MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

Cryptographie

Polycopié de Cours et Exercices Corrigés

3ème année licence- SI

Dr. BENIDRIS Fatima zohra

Juin, 2020

Avant-propos

Ce Document est un support de cours destiné aux étudiants de la troisième année licence informatique, option systèmes informatique. Il est enseigné en tant que élément constitutif de l'unité d'enseignement optionnelle. Il est composé d'un ensemble de cours et travaux dirigés. Le contenu de ce cours est conçu conformément au programme de la matière fixé par la tutelle.

Objectifs du cours

- En termes de connaissance : apprendre aux étudiants les notions de base de la cryptographie, les différentes méthodes classiques et modernes et la différence entre la cryptographie symétrique et asymétrique.
- En termes de savoir-faire : apprendre aux étudiants à chiffrer et déchiffrer des messages entre un émetteur et un récepteur selon une méthode de chiffrement donnée.
- En termes de savoir-être : sensibiliser les étudiants au respect des exigences de communication et la sécurité des donnés.

Pré-requis

- Concepts de sécurité de données (confidentialité, intégrité, authentification, non répudiation...)
- Concepts mathématiques et logique (complexité, division euclidienne, inverse modulo, Ou exclusif...)
- Système binaire et hexadécimal (conversion, addition, multiplication...)

Table des matières

Chapitre 1: Initiation à la cryptographie 4
1.1 Introduction 4
1.2 Qu'est-ce que la cryptographie ? 4
1.3 Définition formelle de la cryptographie 4
1.4 Notions de base 5
1.5 Protocole de chiffrement 6
1.6 Qualités d'un cryptosystème 6
1.7 La cryptanalyse 7
1.7.1 Techniques de cryptanalyse 8
1.7.2 Sécurité des cryptosystèmes 8
1.8 Conclusion 9
Chapitre 2 : Cryptographie classique 10
2.1 Introduction 10
2.2 Chiffre à substitution 10
2.3 Chiffre à transposition 11
2.4 Les méthodes de chiffrement classiques 11
2.4.1 La scytale 11
2.4.2 Le code de César 11
2.4.3 Chiffrement par décalage 12
2.4.5 Permutation des lettres 12
2.4.6 Chiffre de Vigenère 13
2.4.7 Chiffrement de Hill 15
2.4.8 Chiffrement affine 16
2.4.9 Chiffre de Vernam 16
2.5 Chiffrement par transposition 17
2.5.1 Transposition à base matricielle 17
2.5.2 Transposition à base matricielle complexe 18
2.6 Difficultés de la cryptographie classique 19
2.7 Exercices 19
Chapitre 3 : Cryptographie moderne à clé secrète 21
3.1 Introduction 21

3.2 Les familles de codes modernes 21
3.2.1 Les codes par flot 21
3.2.2 Les codes par blocs 22
a) Le mode Electronic Codebook (ECB) 22
b) Le mode Cipher Block Chaining (CBC) 23
c) Le mode Cipher Feedback (CFB) 23
d) Le mode Output Feedback (OFB) 24
3.3 Chiffrement symétrique 25
3.3.1 Chiffrement de Feistel 25
3.3.2 DES (Data Encryption Standard) 26
a) Fonctionnement de DES 27
b) Construction des sous-clés 30
c) Cryptanalyse 31
d) Triple DES 31
3.3.4 AES (Advanced Encryption Standard) 31
a) Description de l'algorithme AES 32
b) Construction des sous-clés 35
c) Déchiffrement AES 36
d) Caractéristiques et points forts de l'AES 37
e) Cryptanalyse 37
3.4 Problèmes de chiffrement symétrique 38
3.5 Exercices 38
Chapitre 4: Cryptographie à clé publique 41
4.1 Introduction 41
4.2 Principe de la cryptographie 41
4.3 Fonction à sens unique : 42
4.4 Le protocole de Diffie et Hellman 43
4.5 Le système RSA 43
4.5.1 Principe 44
4.5.2 Discussion 45
4.5.3 Système hybride 45
4.6 La signature numérique 45
4.7 Protocole de signature RSA 46
4.8 Robustesse du chiffrement asymétrique 46

4.9 Fonction de hachage 46
4.9.1 Principe 47
4.9.2 Efficacité de l'opération 48
4.9.3 Propriétés des fonctions de hachage 48
4.9.4 MD5 (Message Digest 5) 48
a) Principe de fonctionnement 48
b) Algorithme MD5 50
c) La cryptanalyse 51
4.10 Protocole de signature à clé publique et fonction de hachage. 51
4.11 Infrastructure des systèmes à clef publique 52
4.10.1 Certificat 53
4.10.2 Structure du certificat X.509v3 54
4.10.3 Certificat et vérification 54
4.11 Exercices 56
Conclusion générale 58
Corrigés des exercices 59

Bibliographie

1. Philippe Guillot. La cryptologie, L'art des codes secrets. EDP Sciences 2013
2. Gilles Dubertet, Initiation à la Cryptographie, Vuibert 2002.
3. Bruce Schneier. Cryptographie appliquée. Vuibert 2001
4. Daniel Lamas. La cryptographie. Bachelor HES à l'école de gestion de Genève. 2015.
5. Touradj Ebrahimi, Frank Leprévost et bertrand Warusfel. Cryptographie et sécurité des systèmes et réseaux. Lavoisier 2006
6. Laurent Bloch et Christophe Wolfhugel. Sécurité informatique : Principes et méthode à l'usage des DSI, RSSI et administrateurs. 2e édition, Eyrolles, Paris, 2009.
7. Daniel Barsky et Ghislain Dartois. Cours de Cryptographie. Université Paris 13. 2010.
8. Raphael Yende. Support de cours de sécurité informatique et crypto. Master Congo-Kinshasa. 2018. ffcel-01965300ff
9. Saiida Lazaar. Polycopié de cours de cryptographie. (ENSA) Université de AbdelMalek Essaadi, Mroc.2019.

Site internet

1. http://cryptosec.lautre.net/index.php3
2. http://www.securite.org/db/crypto/
3. http://www.securiteinfo.com/crypto/aes.shtml
4. http://www.bibmath.net/crypto/moderne/aes.php3

Chapitre 1: Initiation à la cryptographie

1.1 Introduction

Depuis l'antiquité, l'homme a perçu le besoin de cacher des informations personnelles ou confidentielles en utilisant des codes qui ont servi à protéger le contenu de certains messages des inévitables curieux d'où l'apparence de la cryptographie.

La cryptographie est une science très ancienne, il faut remonter à environ 3000 ans avant notre ère, pour en trouver les premières traces. De ce temps-là et au long de l'histoire, la cryptographie a été utilisée exclusivement à des fins militaires. A l'heure actuelle, la généralisation rapide des communications par Internet exigent une phase de cryptographie comme mécanisme fondamental afin d'assurer la confidentialité des informations transmises ou stockées.

L'objectif fondamental de la cryptographie est de permettre à deux personnes appelées traditionnellement, Alice et Bob de communiquer à travers un canal peu sûr de telle sorte qu'un opposant passif Eve ne puisse pas comprendre ce qui est échangé et que les données échangées ne puissent pas être modifiées ou manipulées par un opposant actif Martin.

1.2 Qu'est-ce que la cryptographie ?

Avant d'expliquer la cryptographie il faut tout d'abord la distinguer du terme « Stéganographie » car les deux méthodes visent à protéger l'information mais chacune à sa façon :

Steganographie: cacher le message pour que l'ennemi ne le trouve pas.

Durant l'antiquité, certains généraux rasaient le crâne de leurs esclaves, leur tatouaient un message et attendaient que les cheveux repoussent pour faire passer des informations importantes. La stéganographie permet de nos jours à dissimuler un fichier, une musique, un dessin, un texte dans un autre document numérique, musique, texte, code html, ... Avec l'informatique la stéganographie prend une nouvelle ampleur, mais la dissimulation de message ne date pas d'hier.

Cryptographie: rendre le message incompréhensible par l'ennemi.

Le mot cryptographie vient des mots en grec ancien kruptos « caché » et graphein « écrire ». La cryptographie ou science du secret est l'étude des méthodes donnant la possibilité d'envoyer des données de manière confidentielle sur un support donné.

