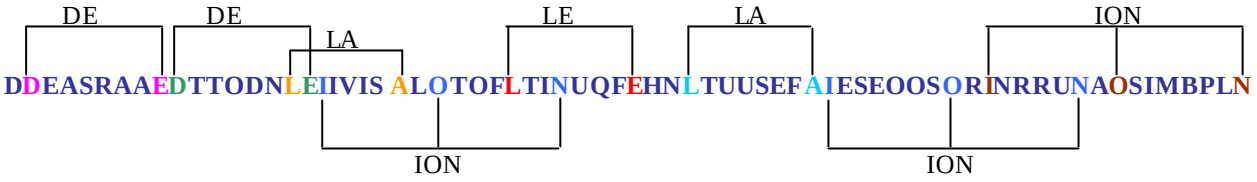
Par exemple :

Les paires de caractères comme : ou, oi, tr, pr, qu, il, le, la, de, etc…

Les triplets comme : ent, ion, les, des, etc…

La distance entre les caractères d'une même paire ou d'un même triplet correspond au nombre de rangées ou bien un multiple de ce nombre moins un.

Exemple : Soit le texte chiffré suivant :



Dans l'exemple les paires utilisées sont : DE, LA, Le ; et le triplet est : ION

On peut facilement déduire que le nombre de rangés utilisés est 6+1=7, où 6 = la distance entre deux caractères des différents diagrammes, par exemple pour le 1er "DE" en rose, la distance entre D et E est de 6 caractères. Donc pour reconstituer le texte en clair, on divise le texte chiffré en N morceaux sur 7 caractères et on les dispose en colonnes.

Donc :

le 1er morceau : **DDEASRA**

le 2eme morceau : **AEDTTOD**

le 3eme morceau : **NLEIIVI**

le 4eme morceau : **S ALOTOF**

le 5eme morceau : **LTINUQF**

le 6eme morceau : **EHNLTUU**

le 7eme morceau : **SEFAIES**

le 8eme morceau : **EOOSORI**

le 9eme morceau : **NRRUNAO**

le 10eme morceau : **SIMBPLN**

Donc, si on fait la lecture des morceaux en colonnes on trouve le texte en clair :

DANS LE SENS DE LA THEORIE DE L INFORMATION LA SUBSTITUTION PROVOQUERA LA DIFFUSION

1.3.3. La cryptanalyse Moderne

Si le but principal de la cryptographie est la mise en œuvre des méthodes de protection, la cryptanalyse a pour but le contraire : casser ces protections. Une tentative de cryptanalyse d'un cryptosystème est appelée attaque, et le résultat peut être :

- Un cassage complet : où le cryptanalyste découvre la clef complète.
- Obtention globale : dans ce cas le cryptanalyste trouve un algorithme de déchiffrement qui ne nécessite pas la connaissance de la clef de déchiffrement.
- Obtention locale : le cryptanalyste retrouve le texte en clair correspondant à un message chiffré.
- Obtention d'information : le cryptanalyste obtient des indices sur le texte en clair ou la clef (une partie des bits de la clef ou bien des indications sur la forme du texte en clair,...)

L'hypothèse de base est qu'on suppose toujours que le cryptanalyste connaît les détails des cryptosystemes, ou bien les protocoles employés. Même si ce n'est pas toujours le cas en pratique, il serait risqué de se baser seulement sur le secret des mécanismes utilisés pour assurer la sécurité d'un système, d'autant plus que l'usage grandissant de l'informatique rend de plus en plus facile la décompilation (reconstitution) de l'algorithme à partir du programme.

1.3.3.1. Attaques des fonctions de crypatge

Les fonctions de cryptage sont supposées rendre impossible le décryptage, c'est-à-dire la récupération du texte clair sans l'utilisation de la clef. A fortiori, ces fonctions doivent protéger le secret des clefs.

a. Classement des attaques en fonction des données dont dispose le cryptanalyste

On distingue plusieurs types d'attaques suivant les informations qui sont à la disposition du cryptanalyste. Ce sont :

- **L'attaque à texte chiffré seulement :** dans cette attaque le cryptanalyste ne connaît qu'un ensemble de textes chiffrés ; il peut soit retrouver seulement les textes en clair,

soit retrouver la clef. En pratique, il est très souvent possible de deviner certaines propriétés du texte en clair, ce qui permet de valider ou non le décryptage.

- **L'attaque à texte en clair connu :** dans ce cas en plus des textes chiffrés, le cryptanalyste connaît aussi les textes en clair correspondants ; son but est alors de découvrir la clef. Du fait de la présence, dans la plupart des textes chiffrés, de parties connues comme les en-têtes de paquets, les champs communs à tous les fichiers d'un même type,... ce type d'attaques est très pratique.

- **L'attaque à texte en clair choisi :** où le cryptanalyste peut choisir des textes en clair à chiffrer d'une manière bien étudiée pour qu'ils apportent plus d'informations sur la clef. Si le cryptanalyste peut de plus adapter ses choix en fonction des textes chiffrés précédents, on parle d'**attaque à texte en clair choisi adaptative**.

- **L'attaque à texte chiffré choisi :** c'est l'inverse de l'attaque précédente : dans ce cas le cryptanalyste peut choisir des textes chiffrés pour lesquels il connaîtra le texte en clair correspondant ; sa tâche est alors de retrouver la clef. Ce type d'attaques est principalement utilisé contre les systèmes à clef publique, pour retrouver la clef privée.

b. Attaques sur les algorithmes symétriques

Attaques au niveau des clefs

L'attaque la plus simple est l'attaque exhaustive ou bien l'attaque en force brute, qui consiste à essayer toutes les clefs possibles. Donc avec une clef de 128 bits par exemple, cela donne 2^{128} cas possibles.

D'autre part, si la clef recherchée est un mot de passe ou bien si elle dérive d'un mot de passe, on a des chances de trouver la clef en testant des mots susceptibles d'avoir été choisis comme mot de passe. Dans ce cas, c'est une attaque par dictionnaire, car le cryptanalyste se constitue un dictionnaire de mots à tester (une liste de noms ou de prénoms, l'ensemble des mots d'une langue donnée,...). Cette attaque est beaucoup plus rapide qu'une attaque exhaustive.

La cryptanalyse différentielle

La cryptanalyse différentielle est utilisée contre les algorithmes de chiffrement par blocs itératifs. C'est une attaque à texte en clair choisi et elle se base sur l'observation des

différences entre deux textes clairs lorsqu'ils sont chiffrés avec la même clef. En analysant ces différences entre paires de texte clair / texte chiffré, il est possible d'attribuer des probabilités à chaque clef possible. À force d'analyser des paires de textes, on finit soit par trouver la clef recherchée, soit par réduire suffisamment le nombre de clefs possibles pour pouvoir mener une attaque exhaustive rapide.

Ce type d'attaques a été utilisé pour la première fois par Murphy en 1990 contre l'algorithme de cryptage FEAL-4. Biham et Shamir ont mené des attaques différentielles sur le standard de cryptage DES, mais leur efficacité reste limitée à cause de la conception des tables de substitution S-box du DES, qui avaient été optimisées contre ce type d'attaques. La meilleure attaque différentielle contre le DES complet à 16 rondes nécessite 2^{47} textes en clair choisis. Cela reste énorme et demande beaucoup trop de chiffrement.

La cryptanalyse linéaire

La cryptanalyse linéaire est une attaque à texte en clair connu et consiste à modéliser l'algorithme de chiffrement par blocs par un modèle mathématique ; souvent, c'est une approximation linéaire. Avec un nombre suffisant de paires texte clair / texte chiffré, on peut deviner certaines bits de la clef.

La cryptanalyse linéaire fut utilisée pour la première fois en 1992 par Matsui et Yamagishi pour attaquer le cryptosysteme FEAL. Elle fut étendue par Matsui en 1993 pour attaquer le standard de cryptage DES. Les tables de substitution S-box du DES ne sont pas optimisées pour contrer la cryptanalyse linéaire, et la meilleure attaque linéaire actuelle contre le DES nécessite 2^{43} textes en clair connus en moyenne. Une réalisation logicielle de cette attaque a découvert une clef DES en 50 jours avec 12 stations de travail.

En 1994, Langford et Hellman introduisirent une attaque appelée cryptanalyse différentielle linéaire qui combine des éléments des deux méthodes précédentes.

c. Attaques sur les algorithmes asymétriques

Attaques au niveau des clefs

Avec les algorithmes asymétriques, le problème n'est pas de trouver la bonne clef par attaque exhaustive, mais de dériver la clef secrète à partir de la clef publique. La plupart des algorithmes asymétriques reposant sur des problèmes mathématiques difficiles à résoudre ; cela revient généralement à résoudre ce problème. C'est pourquoi les

paramètres qui influencent la difficulté de résolution du problème sont les éléments déterminant la sécurité. Généralement, cela se traduit par la nécessité d'utiliser de grands nombres ; la taille de ces nombres ayant une répercussion sur la longueur des clefs. Cela explique que les clefs utilisées par la cryptographie à clef publique sont généralement bien plus longues que celles utilisées par la cryptographie à clef secrète.

Par exemple, dans le cas de RSA, l'élément déterminant est la taille du module. La factorisation d'un module de 512 bits est à la portée d'une agence gouvernementale, 1024 bits est considéré comme sûr actuellement, et 2048 bits garantit une sécurité à long terme.

Attaque à texte en clair deviné et problème de la faible entropie

Un point faible des algorithmes à clef publique est le caractère public de la clef de chiffrement : le cryptanalyste ayant connaissance de cette clef, peut mener une attaque à texte en clair deviné, qui consiste à tenter de deviner le texte en clair et à le chiffrer pour vérifier son exactitude. Cette particularité implique donc une restriction sur l'utilisation des algorithmes asymétriques : il faut absolument éviter de les utiliser avec un ensemble de textes en clair possibles restreint, c'est le probleme de faible entropie. En effet, dans ce cas, l'attaquant peut aisément se constituer une liste exhaustive des textes en clair possibles et des textes chiffrés correspondants.

Attaque temporelle (par timing)

Un type d'attaque apparu récemment est l'attaque temporelle, qui utilise la mesure du temps pris par des opérations cryptographiques pour retrouver les clefs privées utilisées. Cette attaque a été réalisée avec succès contre des cartes à microcircuits, et contre des serveurs de commerce électronique à travers l'Internet. La société Counterpane Systems, a généralisé ces méthodes pour y inclure des attaques sur des systèmes en mesurant la consommation, les émissions radioélectriques; ils les ont mises en œuvre contre plusieurs types d'algorithmes à clef publique ou à clef secrète utilisés dans des calculettes. Une recherche voisine s'est intéressée à l'introduction d'erreurs dans les processeurs cryptographiques pour tenter d'obtenir des informations sur la clef.