1.3 Définition formelle de la cryptographie

Un algorithme de chiffrement est un quintuplait A = (M, C, K, E, D) où:

- M est un ensemble fini de blocs de textes clairs possibles,
- C'est un ensemble fini de blocs de textes chiffrés possibles,
- K est un ensemble fini de clefs possibles,
- E est une fonction de chiffrement, $E : M \times K \rightarrow C$
- D est une fonction de déchiffrement, $D : C \times K \rightarrow M$
- Pour chaque clef K, il existe au moins une clef K' telle que pour tout $x \in M$, $D (E (x, K), K') = x$

Le type de relation qui unit les clés K et K' permet de définir deux grandes catégories de systèmes cryptographiques :

- Les systèmes à clef secrètes ou symétriques: (DES, AES, IDEA,...)
- Les systèmes à clefs publiques ou asymétriques: (RSA, El-Gamal,...)

1.4 Notions de base

Voici les quelques terminologies liées à la cryptologie :

- **Cryptologie** : Il s'agit d'une science mathématique comportant deux branches : la cryptographie et la cryptanalyse

- **Cryptographie** : C'est l'étude des méthodes donnant la possibilité d'envoyer des données de manière confidentielle sur un support donné.

- **Cryptanalyse** : est l'art pour une personne non habilitée, de décrypter, de décoder, de déchiffrer, un message. C'est donc l'ensemble des procèdes d'attaques d'un système cryptographique. elle a pour but de retrouver le texte clair à partir de textes chiffrés en déterminant les failles des algorithmes utilisés.

- **Chiffrement :** Le chiffrement consiste à transformer une donnée (texte, message,...) afin de la rendre incompréhensible par une personne autre que celui qui a créé le message et celui qui en est le destinataire.

- **Cryptogramme :** Appelé également le texte chiffré, c'est le résultat de l'application d'un chiffrement à un texte clair. Dans le cas d'un algorithme symétrique, la clef est identique lors des deux opérations. Dans le cas d'algorithmes asymétriques, elle diffère pour les deux opérations.

- **Clef :** Il s'agit du paramètre impliqué et autorisant des opérations de chiffrement et/ou déchiffrement.

- **Crypter** - synonyme de "chiffrer".

- **Cryptosystème :** Il est défini comme l'ensemble des clés possibles, des textes clairs et chiffrés possibles associés à un algorithme donné.

- **Code** : Système de chiffrement dans lequel chaque lettre (ou mot, syllabes,...) est remplacé par un -ou plusieurs- symboles (caractères, dessins, ...), par un processus d'opérations.(le plus souvent des tables de correspondances entre la lettre et son symbole). Les codes, s'ils peuvent être secrets, ne le sont généralement pas. Parmi les plus connus, on peut citer le code morse et le code ASCII.
- **Chiffre** : nom donné à un code secret , c'est à dire soit un code dont le processus d'opérations (l'algorithme) est tenu secrète , soit un code dont l'algorithme est connu, mais dont la clef est secrète.
- **Déchiffrer** : opération inverse du chiffrage : transformer un texte chiffré en un texte en clair en connaissant le procédé de secret utilisé. (c'est à dire l'algorithme du chiffre et sa clé, s'il en a une)
- **Décrypter** : transformer un texte chiffré en un texte en clair, sans connaître le procédé de secret utilisé. C'est sur ce point que déchiffrage et décryptage s'opposent et ne sont, de ce fait, pas synonymes.
- **Cryptolecte** : jargon réservé à un groupe restreint de personnes désirant dissimuler leur communication.

1.5 Protocole de chiffrement

L'information qu'Alice souhaite transmettre à Bob, que l'on appelle texte en clair (ou message) clair, peut être un texte écrit en certaine langue ou des données numériques.

Alice transforme le texte clair par un procédé de chiffrement, en utilisant une clé prédéterminée, et envoie le message chiffré au travers du canal (peut être par exemple une ligne de téléphone, Internet, ou autre).

Eve, qui espionne éventuellement le canal, ne peut retrouver le texte clair, mais Bob, qui connaît la clef pour déchiffrer, peut récupérer le message clair à partir du texte chiffré.

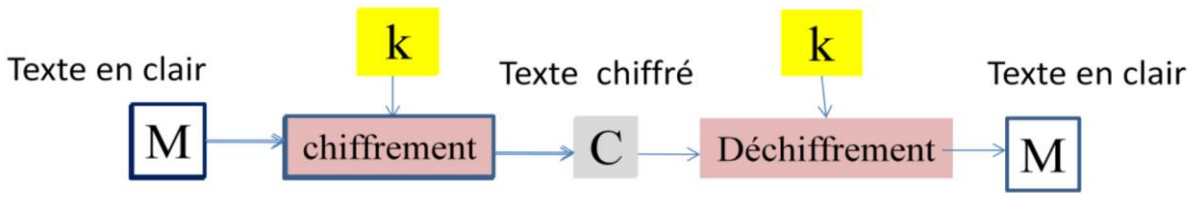


Figure 1.1 Principe d'un cryptosystème

1.6 Qualités d'un cryptosystème

Protéger un message ne signifie pas seulement de le rendre incompréhensible. En effet, on peut distinguer quatre types de protection:

- **Confidentialité**: seules les personnes habilitées ont accès au contenu du message.
- **Intégrité des données**: le message ne peut pas être falsifié sans qu'on s'en aperçoive.
- **Authentification**:
  - l'émetteur est sûr de l'identité du destinataire c'est à dire que seul le destinataire pourra prendre connaissance du message car il est le seul à disposer de la clef de déchiffrement.
  - le receveur est sûr de l'identité de l'émetteur
- **Non-répudiation** qui se décompose en trois:
  - non-répudiation d'origine l'émetteur ne peut nier avoir écrit le message et il peut prouver qu'il ne l'a pas fait si c'est effectivement le cas.
  - non-répudiation de réception le receveur ne peut nier avoir reçu le message et il peut prouver qu'il ne l'a pas reçu si c'est effectivement le cas.
  - non-répudiation de transmission l'émetteur du message ne peut nier avoir envoyé le message et il peut prouver qu'il ne l'a pas fait si c'est effectivement le cas.

Les qualités pratiques attendues d'un cryptosystème :

1. Il doit résister aux attaques connues et si possible avoir une sécurité prouvée.
2. Il doit permettre de coder et décoder rapidement (en temps réel pour certaines applications).

Il n'existe pas de codes qui réunissent toutes ces qualités simultanément. Il faut donc un compromis adapté à chaque situation.

1.7 La cryptanalyse

La cryptanalyse ou l'attaque sur un chiffrement est l'ensemble des procédés d'attaque d'un cryptosystème. Elle est indispensable pour l'étude de la sécurité des procédés de chiffrement utilisés en cryptographie. Son but ultime est de trouver un algorithme de déchiffrement des messages. Le plus souvent on essaye de reconstituer la clef secrète de déchiffrement.

On doit distinguer entre les types d'attaques d'un adversaire et les buts des attaques d'un adversaire. Les principaux types d'attaques:

- **attaque à texte chiffré connu:** l'opposant ne connait que le message chiffré y.
- **attaque à texte clair connu:** l'opposant dispose d'un texte clair x et du message chiffré correspondant y
- **attaque à texte clair choisi:** l'opposant a accès à une machine chiffrante. Il peut choisir un texte clair et obtenir le texte chiffré correspondant y, mais il ne connait pas la clef de chiffrement.

- **attaque à texte chiffré choisi:** l'opposant a accès à une machine déchiffrante. Il peut choisir un texte chiffré, y et obtenir le texte clair correspondant x, mais il ne connait pas la clef de d´déchiffrement.

En plus de ces attaques basées sur une étude de messages codés, il y a aussi des attaques physiques. Le principe de ces attaques est d'essayer de reconstituer la clef secrète par exemple en espionnant la transmission entre le clavier de l'ordinateur et l'unité centrale ou en mesurant la consommation électrique du microprocesseur qui effectue le décodage du message ou encore en mesurant son échauffement. Ensuite on essaye de remonter de ces données physiques aux clefs de codage et décodage.

Le but de l'attaque d'un adversaire peut être soit de découvrir la clef du chiffrement et de pouvoir ainsi décrypter tous les messages de l'émetteur ou plus modestement de décrypter un message particulier sans nécessairement disposer de la clef du code.

Afin de garantir la confidentialité des communications entre Alice et Bob, nous allons supposer donc qu'Eve ne peut pas :

- trouver M à partir de E(M) (le crypto-système doit être résistant aux attaques sur le message codé)
- trouver la méthode de déchiffrement D à partir d'une famille de couples, $\{(M_i, E(M_i)\}$, (message clair, message codé correspondant).
- accéder à des données contenues dans le micro-processeur qui code et décode et plus généralement ne puisse pas espionner les ordinateurs d'Alice et de Bob

1.7.1 Techniques de cryptanalyse

a) Recherche exhaustive de la clef

Cette technique consiste simplement à essayer toutes les clefs possibles, jusqu'à ce qu'on trouve la bonne. Pour les chiffres à alphabet décalé, cette recherche est envisageable, puisqu'il y a peu de possibilités (par exemple 26 avec l'alphabet latin occidental)

b) Analyse des fréquences

Dans le cas d'un chiffre mono alphabétique, c'est-à-dire quand l'alphabet est désordonné, ou que chaque lettre est remplacée par un symbole, on peut s'appuyer sur une analyse des fréquences des lettres ou des bigrammes.

c) Technique du mot probable

Une technique très puissante de décryptement consiste à supposer qu'une séquence de lettres du cryptogramme correspond à un mot que l'on devine.

1.7.2 Sécurité des cryptosystèmes

Les différents algorithmes de chiffrement ont des niveaux de sécurité divers, plus ou moins difficiles à casser. Si le coût nécessaire pour casser un algorithme dépasse la valeur de

l'information chiffrée, alors cet algorithme est probablement sûr. Si le temps nécessaire pour casser un algorithme est plus long que le temps durant lequel l'information chiffrée doit rester secrète, alors cet algorithme est probablement sûr. S'il faut plus d'information pour casser l'algorithme qu'il n'en a été chiffré avec la même clef, alors votre algorithme est probablement sûr. Nous disons « probablement » car il est toujours possible qu'une nouvelle avancée soit faite en cryptanalyse. D'un autre côté, une information perd de sa valeur avec le temps. Il est important que la valeur d'une information reste toujours inférieure au coût nécessaire pour briser la protection qui l'entoure.

Il existe différentes manières de casser un algorithme. Les voici par ordre décroissant de sévérité:

1. **Cassage complet :** Un cryptanalyste trouve la clef k telle que $D_k(C) = M$.
2. **Obtention globale :** Un cryptanalyste trouve un algorithme de remplacement A équivalent à $D_k(C)$ sans connaître k.
3. **Obtention locale :** Un cryptanalyste trouve le texte en clair d'un message chiffré qu'il a intercepté.
4. **Obtention d'information :** Un cryptanalyste glane quelque information à propos du texte en clair ou de la clef. Cette information pourrait être certains bits de la clef, un renseignement sur la forme du texte en clair, et ainsi de suite.

Un algorithme est inconditionnellement sûr si, peu importe la quantité de texte chiffré dont le cryptanalyste dispose, il n'y a pas d'information suffisante pour retrouver le texte en clair.

1.8 Conclusion

Dans les chapitres suivants, nous allons focaliser essentiellement sur les différentes méthodes de chiffrements apparues au fils du temps : les méthodes de chiffrement dites classiques ou manuelle et celles dites modernes ou informatique.

Chapitre 2 : Cryptographie classique

2.1 Introduction

De l'époque de Jules César aux alentours du XVIème siècle à la fin des années 1970, un grand nombre de systèmes de chiffrement ont été inventés, qui consistaient à faire subir à un texte clair une transformation plus ou moins complexe pour en déduire un texte inintelligible, dit chiffré. La cryptographie classique traitait des cryptosystèmes basés sur les lettres (ou caractères). Les différents algorithmes cryptographiques remplaçaient des caractères par d'autres ou transposaient les caractères. Les meilleurs systèmes faisaient les deux opérations plusieurs fois. Cependant, ces méthodes se décomposent en deux grandes familles de chiffrement :

- par substitution.
- par transposition.

2.2 Chiffre à substitution

Un chiffre à substitution est un chiffre dans lequel chaque caractère du texte en clair est remplacé par un autre caractère dans le texte chiffré. Le destinataire applique la substitution inverse au texte chiffré pour recouvrer le texte en clair.

En cryptographie classique, il y a quatre types de base de substitution :

- **Un chiffre à substitution simple** est un chiffre dans lequel chaque caractère du texte en clair est remplacé par un caractère correspondant dans le texte chiffré. Les cryptogrammes publiés dans les journaux sont des exemples de chiffres à substitution simple.
- **Un chiffre à substitution homophonique** (ou chiffre à substitution simple à représentation multiple) est comme un chiffre à substitution simple, sauf qu'à un caractère du texte en clair on fait correspondre plusieurs caractères dans le texte chiffré. Par exemple, « A » peut correspondre à 5, 13, 25 ou 56, « B » peut correspondre à 7, 19, 31 ou 42 ; etc.
- **Un chiffre à substitution simple par polygames** est un chiffre pour lequel les caractères sont chiffrés par blocs. Par exemple, « ABA » peut être chiffré par « RTQ » tandis que « ABB » est chiffré par « SLL ».
- **Un chiffre à substitution polyalphabétique** est composé à partir de plusieurs chiffres à substitution simple. Par exemple, il peut y avoir 5 chiffres à substitution simple utilisés ; celui qui est utilisé dépend de la position du caractère à chiffrer dans le texte en clair.

Le code secret de Jules César est un exemple historique de chiffrement par substitution.

2.3 Chiffre à transposition

Un chiffre à transposition est un chiffre dans lequel les caractères du texte en clair demeurent inchangés mais dont les positions respectives sont modifiées.

Un exemple historique dont le principe est encore utilisé est la méthode de la grille (principe de la scytale utilisée par les spartiates vers -450 avant Jule César).

Pour appliquer la transposition simple en colonnes, on écrit le texte en clair horizontalement sur un morceau de papier quadrillé de largeur fixe et l'on relève le texte chiffré verticalement. Pour déchiffrer le texte chiffré, il suffit d'écrire verticalement celui-ci sur un morceau de papier quadrillé de la même largeur et de lire horizontalement le texte en clair.

2.4 Les méthodes de chiffrement classiques

2.4.1 La scytale

Entre le Xème et le VIIème siècle avant Jules César, les Grecs utilisaient des scytales, des sortes de bâtons en bois. Quand l'émetteur voulait communiquer, il enroulait une bande de cuir sur la scytale et y inscrivait le message (une lettre par bout de bande). Une fois la bande déroulée, les lettres n'étaient plus ordonnées et n'avaient donc plus aucun sens. Le seul moyen de pouvoir comprendre le message était d'enrouler la bande sur une scytale de même diamètre pour que les lettres puissent s'aligner correctement.

Exemple:

Le message en clair est: « *KILL KING TOMORROW MIDNIGHT* ». Le message, une fois déroulé, n'est plus compréhensible. Le message crypté est:
« KTMIOILMDLONKRIIRGNOHGWT »

Figure 2.1 La scytale

Cryptanalyse: Un casseur peut déchiffrer le message en essayant des cylindres de diamètre successifs différents,

2.4.2 Le code de César

Méthode de chiffrement très simple utilisée par Jules César dans ses correspondances secrètes. Le chiffre de César est un des premiers chiffrements par substitution. Son principe est simple, il fonctionne par décalage des lettres de l'alphabet de 3 positions.

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

Figure 2.2 Correspondance entre les lettres du texte clair et celles du texte chiffré.

Exemple:

Le message clair est: CECI EST UN CRYPTOGRAMME
On décale les lettres de 3 positions :
Le texte chiffré est: FHFL HVW XQ FUBSWRJUDPPH

2.4.3 Chiffrement par décalage

Ce chiffrement est une amélioration de celui de Jules César où le décalage des lettres n'est plus fixe à trois positions. Il peut être représenté comme suit:

- Transformer chaque lettre en un nombre (A = 0, B = 1,..., Z = 25),
- Coder une lettre ***x*** avec une clé ***k*** en appliquant la formule: $E(x) = (x + k) \bmod (26)$
- Le déchiffrement consiste à utiliser la clé opposée: $D(y) = (y - k) \bmod (26)$

Cryptanalyse par force brute : Si l'algorithme est connu ainsi que le langage utilisé alors le déchiffrement est très simple : 25 clés à essayer.

2.4.5 Permutation des lettres

C'est un chiffrement par substitution mono-alphabétique qui consiste à appliquer une permutation plus complexe qu'un simple décalage où on associe à chaque lettre une autre lettre sans ordre fixe ou règle générale. Cependant, l'effet sur le texte chiffré reste le même : une même lettre dans le texte clair sera chiffrée par une même lettre dans le texte chiffré.