1.3.3.2. Attaques des protocoles cryptographiques

Un protocole cryptographique est un protocole entre deux personnes, qui utilisent des procédés cryptographiques dans le but de garantir la confidentialité ou pour partager un

secret pour le calcul d'une valeur ou bien générer une suite aléatoire, et aussi pour confirmer l'identité. On distingue deux types d'attaques sur les protocoles : les attaques passives et les attaques actives. Dans le premier cas, l'attaquant ne peut qu'espionner les données échangées par les tiers communicants, alors que dans le second cas il peut modifier ou supprimer des messages, en ajouter des nouveaux ou des anciens.

a. Attaque par mascarade

On parle d'attaque par mascarade (*spoofing attack*) lorsqu'un attaquant essaye de se faire passer pour un utilisateur légitime en exécutant un protocole à la place de celui-ci. La conception d'un protocole a bien sûr pour but principal de contrecarrer ce type d'attaques.

b. Attaque par rejeu

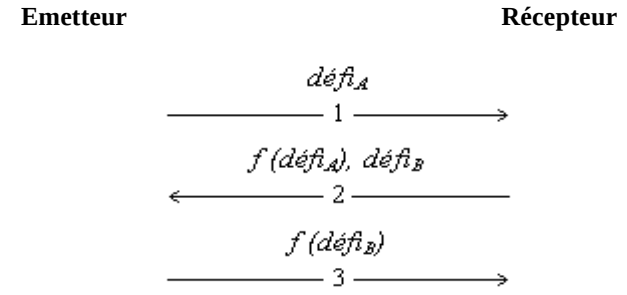
Une attaque par rejeu (*replay attack*) consiste à envoyer, dans une communication, des messages interceptés au cours d'une autre communication ou plus tôt dans la communication. Ce type d'attaques permet de contourner des protocoles simples comme, par exemple, une authentification par mot de passe : il suffit à un adversaire d'avoir espionné un échange pour connaître le mot de passe et donc pour pouvoir se faire passer pour un utilisateur légitime.

Les méthodes pour se prémunir contre ce type d'attaques sont l'utilisation de marquages temporels (*timestamps*) ou de défis imprévisibles et à usage unique.

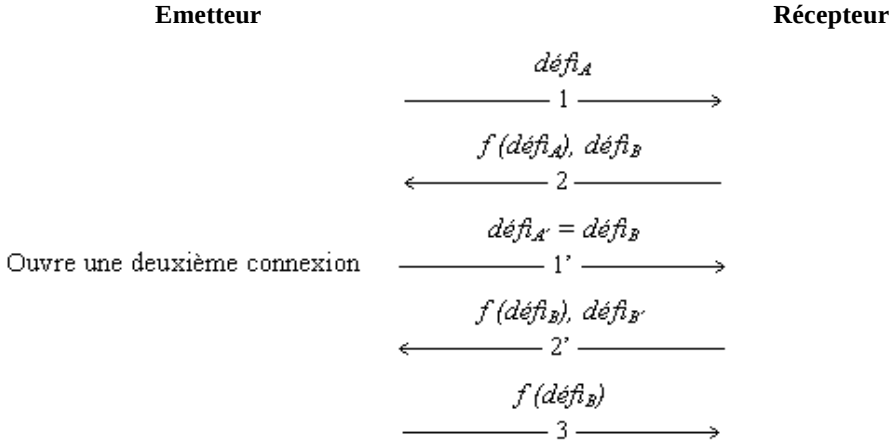
c. Attaque par réflexion

Une attaque par réflexion (*reflection attack*) est une attaque pour laquelle l'adversaire exploite le caractère symétrique d'un protocole pour répondre aux défis de son interlocuteur en utilisant des réponses fournies par l'interlocuteur lui-même.

Considérons par exemple le protocole d'authentification mutuelle suivant :



Pour se faire passer comme Emetteur auprès du Récepteur, il suffit à un adversaire d'établir deux connexions en parallèle avec le Récepteur, et d'envoyer $défi_{A'} = défi_B$ comme second défi au Récepteur. Celui-ci fournira alors la réponse attendue à son premier défi :



d. Attaque de l'intercepteur

Le principe de l'attaque de l'intercepteur (*man-in-the-middle attack*) est que, pendant que deux tiers procèdent à un échange de clef, en utilisant un algorithme du type Diffie–Hellman par exemple, un adversaire se positionne entre les deux tiers et intercepte les échanges. De cette façon, il procède à un échange de clef avec chaque tiers. A la fin du protocole, chaque tiers utilisera donc une clef différente, chacune de ces clefs étant connue de l'intercepteur. Pour chaque message transmis par la suite, l'intercepteur procédera à son déchiffrement avec la clef correspondante puis le rechiffrera avec l'autre clef avant de l'envoyer à son destinataire : les deux tiers croiront communiquer de façon sûre alors que l'intercepteur pourra en fait lire tous les messages, voire même créer de faux messages.

1.4. Conclusion

Dans cette première partie, on a fait le tour d'horizon de la cryptologie où on a présenté les deux principales branches : la cryptographie et la cryptanalyse ; la première branche s'intéresse à la protection des messages tandis que la deuxième fait l'inverse, essayés de briser ces protections, et dans les deux cas on a présenté les intérêts, les avantages et les inconvénients des deux branches : la cryptographie et la cryptanalyse.

Partie II. Le standard de cryptage AES

2.1. Introduction

L'AES (Advanced Encryption Standard) est le nouveau standard de cryptage symétrique (à clef secrète), destiné à remplacer le DES (Data Encryption Standard) qui est devenu trop faible au regard des attaques actuelles. L'AES a été choisi en octobre 2000 parmi les 15 cryptosystèmes proposés en réponse à l'appel d'offres lancé par le NIST (National Institute of Standards and Technology). Cet algorithme, initialement appelé **RIJNDAEL [4]**, opère sur des blocs de messages de 128 bits et est disponible pour trois tailles de clef différentes : 128, 192 et 256 bits.

2.2. Faiblesses du standard DES

Le standard de chiffrement DES (Data Encryption Standard) conçu par la société IBM en 1976 en réponse à un appel qui date de 1972 du NBS (National bureau of standard, maintenant dénommé NIST, bureau fédéral des standards des Etats-Unis) en quête d'un standard de cryptage d'un niveau de sécurité élevée, simple, et facilement programmable. Son principal avantage réside dans sa rapidité (relativement par rapport à la sécurité qu'il apporte). Il peut être développé en moins de 200 lignes de programme. L'algorithme à été fait tel qu'il était facilement implémentable en hardware, sur des cartes électroniques spécialisées, ou sur des systèmes de communication. En effet, il n'utilise que des opérations arithmétiques et logiques sur des blocs de 64 bits. Le schéma général du standard DES est illustré dans la figure 1.

La NSA (National Agency of security) a choisi les tables-S du DES, qui sont des tables de constantes qui contrôlent les substitutions dans l'algorithme. Peut-être pour s'assurer qu'IBM n'ait pas introduit une brèche secrète. Les cryptologues ont longtemps mis en doute la NSA, d'avoir mis elle-même une brèche secrète. La NSA a également ajouté que la conception des tables-S était des détails sensibles qui ne seraient pas rendus publics.

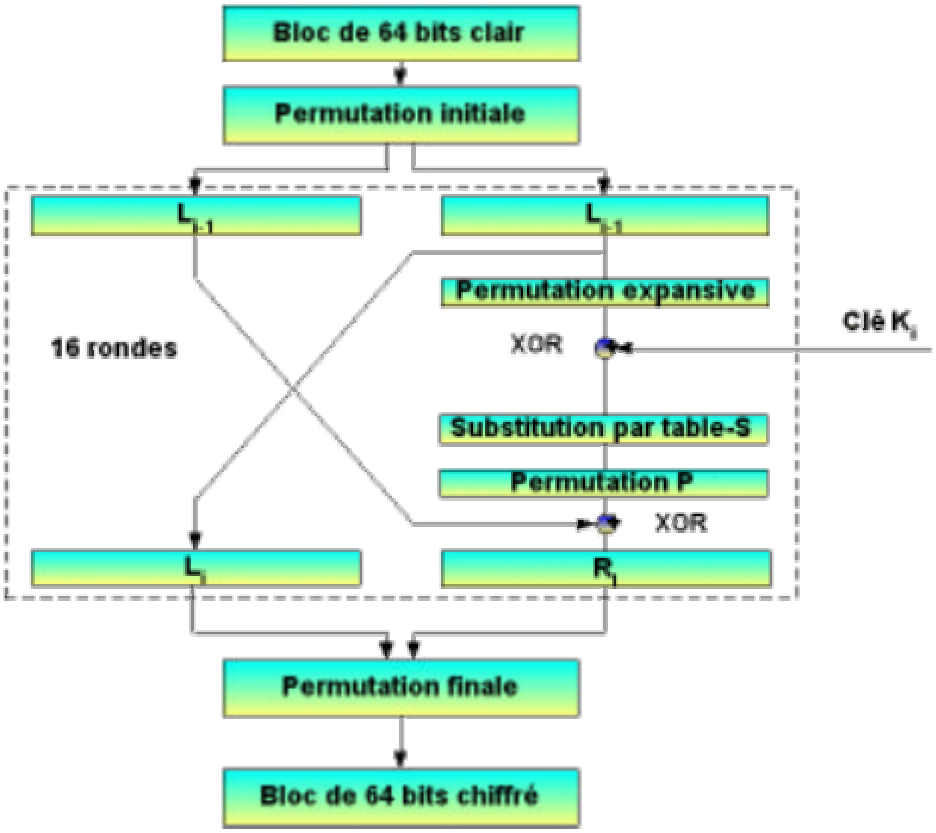


Figure 2.1. Schéma général de DES

Deux entreprises, Lexar Corp. et Bell Lab. examinèrent le fonctionnement des tables-S. Leurs analyses ne révélèrent aucune faiblesse, bien qu'ils trouvèrent des caractéristiques inexplicables en comparant les tables-S par rapport à des tables choisies aléatoirement.

- Lexar conclut « Certaines structures découvertes dans le DES ont sans doute été insérées pour renforcer le système contre certains types d'attaques. D'autres structures tendent à affaiblir ce système »

- Bell indiqua que les tables-S pouvaient avoir une brèche secrète.