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

Table 1.1 Permutation des lettres

Message en clair: ATTAQUE AU MATIN
Cryptogramme: MIIMUO MW SMIQH

Cryptanalyse par analyse de fréquence :

Au IXème siècle, Al-Kindi écrit le premier manuscrit traitant de cryptanalyse (retrouvé il y a seulement 30 ans à Istanbul). Il y montre notamment qu'étant donné une langue, la fréquence d'apparition de chaque lettre n'est pas la même. Ainsi dans la langue de Molière, le 'e' apparaît bien plus souvent que le 'w'. En se basant sur la loi des grands nombres et en analysant la fréquence d'apparition des lettres dans la Bible, écrite en français, on obtient le tableau de fréquence d'apparition suivant :

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M
Fréquence	9,42	1,02	2,64	3,39	15,87	0,95	1,04	0,77	8,41	0,89	0,00	5,34	3,24
Lettre	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Fréquence	7,15	5,14	2,86	1,06	6,46	7,90	7,26	6,24	2,15	0,00	0,30	0,24	0,32

Table 1.2 Table des fréquences d'apparition des lettres dans la Bible en langue française.

À partir de cette table, le raisonnement est le suivant. Nous connaissons la fréquence d'apparition des lettres dans le texte chiffré. Si nous faisons l'hypothèse que telle ou telle langue a été utilisée pour le texte clair, et sous réserve que celui-ci soit suffisamment représentatif de la langue supposée, alors comme chaque lettre est toujours chiffrée par la même lettre, la lettre apparaissant le plus dans le texte chiffré sera avec une bonne probabilité celle qui a la plus grande fréquence d'apparition dans la langue utilisée pour la rédaction du texte clair. Par essais successifs au maximum de vraisemblance, on parvient assez rapidement à retrouver la clé secrète, et donc à déchiffrer.

2.4.6 Chiffre de Vigenère

Le chiffre de Vigenère est un système de chiffrement poly alphabétique, c'est un chiffrement par substitution, mais une même lettre du message clair peut, suivant sa position dans celui-ci, être remplacée par des lettres différentes, contrairement à un système de chiffrement mono alphabétique comme le chiffre de César (qu'il utilise cependant comme composant). Cette méthode résiste ainsi à l'analyse de fréquences, ce qui est un avantage décisif sur les chiffrements mono alphabétiques.

Ce chiffrement introduit la notion de clé. Une clé se présente généralement sous la forme d'un mot ou d'une phrase. Pour pouvoir chiffrer notre texte, à chaque caractère nous utilisons une lettre de la clé pour effectuer la substitution. Évidemment, plus la clé ne sera longue et variée et mieux le texte sera chiffré.

Pour chaque lettre en clair, on sélectionne la colonne correspondante et pour une lettre de la clé on sélectionne la ligne adéquate, puis au croisement de la ligne et de la colonne on trouve la lettre chiffrée. La lettre de la clé est à prendre dans l'ordre dans laquelle elle se présente et on répète la clé en boucle autant que nécessaire.

Exemple :

Le texte clair CRYPTOGRAMME est chiffré comme suit avec la clé MINET.
Nous répétons autant de fois le mot MINET pour être aussi long que le texte clair :

C R Y P T O G R A M M E

M I N E T M I N E T M I

En utilisant le carré de Vigenère ci-dessous, Nous obtenons le cryptogramme suivant : O Z L T M A O E E F Y M

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

Figure 2.2 Carré de vigenère

Mathématiquement, on identifie les lettres de l'alphabet aux nombres de 0 à 25 (A=0, B=1...). Les opérations de chiffrement et de déchiffrement sont, pour chaque lettre, celles du chiffre décalage : **E(x) = (x + k) mod (26) / D (x) = (x - k) mod (26)** tel que x est la position d'une lettre du texte en clair et k est la position d'une lettre du mot clé qui lui correspond.

La grande force du chiffre de Vigenère est que la même lettre sera chiffrée de différentes manières. Ce qui rend inutilisable l'analyse des fréquences classique.

Cryptanalyse :

Le nombre de lettres contenues dans les enchaînements se répétant est élevé, moindre est le risque de faux positifs. Nous ne rentrerons pas plus dans les détails mais le calcul de l'indice de coïncidence permet de déterminer si un texte a été chiffré en utilisant un chiffrement mono ou poly-alphabétique, ainsi que la langue utilisée le cas échéant.

Enfin, il est important de remarquer que cette cryptanalyse fonctionne lorsque le mot-clé a une longueur bien inférieure à celle du texte qui est chiffré. Dans le cas (peu pratique) où la clé est de même longueur que le texte clair, ce chiffrement est assimilé à un chiffrement de Vernam. Ce dernier est le seul schéma de chiffrement dit à "secret parfait" connu, c'est-à-dire qu'il n'est théoriquement pas cassable dès lors que la clé n'est utilisée qu'une seule fois, ce qui vaut également à ce schéma le nom de masque jetable ou one-time pad.

2.4.7 Chiffrement de Hill

Le chiffrement que nous allons étudier a été publié par Lester S. Hill en 1929. C'est un chiffrement polygraphique, c'est-à-dire qu'on ne (dé)chiffre pas les lettres les unes après les autres, mais par paquets. On étudie ici la version biographique, c'est-à-dire que l'on groupe les lettres deux par deux, mais on peut envisager des paquets plus grands. Pour coder un message selon ce procédé, on commence par grouper les lettres de ce message deux par deux, puis on remplace chaque lettre par un nombre (son rang dans l'alphabet). Les lettres P_k et P_{k+1} deviennent C_k et C_{k+1}

$$\begin{pmatrix} C_k \\ C_{k+1} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} P_k \\ P_{k+1} \end{pmatrix} (\mathrm{mod} 26)$$

- Chaque digramme clair (P1 et P2) sera chiffré (C1 et C2) selon:

$$C_1 \equiv aP_1 + bP_2 (\mathrm{mod} 26)$$

$$C_2 \equiv cP_1 + dP_2 (\mathrm{mod} 26)$$

Remarque: Les composantes de cette matrice doivent être des entiers positifs. De plus la matrice doit être inversible dans Z_{26}

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

Le nombre de lettres étant impair, on lui adjoint arbitrairement un "x". On découpe alors ce mot en blocs de deux lettres "SU PI NF OX". Après avoir remplacé les lettres par leur rang dans l'alphabet (a=0, b=1, etc.), on obtiendra:

$y1 \equiv 9 \times 18 + 4 \times 20 \equiv 242 \mathrm{mod}\ (26) \equiv 8$ (le rand de la lettre I)

$y2 \equiv 5 \times 18 + 7 \times 20 \equiv 230 \mathrm{mod}\ (26) \equiv 22$ (le rand de la lettre W)

On fera de même avec les autres digrammes. Le résultat est: **IWLBHWKX**

Déchiffrement : C'est le même que pour le chiffrement : on prend les lettres deux par deux, puis on les multiplie par une matrice :

$$\begin{pmatrix} P_1 \\ P_2 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} (\mathrm{mod} 26)$$

Cette matrice doit être l'inverse de matrice de chiffrement (modulo 26). Ordinairement, cet inverse est:

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

Exemple précédent: Pour déchiffrer le message, on doit calculer:

$$\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}^{-1} = \frac{1}{43}\begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} (\mathrm{mod} 26) = (43)^{-1}\begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} (\mathrm{mod} 26)$$

Comme pgcd (43, 26) = 1 et $(43)^{-1}$ existe dans Z_{26} et $(43)^{-1} = 23$. On a la matrice de déchiffrement :

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

On prend donc cette matrice pour déchiffrer le message : **IWLBHWKX**

Après avoir découpé le mot en blocs et remplacé les lettres par leur rang dans l'alphabet (A=0, B=1, etc.), on obtiendra :

$X1 \equiv 5\times14+12\times8 \bmod (26) \equiv 18$

$X2 \equiv 15\times14+25\times8 \bmod (26) \equiv 20$

On fera de même avec les autres blocs. Finalement, le message en clair est : **SUPINFOX**

Cryptanalyse : La technique du mot probable marche bien pour les substitutions polyalphabétiques comme le chiffrement de Hill.