Aujourd'hui, on a remarqué que ces tables-S rendaient la cryptanalyse différentielle plus difficile sur le DES mais tout de même possible.

Malgré cela, il n'y a pas d'attaques très évidentes ou très faciles contre le DES à ce jour, la plus simple étant l'attaque exhaustive, la plus efficace étant la cryptanalyse différentielle et linéaire. En 1981, Diffie et Hellman démontrèrent que retrouver la clé en 2 jours, par attaque exhaustive (qui essaie toutes les combinaisons de clés possibles) avec un ordinateur à architecture parallèle coûtait 50 millions de dollars. Ils considéraient aussi que jusqu'à 1990,

le DES ne serait plus sûr du tout. En 1984, les puces étaient capables de faire 256000 chiffrements de bloc par secondes. En 1987, c'était plus d'un million et en 1995, on trouvait dans le commerce des puces chiffrant 8 millions de blocs / s. En 1993, Michael Wiener a conçu les plans d'une machine coûtant 1 million de dollars, pouvant accomplir une attaque exhaustive (tester les 7.10^{16} de clés possibles) en 35 heures. La machine revient à 100 000 dollars, et casse n'importe quelle clé DES en 35 heures.

2.3. Le cahier des charges pour l'AES

En 1998, le NIST lance un appel au public afin de proposer un nouveau standard de cryptage, et demande que les algorithmes satisfassent au moins les cinq conditions suivantes :

1. Cryptosystème très robuste, par blocs et à clés symétriques pour utilisations gouvernementales et commerciales, pour le 21^{eme} siècle.

2. Plus efficace que le Triple DES, Le Triple DES (noté 3-DES) est une variante de DES, utilisant une clé de longueur triple (3*56 bits) par rapport au DES officiel.

3. Plus sécurisant que le Triple DES.

- Taille des clés : 128, 192, et 256 bits.
- Taille des blocs: 128 bits (autres tailles optionnelles).

4. Elaboré et évalué publiquement, en effet l'élaboration ainsi que l'évaluation des algorithmes en public est une marque importante de garantie contre d'éventuelles brèches secrètes introduites dans les algorithmes par leurs auteurs. Et aussi garantir la propriété intellectuelle libre dans le monde.

2.4. Les propositions pour l'AES

Les 15 candidats retenus lors de la $1^{ère}$ conférence AES sont :

LOKI97 Australian Defence Force Academy, L.Brown, J.Pieprzyk, J.Seberry, Australie.
RIJNDAEL J..Daemen, V. Rijmen.
CAST-256 Entrust Technologies, Inc, Canada.
DEAL Outerbridge, Knudsen, Canada.

FROG TecApro Internacional S.A, Costa Rica.

DFC Centre National pour la Recherche Scientifique (CNRS), France.

MAGENTA Deutsche Telekom AG, Allemagne.

E2 Nippon Telegraph and Telephone Corporation (NTT), Japon.

CRYPTON Future Systems, Corée.

HPC Univ. Arizona, Rich Schroeppel, USA.

MARS IBM, USA.

RC6 RSA Laboratories, USA

SAFER+ Cylink Corporation, USA.

TWOFISH B. Schneier, J.Kelsey, D.Whiting, D.Wagner, C.Hall, N.Ferguson.

SERPENT R. Anderson, Eli Biham, L. Knudsen

Ces 15 algorithmes sont les candidats de la première sélection (1^{ere} conférence AES) ayant commencé en Août 1998. 600 CD-Rom portant ces algorithmes ont été distribués dans plus de 50 pays du monde pour test et évaluation.

Après la 2eme conférence AES (Mars 1999), le NIST annonce les 5 finalistes :

MARS, RC6, RIJNDAEL, SERPENT, TWOFISH.

Et en Octobre 2000 : Lors de la 3eme conférence AES, Le vainqueur est RIJNDAEL.

2.5. La proposition retenue : RIJNDAEL

2.5.1. Présentation générale

C'est un Cryptosystème proposé par Joan Daemen et Vincent Rijmen, de l'Université Catholique de Louvain, Belgique. C'est un Système de chiffrement symétrique (à clef secrète), qui opère sur des blocs de texte de 128, 192 ou 256 bits, mais pour l'AES c'est 128 bits seulement. Il est utilisé avec des clés de 128, 192 ou 256 bits, en termes décimaux, ces différentes tailles de clés signifient concrètement qu'il y a :

3.4×10^{38} clés de 128-bit possibles

6.2×10^{57} clés de 192-bit possibles

1.1×10^{77} clés de 256-bit possibles

Comme pour la plupart des algorithmes de chiffrement par blocs, le processus de chiffrement de **RIJNDAEL** consiste a itérer une permutation paramétrée par une valeur

secrète, appelée sous-clef, qui change à chaque itération. Les différentes sous-clefs sont dérivées de la clef secrète par un algorithme de cadencement de clef. Pour une clef de 128 bits, **RIJNDAEL** effectue 10 itérations de la fonction décrite dans la figure 2, chacune des sous-clefs est également de 128 bits. La première itération est précédée d'un ou exclusif (xor) bit-a-bit entre le message clair et la sous-clef numéro 0; de même pour la dernière itération mais avec une légère différence par rapport aux itérations précédentes.

La fonction itérée se décompose en trois étapes, conformément aux principes fondamentaux de confusion et de diffusion. La première étape, dite de confusion, consiste a appliquer à chacun des 16 octets de l'entrée une même substitution BS. Cette fonction correspond à la fonction inverse dans le corps fini F_{256} plus une transformation non linéaire ; Ensuite, lors de la 2^{eme} étape (diffusion), on permute les bits du mot obtenu suivant une fonction P qui est également composée d'opérations de décalage et de mixage. En fin, on effectue un ou exclusif bit-à-bit entre le résultat et la sous-clef correspondante.

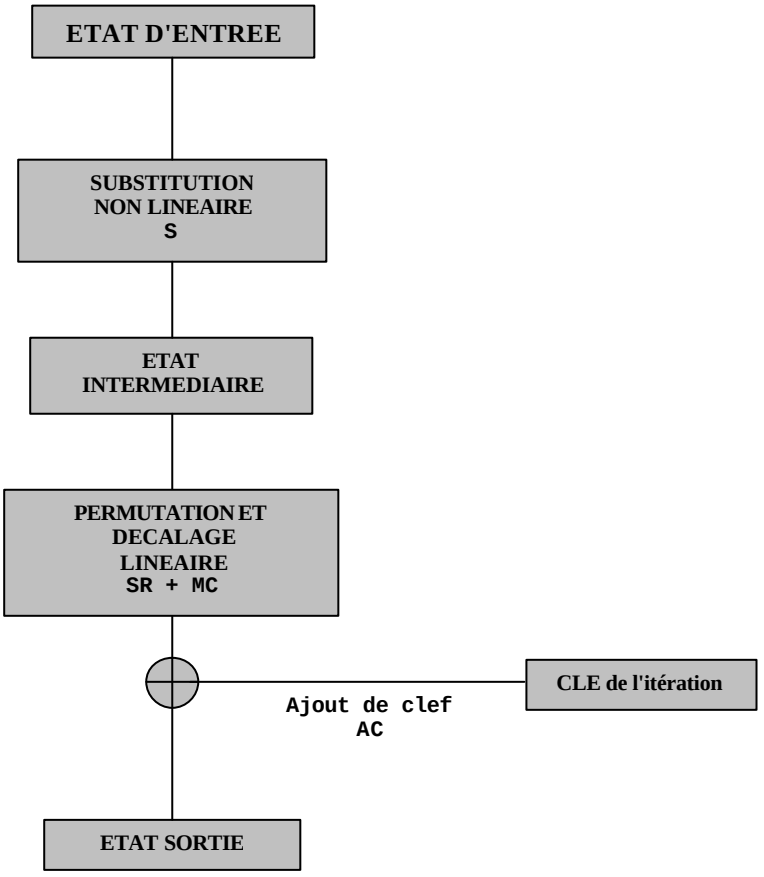


Figure 2.2. Une itération de RIJNDAEL

2.5.2. Préliminaires mathématiques

Plusieurs opérations dans Rijndael sont définies au niveau octet, donc c'est des éléments binaires de 8 bits, représentant des éléments dans le corps fini $GF(2^8)$. Tandis que d'autres opérations sont définies en termes de mots de 4-octets. Dans cette section, nous présentons les concepts mathématiques de base requis et utilisés dans Rijndael.

Le corps fini $GF(2^8)$

Les éléments d'un corps fini peuvent être représentés de différentes manières. En effet pour n'importe quelle puissance d'un nombre premier correspond un seul corps fini ; par conséquent toutes les représentations du corps fini $GF(2^8)$ sont isomorphes.
En dépit de cette équivalence, la représentation a un impact sur la complexité de l'implémentation. Pour Rijndael c'est la représentation polynomiale classique **[4].**

Un octet b est composé des bits $b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$ est considéré comme polynôme avec des coefficients dans {0,1} :

$$b_7x^7+ b_6x^6+ b_5x^5+ b_4x^4+ b_3x^3+ b_2x^2+ b_1x^1+ b_0$$

Par exemple 01010111 correspond au polynôme $x^6 + x^4 + x^2 + x + 1$ donc $b_0=1$, $b_1=1$, $b_2=1, b_3=0, b_4=1, b_5=0, b_6=1, b_7=0$,

Addition

Dans la représentation polynomiale, la somme de deux éléments est le polynôme avec les coefficients qui sont donnés par la somme modulo 2 (c'est a dire 1+1=0+0=0, 1+0=0+1=1) des coefficients des deux termes deux à deux. L'addition correspond aussi à l'opération ou exclusif (xor) bit a bit.

Exemple : $(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$

Multiplication

Dans la représentation polynomiale, la multiplication dans $GF(2^8)$ correspond à la multiplication des polynômes modulo un polynôme binaire irréductible de degré 8.
Un polynôme est irréductible s' il n'a aucun diviseur autres que 1 et lui-même. Dans Rijndael, ce polynôme est noté m(x) et donné par : $m(x)=x^8 + x^4 + x^3 + x + 1$
Exemple : $(x^6 + x^4 + x^2 + x + 1) (x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

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

On remarque que la multiplication est associative et admet un élément neutre e=1. Pour un polynôme b(x) de degré inférieur à 8, l'algorithme d'Euclide étendu peut être utilisé pour calculer les polynômes a(x) et c(x) qui vérifiant :

$$b(x)\,a(x) + m(x)\,c(x) = 1$$

donc $a(x) \cdot b(x) \bmod m(x) = 1$ ou $b^{-1}(x) = a(x) \bmod m(x)$.