2.4.8 Chiffrement affine

On représente l'alphabet latin par des entiers entre 0 et 25. Et on définit l'ensemble de clés comme suit :

$$\mathcal{K} := \{(a, b) \in \mathbb{Z}_{26} \times \mathbb{Z}_{26} : pgcd(a, 26) = 1\} .$$

Pour tout (a, b) ∈ K, on définit les formules de chiffrement et déchiffrement suivantes :

$$E_{(a,b)}(x) := ax + b \mod 26$$

$$D_{(a,b)}(y) := a^{-1}(y - b) \mod 26 .$$

Cryptanalyse : Par le chiffre Affine, on obtient 312 clés possibles. En effet, pour respecter la propriété de a, il n'y a que 12 choix possibles. Et puisque b peut prendre n'importe quelle valeur dans [0, 25], il vient $12 * 26 = 312$.

2.4.9 Chiffre de Vernam

Chiffre de Vernam ou masque jetable est inventé par Gilbert Vernam en 1917 et publié en 1926. Il peut être décrit simplement comme un chiffre de Vigenère, mais où la clé répond aux trois impératifs suivants :

- Aussi longue que le texte à chiffrer;
- Parfaitement aléatoire;
- Utilisée que pour chiffrer un seul message, puis est immédiatement détruite.

C'est Claude Shannon, qui prouva en 1949 le fait que ce chiffre est parfaitement sûr. La seule information dont on dispose si on intercepte le message chiffré est la longueur du message clair. Tout chiffre **parfaitement sûr** est nécessairement **une variante du chiffre de Vernam.**

Système de Vernam :

- Les téléscripteurs transmettent les textes à l'aide d'un codage inscrit sur un ruban perforé.
- Chaque caractère est codé par cinq unités qui vont se traduire par le passage ou non du courant électrique.
- L'idée de Vernam est de combiner le ruban qui contient le texte en clair avec un second ruban.
- La règle de combinaison utilisée est celle du XOR

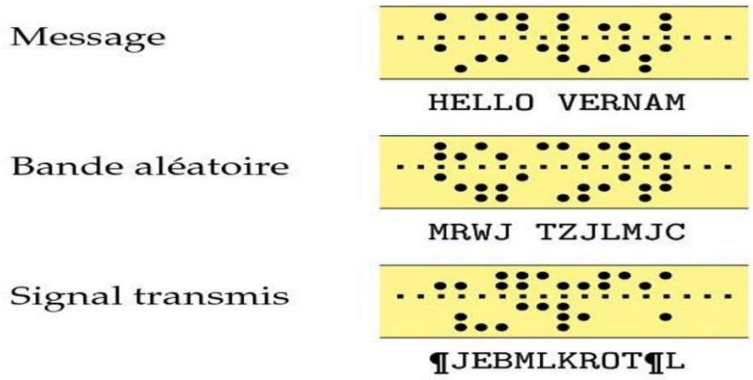


Figure 2.3 Chiffrement de Vernam

2.5 Chiffrement par transposition

- C'est une méthode où toutes les lettres du message sont présentes, mais dans un ordre différent.
- Pour de très brefs messages : méthode est peu sûre car il y a peu de variantes. Exemple: Un mot de trois lettres ne pourra être transposé que dans 6 (=3!) positions différentes.
- Lorsque le nombre de lettres croît : impossible de retrouver le texte original sans connaître la clé.

2.5.1 Transposition à base matricielle

Le message en clair est écrit dans une matrice. La technique de transposition de base consiste à lire la matrice en colonne.

Exemple: Une matrice de 4 lignes et 6 colonnes

Message en clair: MESSAGE SECRET A TRANSPOSER

Message chiffré: METSESAPSETOSCRSARAEGENR

M	E	S	S	A	G
E	S	E	C	R	E
T	A	T	R	A	N
S	P	O	S	E	R

2.5.2 Transposition à base matricielle complexe

- On réarrange l'ordre des colonnes selon une permutation qui est ajoutée à la matrice pour former la clé.
- On peut générer et mémoriser simplement des permutations en prenant une clé sous forme d'un mot.
- On numérote les colonnes dans l'ordre où apparaissent les lettres du mot dans l'alphabet.
- Exemple ESPOIR correspond à la permutation 154362.
- Le texte crypté est: METSARAESCRSSETOGENRESAP

1	**2**	**3**	**4**	**5**	**6**
E	**S**	**P**	**O**	**I**	**R**
M	E	S	S	A	G
E	S	E	C	R	E
T	A	T	R	A	N
S	P	O	S	E	R

1	**5**	**4**	**3**	**6**	**2**
E	**I**	**O**	**P**	**R**	**S**
M	A	S	S	G	E
E	R	C	E	E	S
T	A	R	T	N	A
S	E	S	O	R	P

Cryptanalyse du chiffrement par transposition :

Comme les lettres du texte chiffré sont les mêmes que celles du texte en clair, une analyse statistique de la fréquence des lettres montre que chaque lettre se comporte à peu près comme dans la langue d'origine du texte. Cela donne un indice important au cryptanalyste qui peut essayer différentes techniques pour retrouver l'ordre correct des lettres. Appliquer une deuxième transposition au texte chiffré augmente grandement la sécurité. Il y a des chiffres à transposition encore plus compliqués mais les ordinateurs permettent de les casser pratiquement tous.

2.6 Difficultés de la cryptographie classique

Le problème de ce système est de communiquer les clés de chiffrement ou de trouver un algorithme de génération de clef commun aux deux partenaires :

- **La création de grandes quantités des clefs aléatoires :** n'importe quel système fortement utilisé pourrait exiger des millions de caractères aléatoires de façon régulière.
- **La distribution des clés** : une clé de longueur égale est nécessaire pour l'expéditeur et pour le récepteur.

2.7 Exercices

Exercice 2.1

1. Chiffrer le mot « BONJOUR » avec le code de César.
2. Chiffrer le mot « CODE » avec le chiffrement par décalage tel que la clé égale à 7.
3. Identifier le défaut principal de ce chiffrement.

Exercice 2.2

L'analyse des fréquences d'apparition des lettres dans un message codé montre que ceux sont les lettres K et O les plus fréquentes dans ce message. Dans un texte en français les lettres les plus fréquentes sont le A (8.4%) et le E (17.26%). Sachant que le message est en français, codé en utilisant le chiffrement par décalage sur les 26 lettres de l'alphabet, déterminer la clef et déchiffrer le message :

SVOXFYIKNKXCVKVSQEBSOKMRODOBNOCCYVNKDC

Exercice 2.3

En utilisant la correspondance : $Z_{26} \rightarrow \{0,\ldots,25\}$

1. Chiffrer le message « CRIME » avec une méthode par substitution en utilisant la clé K: $Z_{26} \rightarrow Z_{26}$ définie par :

$$K(\lambda) = \begin{cases} \lambda + 1 \; si \; \lambda \neq 7 \; et \; 25 \\ 0 \; si \; \lambda = 7 \\ 8 \; si \; \lambda = 25 \end{cases}$$

2. Donner la fonction réciproque de K

Exercice 2.4

Combien y a-t-il de cryptages par permutation différents ?

Exercice 2.5

1. Chiffrer le message « CHIFFRE DE VIGENERE » avec la méthode de Vigenère en utilisant mot BACHELIER comme clé.

2. Qu'apporte le chiffrement de Vigenère en matière de sécurité par rapport à une simple substitution alphabétique (monoalphabétique) ?

Exercice 2.6

Soit (a; b) = (7; 3) les valeurs des coefficients choisis pour le chiffrement affine.

1. Chiffrer le mot CLE
2. Déchiffrer le mot JXSG

Exercice 2.7

Chiffrer avec la méthode de Hill le message « DZ » en utilisant la matrice clé suivante :

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

Exercice 2.8

Soit le message en clair « TUER LE ROI DEMAIN A MINUIT »

1. Chiffrez le message M avec le chiffrement par transposition en utilisant la clé K=5x5.
2. Chiffrer le message M avec le chiffrement par transposition complexe en utilisant le mot clé CRIME.
3. Déchiffrer le message C=IMANIXHEPASNFNTPIXCRTROOFERSTX sachant qu'il était chiffré avec transposition tel que K=6x5 et la lecture était faite suivant les colonnes 3-2-5-1-4.

Chapitre 3 : Cryptographie moderne à clé secrète

3.1 Introduction

L'invention de l'ordinateur a bien sûr donné un essor considérable à la cryptographie et à la cryptanalyse. Contrairement à la cryptographie classique, la cryptographie moderne manipule des séquences binaires (le message à chiffrer est une suite de bits) et repose sur l'utilisation des algorithmes sophistiqués et complexes associé à des clés courtes. Il est donc constitué :

- D'un algorithme de chiffrement, supposé connu de tous, dépendant d'un paramètre qui est la clé de chiffrement. L'algorithme est fixé et public, seule la clé change.
- De la valeur de la clé de chiffrement, qui est secrète ou non suivant que l'on a affaire à un code à clé secrète (symétrique) ou à un code à clé publique (asymétrique).