2.5.3. Spécification technique

L'AES opère sur des blocs de 128 bits (texte en clair) qu'il transforme en blocs cryptés de 128 bits (texte chiffré) par une séquence de Nr itérations, à partir d'une clé de 128, 192 ou 256 bits. Suivant la taille de celle-ci, le nombre d'itérations diffère. Chaque itération opère sur des résultats intermédiaires.

2.5.3.1. L'état, la clef et le nombre de tours

Un état est un bloc de texte vu comme une matrice 4 x Nb d'octets, où Nb= taille du bloc / 32; donc pour l'AES Nb=128/32=4. D'autre part, la clef de chiffrement qui est un bloc de textes est vu comme une matrice de 4 lignes et Nk colonnes où Nk = longueur de la clef / 32.

Le nombre d'itérations noté Nr dépend des valeurs de Nb et Nk (voir Tableau 1)

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

Figure 2.3. Exemple d'état.

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

Figure 2.4. Exemple de clef de chiffrement.

Nr	Nb=4	Nb=6	Nb=8
Nk=4	10	12	14
Nk=6	12	12	14
Nk=8	14	14	14

Tableau 3. Nombre d'itérations Nr en fonction de Nk

2.5.3.2. Les transformations élémentaires

Chaque itération est composée de 4 transformations élémentaires (figure 2) :

- Substitution des octets "S".
- Décalage de lignes "SR".
- Mixage de colonnes "MC".
- Addition de clef "AC".

La dernière itération ne comporte pas de mixage de colonnes.

La substitution "S"

C'est une substitution non-lineaire d'octet qui opere sur chacun des octets d'un etat. La table de substitution (Table-S) est inversible. La table-S est construite par composition des deux transformations suivantes :

1- Calcul de l'inverse de l'octet $a_{i,j}$ dans le corps fini GF(2^8), soit $x = a_{i,j}^{-1}$.

2- Effectuer une transformation affine sur GF(2^8) de $x=(x_0, x_1,\ldots,x_7)$ comme suit:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

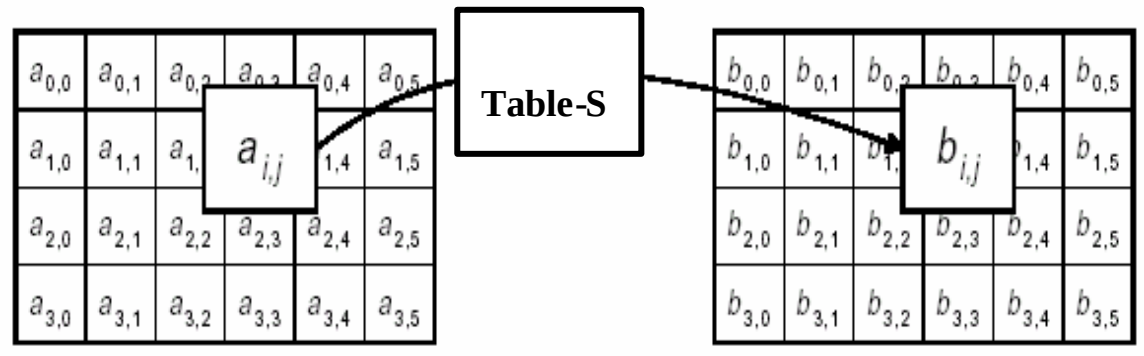


Figure 2.5. La substitution d'un octet dans un état

Décalage de ligne "SR"

C'est un décalage cyclique à gauche des lignes de l'état, pour la ligne 0 pas de décalage, la ligne 1 est décalée de C1 positions, la ligne 2 est décalée de C2 positions et la ligne 3 est décalée de C3 positions. C1, C2 et C3 dépendent de la longueur de blocs Nb, les différentes valeurs sont données dans le tableau 2.

Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Tableau 4. Nombre de positions de décalage

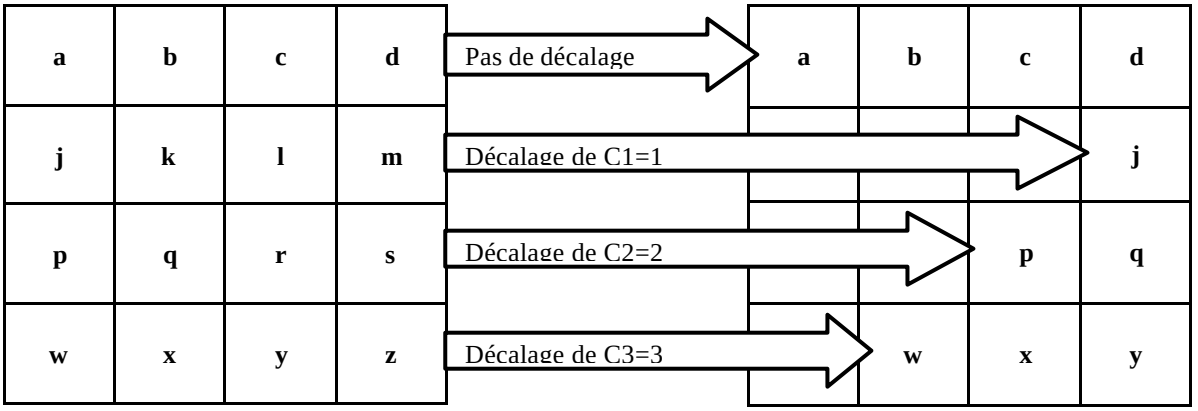


Figure 2.6. Décalage de lignes

Mixage de colonne "MC"

Dans l'opération de mixage chaque colonne de l'état est considérée comme un polynôme sur GF(2^8), ensuite il est multiplié parc un polynôme fixe c(x) modulo $x^4 + 1$; c(x) est donné par :

$$c(x) = \text{'03'}\ x^3 + \text{'01'}\ x^2 + \text{'01'}\ x + \text{'02'}$$

Ce polynôme est copremier à $x^4 + 1$ et donc inversible, ceci peut être écrit comme multiplication de matrice $b(x) = c(x) \otimes a(x)$,

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

L'application de cette opération pour toutes les colonnes de l'état est le mixage des colonnes. La figure 7 illustre l'effet du mixage de colonnes sur un état.

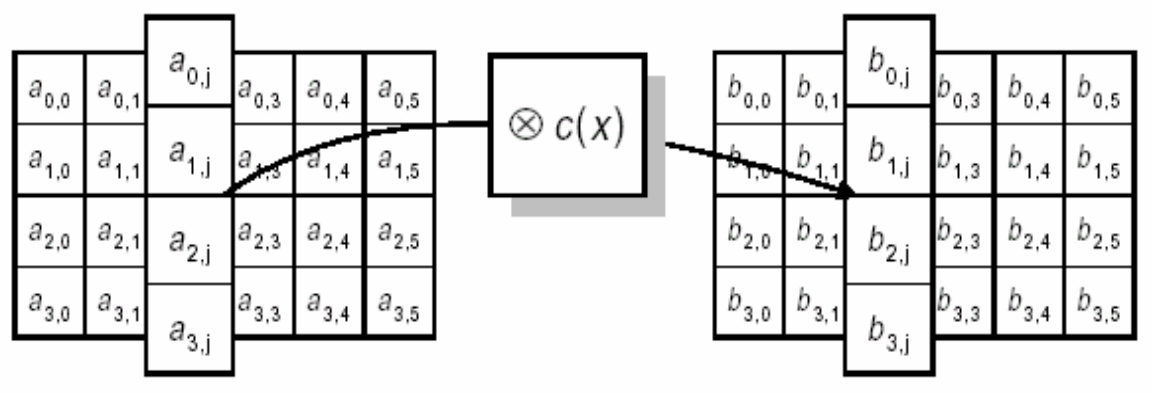


Figure 2.7. Opération de mixage sur les colonnes d'un état.

Addition de clef "AC"

Dans cette opération, une clef d'itération est appliquée à l'état par un simple ou exclusive (xor) bit par bit. La figure 8 illustre cette opération;

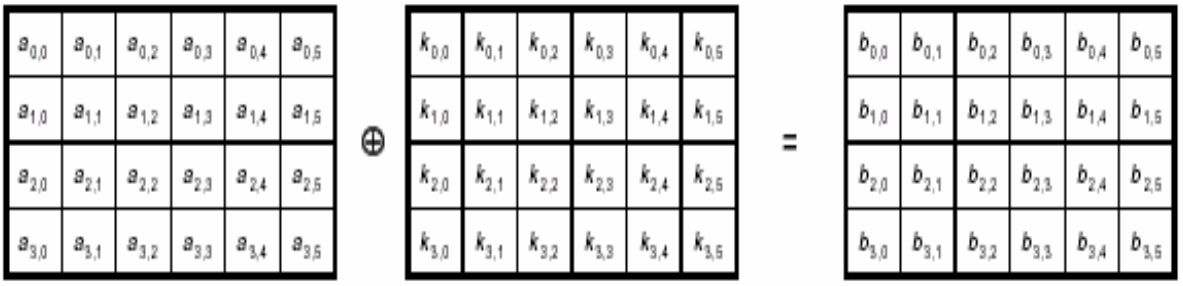


Figure 2.8. Addition de clef dans une itération

Remarque:

Les quatre transformations élémentaires utilisées dans le cryptosysteme RIJNDAEL : la substitution S, le décalage SR, le mixage MC et l'addition de clef AC, sont inversibles. Donc les transformations inverses S^{-1}, D^{-1}, M^{-1} et AC^{-1} existent et sont utilisées dans l'opération de déchiffrement.