3.2 Les familles de codes modernes

Les principales familles de codes modernes sont :

3.2.1 Les codes par flot

Ils imitent le chiffre de Vernam, en agissant directement sur chaque bit du texte. Le principe est de produire, à partir d'une clé courte fixée, une clé arbitrairement longue qui semble parfaitement aléatoire. Leur fonctionnement se base sur un générateur de nombres pseudo-aléatoires et un mécanisme de substitution bit à bit. Les algorithmes se basant sur ce principe sont réputés rapides mais moins résistants que les chiffrements par blocs. Ils et permettent de chiffrer et déchiffrer un message en continu, sans avoir besoin de connaitre tout le message.

Exemple d'algorithmes: RC4, Bluetooth E0/1, GSM A5/1…

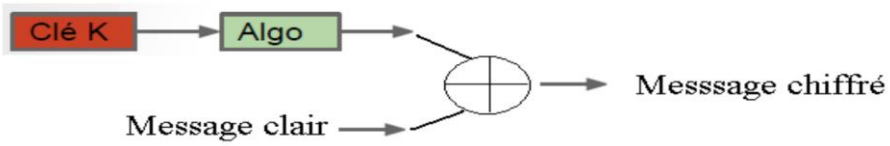


Figure 3.1 Principe de chiffrement par flot.

Propriétés :

- La séquence qui sert au chiffrement ne dépend pas du message clair, mais uniquement de la clé.
- Il est possible de chiffrer des messages de tailles variables.
- Le chiffrement et le déchiffrement s'effectuent de la même manière (XOR)
- L'impact de la modification d'une partie du message chiffré pendant la transmission du message est limité à cette partie du message déchiffré.

Une des principales caractéristiques des algorithmes de chiffrement par flot est qu'ils permettent d'atteindre un très haut niveau de performances. Ces performances s'expriment soit en termes de vitesse de chiffrement soit en termes d'efficacité matérielle. On distingue deux principaux types d'algorithmes par flot :

1. Les algorithmes adaptés à une implantation logicielle, qui peuvent atteindre des vitesses de chiffrement très élevées (plusieurs Gbits/s).
2. Les algorithmes adaptés à une implantation matérielle, dont les implantations sont efficaces en termes de taille ou de consommation électrique.

3.2.2 Les codes par blocs

Le chiffrement par blocs fonctionne différemment. Au lieu de prendre chaque bit un par un, les messages sont découpés en blocs (la taille des blocs dépend de la clé). Ensuite, chaque bloc est additionné à la clé et un traitement de type permutation, opération XOR ou autre est appliqué à chaque bloc. La clé doit être suffisamment grande pour un bon algorithme, la meilleure attaque doit coûter 2^k opérations. Les modes opératoires permettent généralement des attaques quand plus de $2^{n/2}$ blocs sont chiffrés avec une même clé.

Exemple d'algorithmes : DES, AES, IDEA, RC6…

Il existe quatre modes de chiffrement par bloc :

a) Le mode Electronic Codebook (ECB)

Dans ce mode, le message M est découpé en blocs M_i de taille fixe. Chaque bloc est alors chiffré séparément par un algorithme de chiffrement, paramétrée par une clé k_i. Ainsi un bloc de message donné sera toujours codé de la même manière. Ce mode de chiffrement est le plus simple mais il est très vulnérable aux attaques.

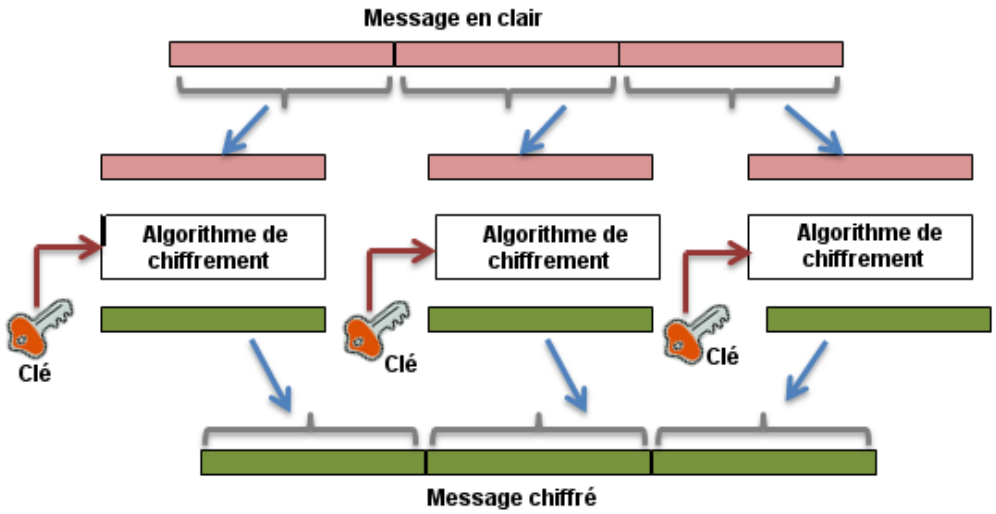


Figure 3.2 Le mode ECB

Le déchiffrement nécessite l'inverse de la fonction de codage : $D_k = E_k^{-1}$ alors : $M_i=D_k(C_i)$

b) Le mode Cipher Block Chaining (CBC)

Pour remédier aux risques de sécurité constatés dans le mode ECB, le mode CBC a été introduit pour qu'un bloc ne soit pas codé de la même manière s'il est rencontré dans deux messages différents. Il faut ajouter une valeur initiale aléatoire. Chaque bloc est d'abord modifié par XOR avec le bloc chiffré précédent avant d'être lui-même chiffré. CBC est le mode de chiffrement le plus utilisé.

Pour chiffrer un texte en mode CBC, on effectue par conséquent les opérations suivantes:
$C_1 = E\ (M_1 \oplus VI)$
$C_n = E\ (M_n \oplus C_{n-1})$, pour $n > 1$.

Et pour le déchiffrement :

$M_1 = D\ (C_1) \oplus VI$
$M_N = \mathrm{D}\ (C_N) \oplus C_{n-1}$, pour $n > 1$.

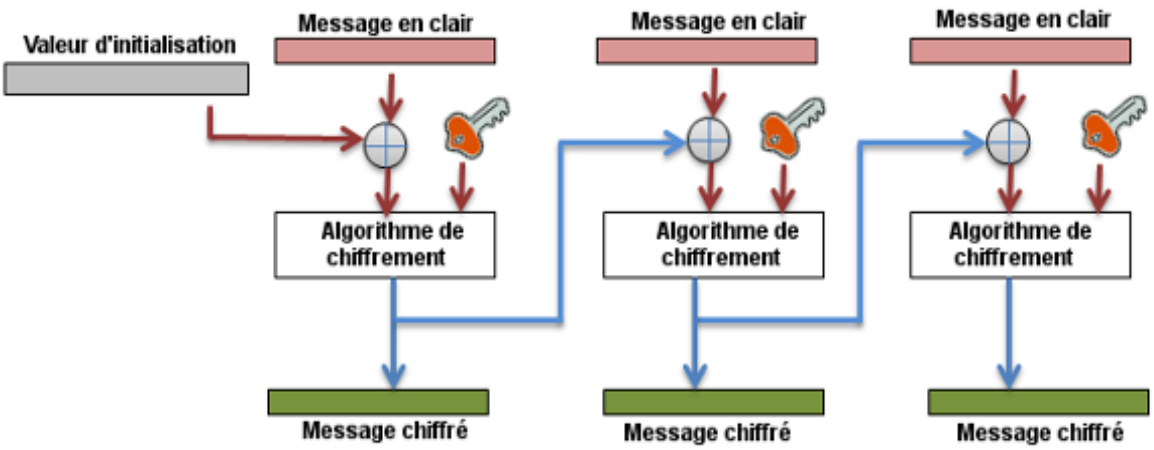


Figure3.3 Le mode CBC

c) Le mode Cipher Feedback (CFB)

Chaque chiffrement de bloc dépend du résultat du bloc précédent en manipulant un registre de décalage de la taille d'un bloc de texte en clair. Ce registre de décalage est rempli par un vecteur d'initialisation et chiffré avec l'algorithme de chiffrement utilisé. L'intérêt de ce mode est que le déchiffrement ne nécessite pas l'implémentation de la fonction : $D_k = E_k^{-1}$. Ce mode est donc moins sûr que le CBC.