2.5.3.3. Cadencement des clefs

La sous-clef de chiffrement d'une itération est un bloc de N mots de 32 bits chacun, et l'ensemble des sous-clefs de chiffrement pour R itération est vu comme un vecteur W[] de 4*(R+1) mots. Les N mots de la 1^{ere} sous clef W[0], …, W[N-1] sont initialisés directement par les N mots de la clef de chiffrement **[6].**

Soit *const(k)* une fonction constante sur 32 bits dépendant de k, elle est fixe pour k=1, 2, 3,…
Et soient : *f, g : {0, 1}^{32}? {0, 1}^{32}* des permutations non linéaires fixées dans la définition de RIJNDAEL. Alors pour i=N,…,4*(R+1)-1 le mot W[i] est défini récursivement comme suit :

Si (i mod N)=0
Alors W[i]=W[i-N] xor f(W[i-1]) xor const(i div N)
Sinon si ((N>6) et (i mod N)=4)
Alors W[i]=W[i-N] xor g(W[i-1])
Sinon W[i]=W[i-N] xor W[i-1]

Avec const(k)=(RC(k), 0, 0, 0) où RC(k)=x^{k-1} (c'est l'octet du polynôme x^{k-1})

Remarque : Si on connaît les deux mots W[i-1] et W[i-N] on peut alors déduire le mot W[i]. La même chose pour l'inverse, si on connaît W[i] et W[i-1] alors on peut déduire W[i-N]. Donc, n'importe quelle série de N mots consécutif W[k],…,W[k+N-1] de la clef étendu suffit pour générer la clef initiale. Cette propriété sera utile pour l'attaque algébrique (carré) qui sera développée dans la partie suivante **[4].**

2.5.3.4. Mode de fonctionnement du cryptosysteme

Le chiffrement consiste en 4 étapes principales :

1- Cadencement de clef;
2- Addition de clef initiale;
3- Nr - 1 itérations;
4- Itération finale (pas de mixage);

La figure 9 illustre le schéma général de RIJNDAEL.

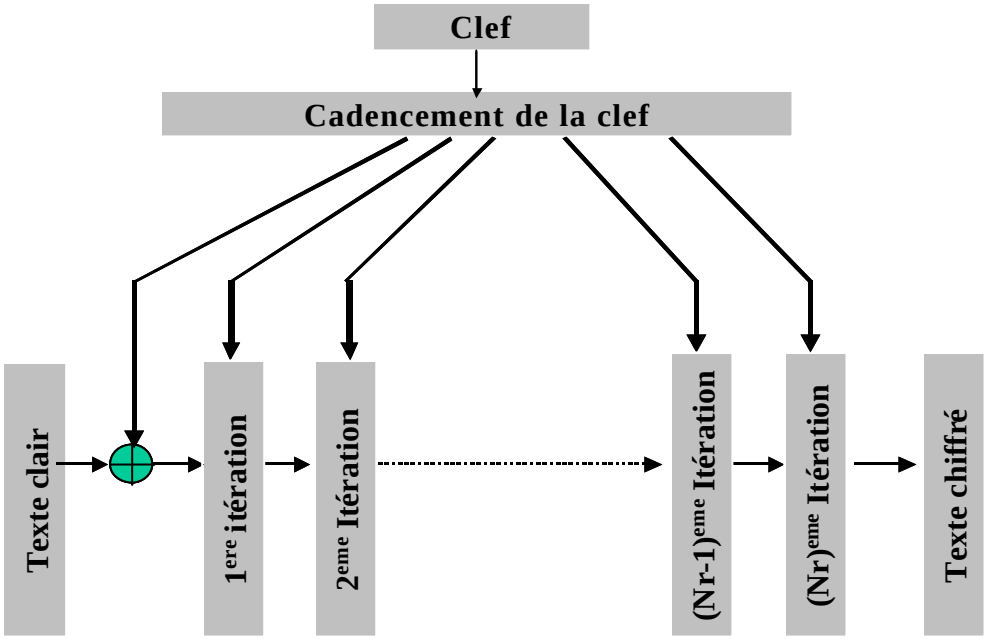


Figure 2.9. Schéma général de RIJNDAEL

2.5.4. Caractéristiques et points forts de l'AES

Le choix de cet algorithme répond à de nombreux critères plus généraux dont nous pouvons citer les suivants :

- sécurité par rapport à l'effort requis pour une éventuelle cryptanalyse.
- facilité de calcul : cela entraîne une grande rapidité de traitement.
- besoins en ressources et mémoire très faibles.
- flexibilité d'implémentation : cela inclut une grande variété de plateformes et d'applications ainsi que des tailles de clés et de blocs supplémentaires.
- hardware et software : il est possible d'implémenter l'AES aussi bien sous forme logicielle que matérielle.
- simplicité : le design de l'AES est relativement simple.

Si l'on se réfère à ces critères, on voit que l'AES est également un candidat particulièrement approprié pour les implémentations embarquées qui suivent des règles beaucoup plus strictes en matière de ressources, puissance de calcul, taille mémoire, etc...

Contrairement au DES, l'AES est résistant aux attaques de type différentielle et linéaire.

2.6. Conclusion

L'AES est un cryptosysteme symétrique composé d'opérations algébriques simples sur des octets dans le corps fini GF(2^8), cet aspect le rend très rapide, soit pour les architectures des processeurs des cartes à puce ou bien pour les implémentations logicielles. D'autre part les tailles des clefs proposées le rend plus sûr que ses prédécesseurs DES et 3-DES ; de plus il présente une plus grande résistance aux attaques connues. Mais cela n'empêche pas qu'il y ait des attaques applicables sur lui, mais à un nombre d'itérations limité, c'est ce qu'on va voir dans la prochaine partie.

Partie III. Cryptanalyse de l'AES

3.1. Introduction

Depuis la première publication de RIJNDAEL comme étant un candidat à l'AES, plusieurs attaques ont été proposées, soit pour l'évaluation du cryptosystème avant sa mise en service, ou bien pour le casser pour d'autres fins que l'évaluation et l'amélioration spécialement après l'officialisation par le NIST de RIJNDAEL comment étant le nouveau standard de cryptage. Parmi les attaques proposées il y a celles qui se basent sur des modèles mathématiques formels et ceux qui se basent sur des failles dans les implémentations.

3.2. Attaque basée sur les codes de répétition des textes en clair dépendants

Cette attaque a été présentée dans **[7]** par Eric FILIOL en janvier 2003, donc après la mise en service de l'AES ; dans cette partie on présente les grandes lignes de cette attaque ainsi que les premiers résultats de celle-ci.

3.2.1. Théorie et notation

- **Les Codes de répétition**

Considérons un canal de transmission symétrique binaire (BSC) de paramètre p utilisé pour transmettre des messages sur un alphabet binaire. En d'autres termes, si un émetteur envoie un bit b_t alors le bit $b_t' = b_t + e_t$ sera reçu réellement avec une probabilité p (probabilité d'erreur du canal). Pour remédier aux erreurs de transmission on utilise des codes correcteur d'erreurs et en particulier les codes linéaires. Un code linéaire binaire [n, k, d] est un sous espace vectoriel de dimension k du corps fini F_2^n = {ensemble des mots de n bits}. Sa distance minimal d est la distance minimale de Hamming de tous les code-mots non nuls (c'est-à-dire les mots avec n bits). En d'autres termes $d=\min_x\{wt(x)\}$ avec x ? F_2^n et wt(x) c'est le nombre de positions non nulles dans $x=(x_1,\ldots, x_n)$. Alors le nombre d'erreurs sur un code-mot qui peuvent être corrigées par un code de distance minimale d est donné par (d-1)/2.

Un n-code de répétition, sur un ensemble de deux symboles, est un code linéaire [n,1,n] et consiste en deux codes-mots, chacun d'eux est composé de n symboles identiques.

D'autre part, quand q > p où q=1-p, la probabilité maximale de décodage (MLD) revient à trouver dans le vecteur reçu quel est le symbole répété le plus. Le vecteur sera décodé comme 0 si sa distance de Hamming au vecteur nul est inférieure à sa distance au vecteur (1,1,1,...,1), sinon il est décodé comme 1.

Proposition : Soit n = 2s+1. Alors le n-code de répétition corrige au plus s erreurs. Sa probabilité d'erreur (erreur de décodage résiduel) est donné par:

$$P_{err} = \sum_{i=s+1}^{n} \binom{n}{i} p^i \cdot q^{n-i}. \qquad (1)$$

Finalement la probabilité d'un décodage réussi est donnée par :

$$P_{succ} = 1 - P_{err}$$

- **Chiffrements par bloc et cryptanalyse linéaire**

Un chiffrement par bloc qui opère sur des blocs de textes clair P_i de m bits avec une clé secrète K de n bits (Noté (m,n)-bloc chiffre) est une projection de F_2^m x F_2^n dans F_2^m. Pour une clef choisie K, la projection résultante est une permutation sur F_2^m. Donc un chiffrement en bloc est un ensemble de 2^n permutations sur F_2^m. Noter qu'il représente un très petit sous-ensemble de toutes ces permutations ($(2^m)!$ dans le total).

La cryptanalyse linéaire est une attaque à texte clair connu dans laquelle un très grand nombre de paires (texte-clair, texte-chiffré) sont utilisées pour déterminer la valeur d'un sous-ensemble de bits de la clef, et par conséquence réduire considérablement la partie de la recherche exhaustive. Une condition pour appliquer la cryptanalyse linéaire à un tel schéma en bloc est de trouver des expressions linéaires probabilistes entre tout bloc de texte clair Pi, tout bloc de texte chiffré Ci et toute clef K de la forme :

$$<P_i, u> xor <C_i, w> \tilde{} <K, v> \qquad (2)$$

Où <. , .> c'est le produit scalaire sur F_2^m. Si cette équation est vérifiée avec une probabilité p ? 1/2, alors, en vérifiant le côté gauche de l'équation (2) pour un grand nombre N de paires (texte-clair, texte-chiffré), le côté droit de cette équation peut être estimé par un

simple décodage de la probabilité maximale, et dans ce cas une seule information bit sur la clef est obtenue. Cette cryptanalyse est efficace si l'écart | p - ½ | est assez grand.

3.2.2. La cryptanalyse par les codes de répétition des chiffrements par bloc

- **Les chiffrements par bloc et les codes de répétition**

Considérons une propriété donnée T et soit $P_E[T]$ la probabilité que T soit satisfaite sur l'ensemble E. Alors un chiffrement par bloc peut être cassé si nous avons :

$$P_{F_2^{m+n}}[T] \; ? \; 1/2.$$

Chaque clé K dans l'espace des clés $K=F_2^n$ sélectionne une permutation correspondante sur F_2^m. Donc K peut être trouvé si $P_{F_2^m}[T_K] \; ? \; 1/2$, où T_K est la propriété T relative à la clé K. Alors nous pouvons disposer d'une attaque si nous pouvons trouver une telle propriété vérifiée pour tout K € K (notée T_K). Par exemple dans le cas de la cryptanalyse linéaire, T_K est une équation probabiliste linéaire particulière.