Pour chiffrer un texte en mode CFB à *k* bits, on procède comme suit :

$I_1 = VI$
$I_n = (I_{n-1} \ll k) \mid C_{n-1}$, pour $n>1$
$C_n = M_n \oplus MSBk\ (E(I_n))$, pour $n \geq 1$. Tel que MSBk sont les k bits les plus significatifs

Indication : ≪ est le symbole de décalage et | et celui de la concaténation

Et pour le déchiffrement :
$I_1 = VI$
$I_n = (I_{n-1} \ll k) \mid C_{n-1}$, pour $n>1$
$M_n = C_n \oplus MSBk\ (E(I_n))$, pour $n \geq 1$.

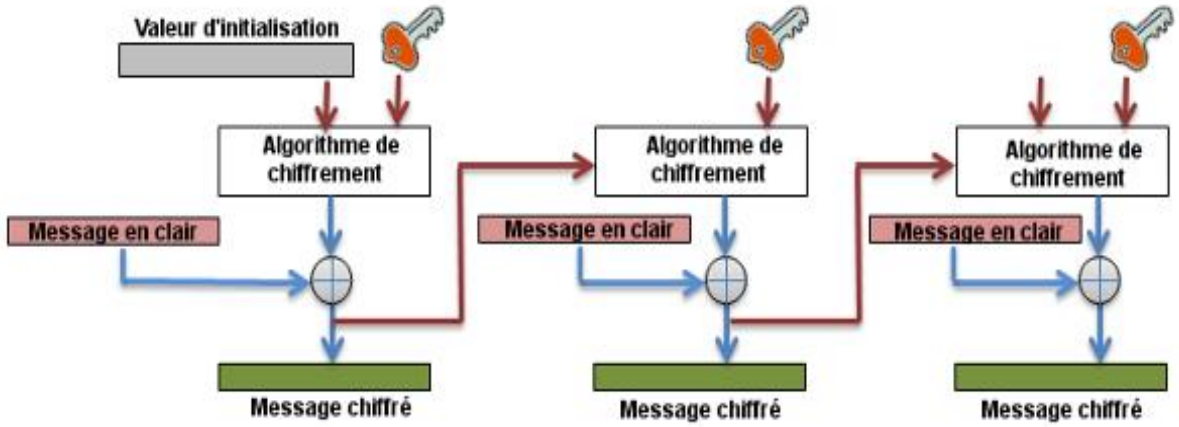


Figure3.4 Le mode CFB

d) Le mode Output Feedback (OFB)

Il manipule également un registre de décalage qui est chiffré par un algorithme de chiffrement puis modifié avec XOR avec le message en clair dont le résultat sera utilisé pour le chiffrement du bloc suivant.

- Pour chiffrer un texte en mode OFB à k bits, on effectue les opérations suivantes :
  $I_1 = VI$
  $I_n = (I_{n-1} \ll k) \mid MSBk\ (E\ (I_{n-1}))$, pour $n > 1$
  $C_n = M_n \oplus MSBk\ (E(I_n))$, pour $n \geq 1$.

- Pour le déchiffrement:
  $I_1 = VI$
  $I_n = (I_{n-1} \ll k) \mid MSBk\ (E\ (I_{n-1}))$, pour $n > 1$

$C_n = M_n \oplus MSBk\,(E(I_n))$, pour $n \geq 1$.

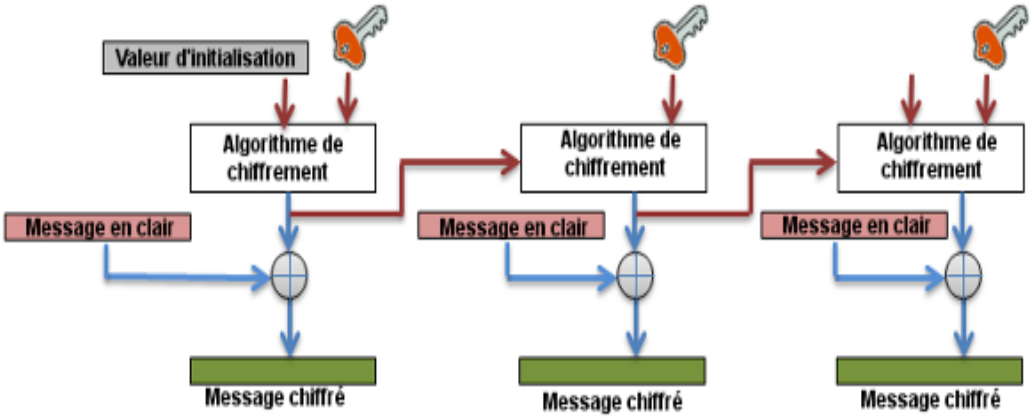


Figure 3.5 Le mode OFB

Les primitives cryptographiques peuvent être classées en deux catégories, à savoir, primitives à clé symétrique et primitives à clé publique. Dans ce chapitre, nous allons voir les méthodes cryptographiques les plus répandues pour le chiffrement symétrique.

3.3 Chiffrement symétrique

Dans la cryptographie symétrique, la clé de chiffrement est la même que la clé de déchiffrement. De ce fait, la clé doit être un secret partagé uniquement entre l'émetteur et le destinataire. Il existe plusieurs algorithmes qui fonctionnent sur ce principe : DES, IDEA, AES, RC4, RC5, Blowfish,….

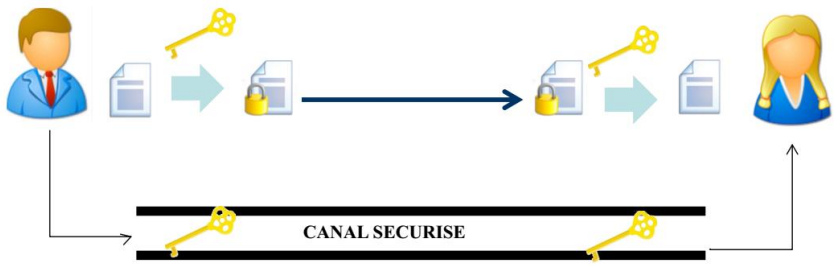


Figure3.6 Chiffrement symétrique

3.3.1 Chiffrement de Feistel

C'est la base de pratiquement plusieurs algorithmes modernes à clé secrète (en particulier DES), il est proposé par Horst Feistel (IBM) en 1973. C'est un chiffrement itératif par blocs opérant sur des blocs de 2n bits.

Principe de chiffrement :

Un bloc de texte en clair est découpé en deux, puis une transformation de ronde (fonction f) est appliquée à la partie droite, et le résultat est combiné avec la partie gauche par le ou

exclusif. Les deux moitiés sont alors inversées pour l'application de la ronde suivante. Le déchiffrement est structurellement identique au chiffrement.

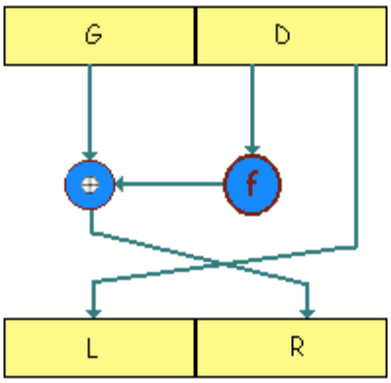


Figure 3.7 Réseau de Feistel

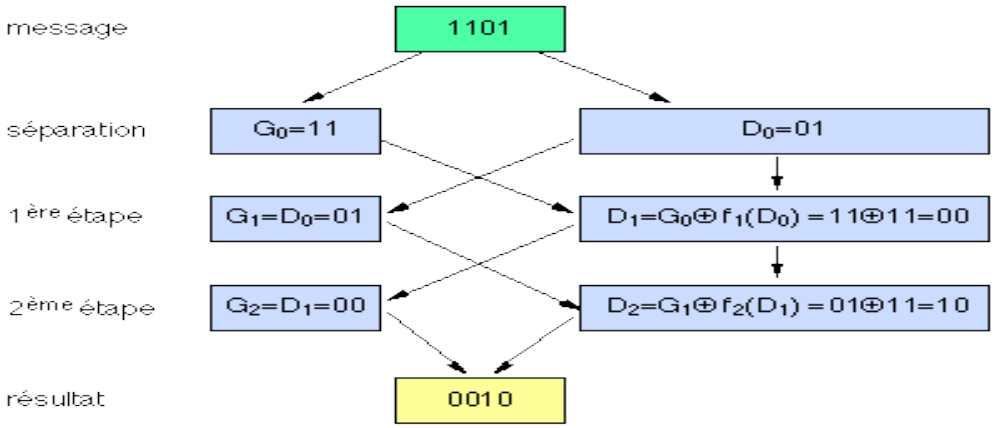
Exemple :