Considérons maintenant l'espace des textes en clair $P = F_2^m$ et une partition $(P_i)_{i=2^k}$ de P pour k € N. On suppose que $|P_i| = 2^{m-k}$ pour tout i. Maintenant supposons qu'il existe P_i tel que $P_{P_i}[T_K] = p_i \; ? \; 1/2$. Puisque la clef de cryptage K ? K, reste la même pour tous les blocs du texte en clair, nous pouvons comparer le processus de cryptage à un canal symétrique binaire (BSC) de paramètre p_i où le bruit est produit par les blocs du texte en clair de P_i. La version bruitée T_K' de T_K est une fonction f(C) des blocs de textes chiffrés C. En d'autres termes crypter N blocs de texte clair P ?P_i peut être défini comme étant la transmission de T_K au moyen de N-code de répétition à travers un canal symétrique binaire de paramètre p_i. Donc on peut dire que sur C_i nous avons $P[T_K = T_K'] = 1- p_i$. Où C_i est un sous ensemble de C.

Le but du concepteur d'un cryptosystème est d'obtenir un ensemble de permutations sur C tel que aucune propriété T ne divulgue de l'information sur la clef. Mais la situation est vraisemblablement très différente quand on considère une restriction à un sous-ensemble C_i de C. Si nous avons :

$$Pc[T]=S\,Pc_i[T].\,P[C_i]) = 1/2, \text{ pour } i=0..2^k$$

Cependant nous pouvons avoir beaucoup de $Pc_i[T]$ différents de 1/2 (il suffit que $Si(e_i)= S(p_i-1/2)=0$). C'est dû au fait que le nombre de permutations sur C représentées réellement par

le cryptosystème est nettement négligeable, comparé au nombre total de permutations sur le même espace du texte en clair.

- **Description de l'attaque**

Avec les définitions données dans la section précédente, nous pouvons décrire maintenant la cryptanalyse basée sur les codes de répétition de textes en clair dépendants. Notons que l'indépendance locale du texte-clair (dû à la restriction à un sous-ensemble particulier Ci de C) nous permet de concevoir une attaque à texte-chiffré seulement. Nous présentons un algorithme A1 qui utilise un seul code de répétition.

Algorithme A1

***Entrée**: N blocs de texte chiffré Cj cryptés par une clé K a partir des texte clair P_j € C_i (1 = j = N) et une information probabiliste T_K tel que T_K ˜ f(Cj) à une probabilité p_i.*

***Sortie**: Valeur exacte T(K) pour la clef.*

1. *initialisation du compteur ct ? 0.*
2. *pour chacun des N blocs du texte-chiffré C_j.*
   - *(a) calculez $f(C_j)$.*
   - *(b) si $f(C_j)=1$ alors ct ? ct + 1.*
3. *fin pour*
4. *si ct = (N+1)/2 alors T(K)=1 sinon T(K)=0.*

La complexité de l'algorithme A1 est facile à évaluer ; en effet il exécute seulement N évaluations de *f*. Donc sa complexité est O(N). Puisque N est la longueur du code de répétition, donc il dépend seulement de p_i et de p_{succ} la probabilité d'estimation réussite de T(K).

Remarque :

Pour obtenir une probabilité de succès aussi grande que possible, un deuxième algorithme A2 a été proposé. Il utilise des codes de répétition concaténés. Le principe est d'utiliser deux codes, donc la combinaison forme un nouveau canal (appelé un super canal). Le but est d'améliorer la capacité de correction du 1er code par usage d'un deuxième code.

3.2.3. Critère de résistance contre l'attaque

L'attaque est possible si, et seulement si, il existe un sous-ensemble C_i de C tel que $P_{C_i}[T_K]$?1/2 pour des propriétés T. Cela nous permet de formuler le critère de résistance suivant contre l'attaque.

Proposition : Soit S un chiffrement par bloc (m, n) et considérons une propriété T sur les bits de la clé par rapport aux bits du texte chiffré. S est à l'abri de l'attaque par rapport à propriété T, si et seulement si, pour tout j ? N la partition (C_1, C_2,..., C_j) de C vérifie :

$$V\ k = j, P_{C_k}[T]=1/2.$$

Le travail du cryptanlyste est de trouver une propriété exploitable T et un sous-ensemble particulier de blocs de texte clair significatifs, pour mener l'attaque sur S. Du point de vue des cryptographes, l'idée peut être plus difficile. Cette difficulté est résumée avec les trois problèmes ouverts suivants :

1. **Le problème d'immunité.** Etant donné une propriété T, est-il possible de concevoir un système S qui est immunisé contre l'attaque par rapport à T ?
2. **Le problème de la trappe.** Soit C_i un sous ensemble de C, est-ce possible de concevoir un cryptosystème S tel que $P_{C_i}[T]$? 1/2 pour T (la trappe) ? Le problème 2 signifie qu'il doit être possible de cacher une trappe T dans le cryptosystème S.
3. **La faisabilité de l'attaque.** Soit S un cryptosystème donné et C_i un sous-ensemble de texte clair, est il possible de trouver quelques propriétés T convenable pour l'attaque sur S.

Conjecture : Il existe toujours une propriété T pour laquelle un système de chiffrement par bloc S n'est pas immunisé contre l'attaque proposée. Si c'est vrai, cela veut dire que les chiffrements par blocs sont des systèmes incertains (non sûrs).

Le problème 3 est clairement le plus important à résoudre, du point de vue des cryptanalystes. Pour essayer de le résoudre, le package combinatoire, statistique CoHS[1] (Combinatorics over Huge Sets) a été utilisé ; ce package est développé pour trouver des propriétés structurelles dans des ensembles complexes de très grande dimension. C'est un package non public qui est encore en développement. Avec CoHS, le chiffrement par blocs

(m, n) est vu comme une famille de 2^{m+n} blocs de m bits où chaque bloc est répété 2^n fois. Pour l'attaque proposée, il faut trouver des structures particulières entre des blocs pour des sous-ensembles donnés. Alors les structures identifiées peuvent être changées éventuellement en propriétés statistiques. L'avantage principal de ce package est qu'il n'exige pas l'utilisation de la totalité de la famille mais seulement un sous-ensemble d'une dimension raisonnable. Les aspects théoriques de CoHS devraient être publiés dans le futur proche dès que son développement est achevé et le processus d'obtention du brevet est complété **[7].**

3.2.4. La cryptanalyse de l'AES

- **Les codes de répétition pour l'AES**

Le problème principal est alors de trouver une propriété convenable T_K qui présente une faille (brèche) pour un sous-ensemble C_i de C. L'idée été d'ajuster CoHS pour travailler sur la base de la cryptanalyse linéaire. Dans ce cas, nous avons besoin d'avoir une approximation de la forme:

$$<P, u> \text{ xor } <C, w> \tilde{} <K, v> \quad \textit{a une probabilité q}$$

où u, v et w sont des masques utilisés pour la sélection de certains bits de la clef. Si on réussit à trouver un sous-ensemble C_i pour lequel il existe v', w' appartenant à $F_2^n \times F_2^m$ tel que :

$$<C, w'> \tilde{} <K, v'> \quad \textit{a une probabilité q'} \qquad (3)$$

avec q' ? 1/2 alors nous aurons une propriété convenable T_K utilisée comme un code de répétition.

Le but est de trouver des paires de valeurs du masque (v', w') vérifiant l'équation de la forme (3). Le package CoHS a été exécuté pendant quatre mois sur quatre ATHLON XP2000 + PC avec 512Mo RAM et 80Go DD. Les paramètres ont été ajustés pour obtenir des valeurs w' de poids aussi bas que possible pour réduire le temps de calcul et produire les premiers résultats. Deux équations du masque 0xEFEFEFEF…ont été produites et confirmées comme convenables avec 100 cryptanalyses ; en fait 27 équations ont été produites, chacune d'elles à une probabilité de cryptanalyse avec succès qui varie de 0.68 à 0.88. Dans cette attaque les

textes clairs utilisés sont en anglais avec un codage ASCII, et toutes les valeurs du masque sont écrits en hexadécimal **[7].**

3.2.5. Résultats de l'attaque

Des probabilités trouvées, et de la Formule (1) nous obtenons le paramètre de répétition convenable N d'où le nombre de blocs de texte-chiffré nécessaire. Le paramètre N est le résultat de l'exécution de l'attaque avec l'Algorithme A1:

$N=2500100001=2^{31}$ et $P_{succ} = 0.7875$.

L'attaque décrite dans algorithme A1 a été implémentée pour 100 différentes, clefs choisies aléatoirement. Le texte-clair a été généré aléatoirement aussi, et le bit a été sélectionné selon la valeur du masque qui correspond à langue anglaise avec le codage ASCII. Chaque expérience a pris 7 heures sur quatre ATHALON XP 2000+PC. Un temps important a été nécessaire pour la génération du texte-clair.

En résumé, dans cette cryptanalyse on utilise N codes de répétition où N est précisément le nombre de blocs de texte chiffré dont on a besoin. Les résultats prévus expérimentalement sont confirmés avec 100 cryptanalyses effectives. L'attaque est parvenue à récupérer deux bits de la clef avec seulement 2^{31} blocs de texte chiffré et une probabilité de succès de 0,68.

3.3. Attaque par le timing

Cette attaque a été développée par F. KOEUNE, J.J. QUISQUATER **[5]**, et présentée dans la 2eme conférence AES (1999). Son but principal est la prévention contre les mauvaises implémentations de RIJNDAEL.

3.3.1. Introduction

Les recherches récentes dans le domaine de la cryptanalyse sont concentrées sur des attaques qui visent les réalisations physiques du cryptosystème. C'est les attaques par "canaux cachés" qui prennent en compte l'existence d'un canal supplémentaire par lequel l'information peut fuir. Des exemples de tels canaux sont le temps d'exécution, l'observation frauduleuse d'un bit tout au long du calcul effectué. C'est des attaques très efficaces comparativement avec les attaques " classiques " qui se basent sur la recherche des failles du cryptosystème en l'observant comme un objet mathématique. Dans cette section, on va présenter une attaque qui se base sur le temps d'exécution de certaines implémentations de L'AES.

3.3.2. Description du model vulnérable

Dans l'AES, le bon choix du polynôme irréductible dans le mixage colonne (voir le paragraphe 2.5.3.2. de la 2eme partie) et de la matrice correspondante, rend l'implémentation de la procédure du mixage de colonnes très simple ; en effet on peut remarquer que '03'='02'+'01' donc la seule multiplication qui doit être exécutée est la multiplication par '02'. D'autre part la multiplication par '02' peut être implémentée par les deux étapes suivantes :

- Un décalage d'une position à gauche;
- Dans le cas de retenue on fait un XOR du résultat avec '1B';

Dans le cas d'une mauvaise implémentation l'opération de multiplication par '02' ne sera pas exécutée en un temps constant, elle sera plus longue dans le cas où il y a une retenue.

Idée de base

Soit le 1er octet d'un texte clair, cet octet subit des modifications durant le processus de cryptage :

- Un XOR avec un octet R1 de la clef de l'itération.
- Une substitution S, à partir de la table de substitution.

- Ensuite l'octet sera décalé à une position sans être modifié.
- Enfin, l'opération de mixage, durant cette opération, l'octet sera multiplié par '02'.

Si on est capable de calculer le temps de calcul de la multiplication, on peut déduire le premier bit de la clef de l'itération.

3.3.3. Description de l'attaque

On commence par la construction d'une matrice qui contient pour chaque cas du premier octet de la clef et pour N valeurs d'octets possibles du texte clair, une valeur qui indique : est ce que la multiplication requiert un XOR additionnel ou pas, la taille de cette table est 256 x N.

Pour i=0 jusqu'à 255
  Pour j=1 jusqu'à N
    T[i,j]=1 si le 1er S(i XOR j)=1 /// S() c'est l'opération de substitution
    T[i,j]=0 sinon
  Fin pour j
Fin pour i

Chaque ligne i correspond à une valeur possible de R1 (le 1er octet de la clef d'itération), tandis que chaque colonne j correspond à une valeur du 1er octet du texte clair.
De la même manière on construit N ensembles de M messages ; c'est l'échantillon où le 1er octet de chaque message de l'ensemble S_i est égal à i et les autres octets sont aléatoires. Donc pour chaque message de S_i la multiplication est exactement la même.
Maintenant on chiffre ces messages et on mesure les temps de calcul.
Si M est assez grand, on peut alors prévoir la moyenne du temps pour S_i. De cette manière on a construit un oracle qui peut déterminer avec une probabilité d'erreur pour i dans [0..N-1] si le 1er bit de l'octet *S(i XOR R1)* est à 1.

Pour déterminer R1 il suffit de comparer l'oracle à la table T : la ligne qui reflète mieux la prédiction de l'oracle doit correspondre à la bonne valeur de R1. D'autres octets de la clef de la 1^{ere} itération peuvent être trouvés en utilisant les autres octets du texte clair.

Résultats pratique

Les expérimentations montrent que pour chaque octet de la clef, on a utilisé 3000 échantillons pour trouver la clef complète, qui a été trouvée avec une très grande probabilité et un coût négligeable, donc dans cette attaque pour une clef de 128 bits on utilise 48000 échantillons.

3.4. Attaque algébrique (Carré)

Cette attaque est la première qui a été proposée contre RIJNDAEL à 4 itérations par les concepteurs du cryptosysteme **[4]**, mais elle a été améliorée par d'autres cryptanalystes notamment S.LUCKS **[6],** pour augmenter le nombre d'itérations vulnérables.

C'est une attaque à texte clair choisi. Pour la décrire on a besoin d'introduire la notion de "P-ensemble" : c'est un ensemble de 256 états qui sont tous différents dans quelques octets, appelés octets actifs, et tous égaux dans les autres octets (octets passifs), autrement dit dans un P-ensemble on a toujours :

$A_{i,j}$? $B_{i,j}$ si l'octet de la position (i,j) est actif; et

$A_{i,j} = B_{i,j}$ si l'octet de la position (i,j) est passif.

Un P-ensemble avec exactement k octets actifs est noté : P^k-ensemble.
Soit P_i l'ensemble de 256 états qui représente le résultat de la i^{eme} itération.

L'attaquant choisi un P^1-ensemble, qui est un texte clair avec un seul octet actif, On remarque que :

- P_1 est un P^4-ensemble car ils ont tous 4 octets actifs dans la même colonne;
- P_2 est un P^{16}-ensemble;
- P_3 est improbable qu'il soit un P-ensemble, mais tous les octets de P_3 sont pondérés c'est a dire, pour tous (i, j) ? $\{0, 1, 2, 3\}^2$ on a $XOR(A_{i,j})=0$, pour A ? P_3.

On considère maintenant le cas P_4, c'est a dire la 4^{eme} itération, la pondération des octets de P_3 peut être exploitée pour trouver la clef K^4 de la 4^{eme} itération.
On utilise comme alias pour la r^{eme} itération de la clef K^r la valeur L^r définit comme suit :

$L^r=SR^{-1}(MC^{-1}(K^r))$. Où SR^{-1} et MC^{-1} c'est les transformations inverses

L'attaque définit un ensemble Q_4 entre P_3 et P_4 comment suit :

1. *Pour tous X de P_4*
   $Y=MC^{-1}(X)$
   $Z=SR^{-1}(Y)$
   Noté par Q_4 l'ensemble Z de 2^8 états
2. *Pour tous (i, j) dans $\{0, 1, 2, 3\}^2$*
   Pour tous a dans $\{0, 1\}^8$
   $b(a)=XOR\ (S^{-1}(Z_{i,j}\ xor\ a))$ avec Z élément de Q_4
   si b(a) ? 0 Alors $(L_{i,j})^4$? a

En résumé, on inverse l'itération 4 étape par étape : inverser le mixage des colonnes; inverser le décalage de lignes; l'ajout de l'octet $L_i^4{}_{,j}$ de la clef et enfin inverser la substitution. Par conséquent si l'octet a = $L_i^4{}_{,j}$ *alors l'ensemble* $S^{-1}(Z_{i,j}$ *xor a)* est pondéré i.e b(a)=0. De cette manière on peut facilement construire un nombre candidat pour L^4 ($<2^{16}$), ainsi chaque candidat correspond à un unique choix pour le chiffrement AES à 128 bits. Pour trouver la clef, on doit choisir un 2^{eme} P^1-ensemble de texte clair, ou bien utiliser une recherche exhaustive sur toutes les clefs candidates mais avec les mêmes paires (texte clair / texte chiffré) (2^8 paires).

Cette attaque a été améliorée dans **[6]** pour augmenter le nombre d'itérations vulnérables. On utilise le même principe mais avec deux itérations de plus, une au début et une à la fin et dans ce cas on utilise 2^{32} textes claires, mais le temps d'exécution augmente d'une manière considérable, il devient 2^{208} U (où U est le temps d'exécution d'une opération élémentaire). Cette extension est efficace pour l'AES avec les clef 192 et 256 bits.

Cette même attaque a été améliorée une 2^{eme} fois dans **[8]** pour atteindre 9 itérations vulnérables. Dans ce cas, on utilise 2^{77} textes clairs, et on a besoin 2^{224} U (opérations élémentaires).
Dans les deux cas, on peut constater que la méthode est impraticable, mais elle ouvre la porte a l'utilisation d'une combinaison avec d'autres attaques pour trouver une partie de la clef.

3.5. Etude comparative

Dans la 3^{eme} partie on a présenté trois méthodes de cryptanalyse différentes, appliquées sur l'AES, deux basées sur des modèles statistiques formels et la troisième basée sur une méthode d'observation des canaux cachés ; dans notre cas c'est le temps d'exécution.

Les trois attaques sont applicables sur des variétés différentes de l'AES, soit en nombre d'itérations ou bien la taille de la clef, et même le type d'implémentation logicielle, en effet l'attaque basée sur les codes de répétition des textes clairs dépendant qui est une attaque basée sur un modèle statistique probabiliste, est applicable sur l'AES avec une clef de 128 bits et un nombre d'itérations complet (10 itérations) ; sont but est d'identifier une partie de la clef (dans notre cas deux bits), contrairement à la deuxième méthode (Attaque algébrique) qui s'applique au trois variantes de l'AES 128, 192 et 256 bits, mais avec un nombre d'itérations limité, au maximum 9 itérations, et le but principal de cette attaque est de trouver la clef

entière. Tandis que la troisième attaque (Attaque par le timing) qui se base sur l'observation du temps d'exécution de certaines versions de lAES, où l'implémentation sur machine n'est pas bien étudiée, mais elle est applicable à toutes les variantes de l'AES 128, 192, 256 bits et avec le nombre d'itérations complet, son but est de trouver la clef entière.

L'attaque basée sur les codes de répétition des textes clairs dépendant est une attaque a texte clair connue, qui se base sur un modèle mathématique probabiliste formel, qui s'inspire de la cryptanalyse linéaire où on essaie à modéliser le cryptosystème par un modèle mathématique. Dans cette attaque, on utilise un dispositif matériel-logiciel composé d'un réseau informatique et un logiciel de calcul probabiliste pour les grands ensembles (CoHS) pour trouver le modèle adéquat, le résultat est exécuté par la suite dans un algorithme polynomial pour identifier une partie des bits de la clef. Par contre, les deux autres attaques peuvent s'exécuter sur des machines isolées (PC), comme on peut les exécuter dans un réseau informatique, et là on a pas besoin d'autre logiciel externe.

On constate que l'algorithme utilisé dans l'attaque basée sur les codes de répétition des textes clairs dépendant est polynomial, mais il dépend du résultat de l'exécution du package CoHS qui est probabiliste, c'est a dire le résultat n'est pas garanti et il dépend des masques initialisés au début, donc le choix des masques est très important dans cette attaque. Tandis que dans l'attaque algébrique qui est une attaque par texte clair choisi, l'algorithme est polynomial pour l'AES à 4 itérations, mais pour un nombre d'itérations supérieures à 4 il devient exponentiel et le temps d'exécution augmente d'une manière très considérable.

Mais dans le cas de l'attaque par le timing qui est une attaque par texte clair choisi, l'algorithme est polynomial et le nombre de textes clairs utilisés est relativement petit contrairement à l'attaque algébrique où le nombre de textes clairs utilisés augmente d'une manière exponentielle pour un nombre d'itérations supérieur à 4, et dans le cas de l'attaque basée sur les codes de répétition des textes clairs dépendants le nombre de textes clairs utilisés est de l'ordre de 2^{30} pour identifier 2 bits, (2^{30} c'est à la portée des moyens de calcul disponibles actuellement). En effet le nombre de textes clairs utilisés désigne l'espace mémoire nécessaire pour les attaques.

Attaque	Tailles des clefs (bits)	Nombre d'itérations vulnérables	Dispositif utilisé	Espace mémoire	Algorithme
Attaque 1	128	Complet (10)	Réseau + CoHS	2^{30}	polynomiale
Attaque 2	128	4	PC	2^{16}	polynomiale
	192	9	impraticable	2^{77}	exponentielle
	256	9	impraticable	2^{77}	exponentielle
Attaque 3	128, 192, 256	complet	PC	/	polynomiale

Tableau 5. Comparaison des trois attaques

Attaque 1 : L'attaque basée sur les codes de répétitions des textes clairs dépendants.

Attaque 2 : L'attaque algébrique.

Attaque 3 : L'attaque par le timing.

En conclusion, on peut dire que les trois attaques présentent des points forts et des points faibles ; en effet l'attaque algébrique est basée sur une faille dans le cadencement des clefs, mais elle est compensée par l'augmentation du nombre d'itérations, donc plus le nombre d'itérations augmente cette attaque devient impraticable. Pour l'attaque par le timing elle n'est pas opérationnelle sur toutes les implémentations de l'AES, mais son principe est très simple et elle traite toutes les versions de l'AES (128, 192, 256 bits). Tandis que l'attaque basée sur les codes de répétition des textes clairs dépendants, qui semble être la plus prometteuse, son principal handicap, d'une part est l'utilisation du package CoHS et sa disponibilité (problème de brevet), et d'autre part le nombre de bits identifies est limités. La question qui se pose ; peut on améliorer cette attaque pour identifier plus de deux bits ?

Dans ce contexte on propose le développement d'un environnement d'attaque parallèle coopératif qui prendra en charge les trois attaques proposées, cet environnement est composé de plusieurs modules interactifs pour l'exécution des attaques dans des sites distants et en parallèle. Ces modules sont administrés par un module administrateur piloté par l'utilisateur, le rôle principal du module administrateur est de filtrer les données introduites par l'utilisateur et décider quel type d'attaque utiliser et cela d'une manière interactive avec l'utilisateur et par la suite affecter les taches aux autres modules et désigner le ou les sites d'exécution.

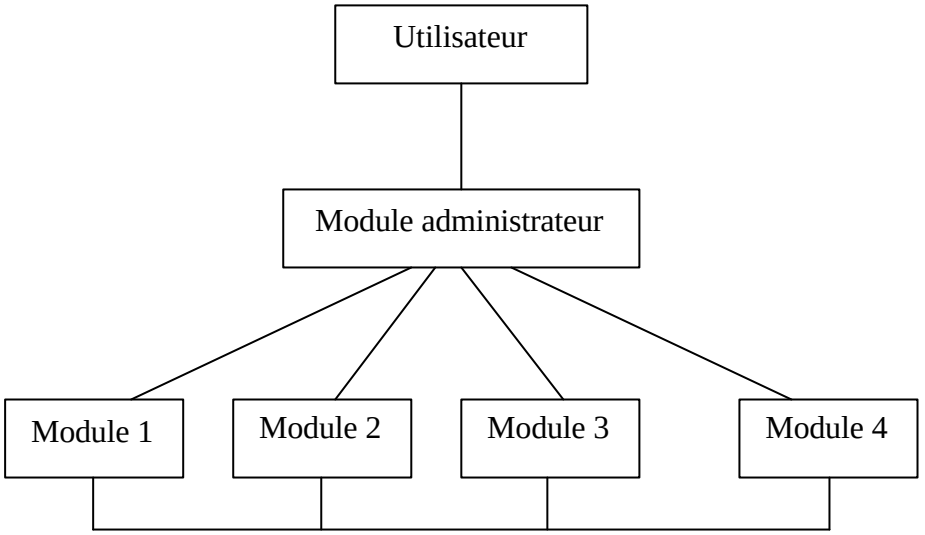


Figure 3.1. Architecture de l'environnement d'attaque.

Dans la figure 3.1 les flèches indiquent le sens de l'interactivité, ou bien le sens de circulation de l'information, on remarque que le module administrateur communique dans les deux sens avec le reste des modules, tandis que le module 1 communique avec les trois autres modules dans un seul sens.
Dans cette architecture le Module 1 s'occupe essentiellement de l'allocation des ressources matérielles : mémoire, espace de stockage, processeurs..., Tandis que chacun des trois autres modules s'occupe d'une attaque.

Remarque : on peut rajouter un autre module qui se charge de l'étude des corrélations possibles dans la table de substitution de l'AES S-box ou bien dans les clefs proposées a l'étude.

Le module administrateur et le module 1 sont liés directement à l'utilisateur par le site d'exécution utilisé par l'utilisateur, mais ce n'est pas le cas pour les autres modules, qui peuvent s'exécuter dans des sites distants avec les paramètres fournis par le module administrateur et les ressources réservées par le module 1.
Cet environnement d'attaque peut utiliser une architecture réseau supervisé par un serveur qui représente l'ordinateur de l'utilisateur où est implémenté le module administrateur et le module 1, tandis que les autres ordinateurs sont utilisés pour l'exécution des trois autres modules parallèlement.

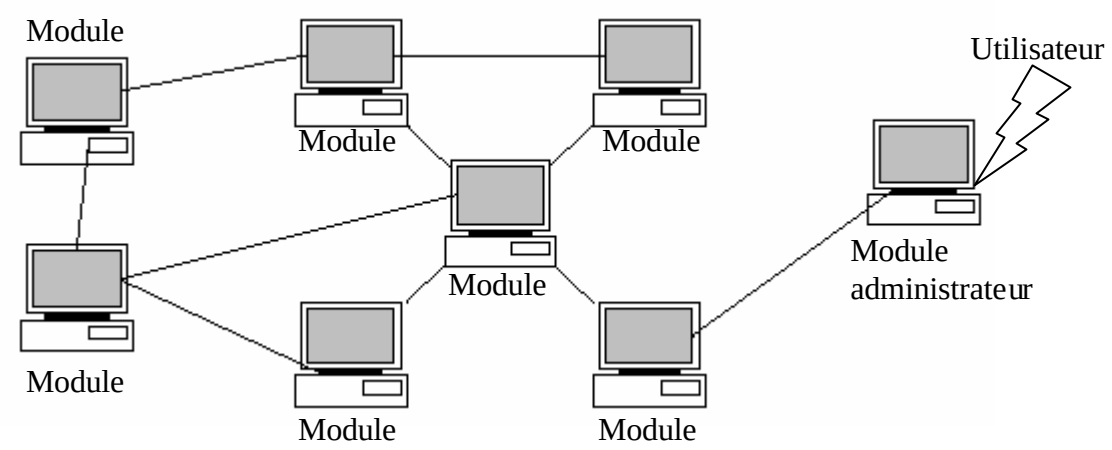


Figure 3.2. Exemple d'exécution de l'environnement d'attaque.

Afin de garantir une bonne intégration des différents modules de l'environnement, on propose d'utiliser la méthodologie orientée objet qui se fonde sur un concept de base qui est l'objet. Un objet est une entité qui regroupe un ensemble de données et un ensemble de procédures utilisant ces données, appelées méthodes.

Dans notre cas on opte pour une architecture centralisée autour d'un objet expert représenté dans notre environnement par le module administrateur. Donc les processus de dialogue et de calcul seront entièrement supervisés par l'objet expert, celui-ci entre en interaction avec les autres modules par le biais de messages envoyés et reçus de part et d'autre. L'interprétation de ces messages permet soit de valider les choix de l'utilisateur ou de présenter à celui-ci des résultats qu'il devra évaluer et apprécier.

3.6. Conclusion

Dans cette Partie nous avons présenté trois cryptanalyses de l'AES, deux basées sur des modèles statistiques formels et une, basée sur des méthodes d'observation des canaux cachés qui révèlent des informations sur les paramètres du cryptosystème. Les attaques présentées sont applicables sur des versions spéciales de l'AES, soit avec un nombre d'itérations réduit, ou bien des implémentations logicielles spécifiques. Par conséquent ces attaques ne sont efficaces que dans le cas de marges de sécurité limitées.

Conclusion générale et perspectives

Au cours de ce mini projet, portant sur la cryptanalyse des cryprosystèmes symétriques, nous avons cherché à identifier les différentes aspects de la cryptologie en particulier les deux aspects *cryptographie* et *cryptanalyse*. Le standard de cryptage AES été au coeur de notre étude. Pour ce faire, nous nous sommes consacrés à une étude de fond du standard.

Après avoir décortiqué le standard dans la deuxième partie, nous avons présenté trois attaques contre l'AES, deux basées sur des modèles statistiques formels et une basée sur des méthodes d'observation des canaux cachés dans notre cas c'est le temps d'exécution, qui révèle des informations sur les paramètres du cryptosystème. Les attaques présentées ne sont pas applicables sur toutes les versions de l'AES mais seulement sur des versions spéciales ou bien avec des conditions limitées. Par conséquent ces attaques ne présentent pas de danger sur l'AES, mais on peut les considérer comme des études académiques pour l'amélioration et le développement futur.

Enfin, et comme perspectives, on note la nécessité de l'étude conceptuelle approfondie et l'implémentation de l'environnement d'attaque proposé pour vérifier les résultats, ou bien trouver des combinaisons qui augmentent le nombre d'itérations vulnérables. D'autre part une étude des corrélations possibles dans la table de substitution S-box est souhaitable, dans le but d'améliorer les attaques proposées.

Références bibliographiques

[1]. W. STALLINGS, *"Cryptography and network security"*, 2nd edition, Prentice hall,1996.

[2]. B. SCHNEIER, *"Cryptographie appliquée"*, 2eme édition, Thomson Publishing, 1996.

[3]. J-P. BARTHELEMEY [et all], *"Complexité algorithmique et problèmes de communication"*, Masson, 1992.

[4]. J. DAEMEN, V. RIJMEN, *"AES proposal Rijndael"*, (2eme version), AES submission, 1999.

[5]. F. KOEUNE, J.J. QUISQUATER, *"A timing attack against Rijndael"*, 1999.

[6]. S.LUCKS, *"Attaking seven Rounds of Rijndael under 192-bit and 256-bit key", 2000.*

[7]. E. FILIOL, *"Plaintext-dependant Repetition Codes cryptanalysis of block ciphers - The AES case"*, 2003.

[8]. N. FERGUSON [and All], *"Improved cryptanalysis of Rijndael", 2002.*

[9]. L. KNUDSEN, D. WAGNER, *"Integral cryptanalysis (Extended abstract)", 2000.*

[10]. *Professeur* B. LANCTOT, *"Cours de Sécurité Informatique"*, université de Montréal, 2002.