Pour un chiffrement Feistel à deux rondes d'un message constitué de quatre bits, nous considérerons les fonctions de ronde suivantes:

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

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

Chiffrons le message 1101 :



3.3.2 DES (Data Encryption Standard)

DES est l'algorithme de chiffrement moderne le plus populaire qui est apparu en 1970 et son premier standard est publié en 1977. Le DES est un standard américain et même international, du Gouvernement américain et il a l'aval de l'armée américaine pour le chiffrement de données de nature sensible mais non secrète. C'est un algorithme à clef secrète initialement conçu par IBM utilisait une clé de 112 bits. L'intervention de la NSA a ramené la taille de clé à 56 bits.

a) Fonctionnement de DES

L'algorithme DES transforme un bloc de 64 bits en un autre bloc de 64 bits. Il manipule des clés individuelles de 56 bits, représentées par 64 bits (avec un bit de chaque octet servant pour le contrôle de parité). Ce système de chiffrement symétrique fait partie de la famille des chiffrements itératifs par blocs, plus particulièrement il s'agit d'un schéma de Feistel. D'une manière générale, on peut dire que DES fonctionne en trois étapes :

- Permutation initiale et fixe d'un bloc (sans aucune incidence sur le niveau de sécurité)
- Le résultat est soumis à 16 itérations d'une transformation, ces itérations dépendent à chaque tour d'une autre clé partielle de 48 bits. Cette clé de tour intermédiaire est calculée à partir de la clé initiale de l'utilisateur (grâce à un réseau de tables de substitution et d'opérateurs XOR). Lors de chaque tour, le bloc de 64 bits est découpé en deux blocs de 32 bits, et ces blocs sont échangés l'un avec l'autre selon un schéma de Feistel. Le bloc de 32 bits ayant le poids le plus fort (celui qui s'étend du bit 32 au bit 64) subira une transformation.
- Le résultat du dernier tour est transformé par la fonction inverse de la permutation initiale.

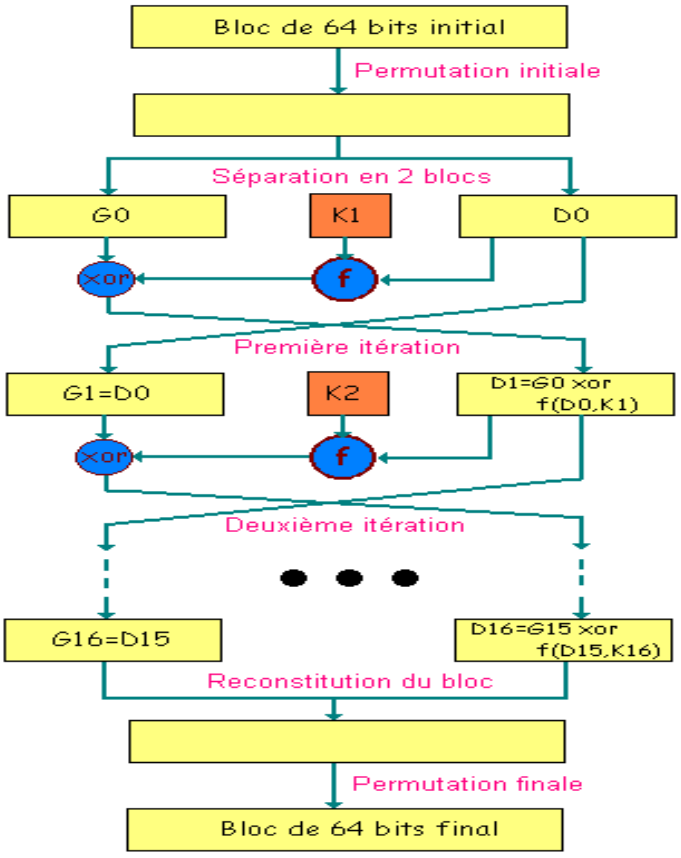


Figure 3.8 Chiffrement DES

Rappelons ci-dessous la table de permutation initiale, elle est représentée par la matrice PI

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

Figure 3.9 Matrice initiale de permutation

Cette matrice de permutation indique que le 58ème bit du bloc du texte de 64 bits se retrouve en première position, le 50ème en seconde position, etc.

Une fois la permutation initiale appliquée, le bloc de 64 bits est divisé en deux blocs de 32 bits, notés respectivement G et D (pour gauche et droite). On note G_0 et D_0 l'état initial de ces deux blocs.

Les blocs G_n et D_n sont soumis à un ensemble de transformations itératives appelés rondes ou itérations appelées aussi tours.

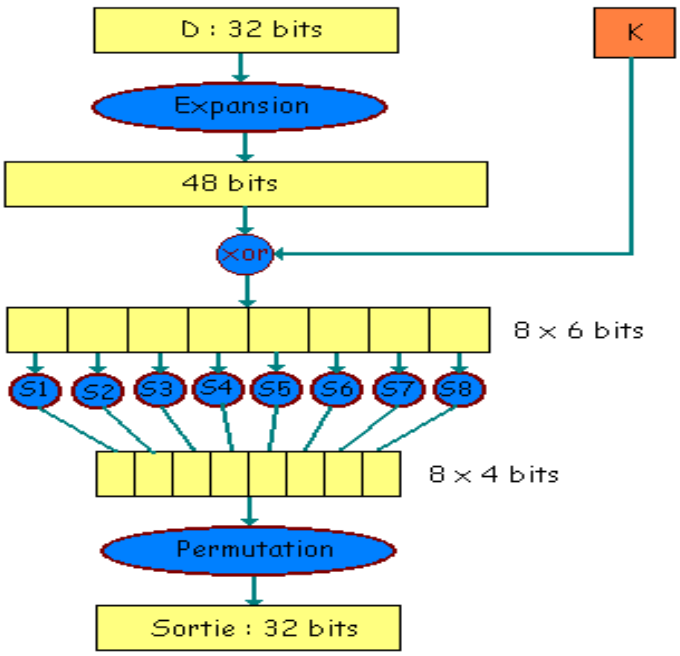


Figure 3.10 Architecture d'une ronde de DES

Les 32 bits du bloc D sont étendus à 48 bits grâce à une table (matrice) appelé table d'expansion (notée E), dans laquelle les 32 bits sont mélangés et 16 d'entre eux sont dupliqués.

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

Figure 3.11 Fonction d'expansion du DES

Ainsi, le dernier bit de D_0 devient le premier, le premier devient le second, etc.

La matrice résultante de 48 bits est appelée $E(D_0)$.

L'algorithme DES procède un OU exclusif entre la première clé K_1 et le résultat de l'application de la fonction d'expansion E sur D_0 à savoir $E(D_0)$. Le résultat obtenu est une matrice de 48 bits que nous appellerons aussi D_0

Chaque bloc de 48 bits est découpé en 8 blocs de 6 bits. Le principe consiste à transformer 6 bits en 4 bits par l'application de la substitution via les 8 fonctions de sélection appelées aussi des SBOX notées généralement S_i. Ces SBOX sont représentées sous formes de matrices fixes. Les premiers et derniers bits de chaque sous bloc détermine (en binaire) la ligne de la fonction de sélection, les autres bits déterminent la colonne.

La sélection de la ligne se fait sur deux bits et la sélection de la colonne se fait sur 4 bits. Elle s'applique sur des blocs de 6 bits pour obtenir en sortie une valeur codée sur 4 bits.

La première fonction de sélection ou SBOX est représentée par la matrice notée ci-dessous par S_1.

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

Figure 3.12 Matrice de la première fonction de sélection ou sbox.

Expliquons maintenant à travers un exemple la transformation de 6 bits données en 4 bits par la fonction S_1.

Etant donné un sous bloc égale à 110101. Le premier et le dernier bit donnent 11, soit 3 en décimal. Il indique la ligne numéro 3 dans S_1. Les 4 bits du milieu soient 1010 correspondent

à 10 en décimal, ils déterminent le numéro de la colonne qui est 10. A, l'intersection de la ligne et colonne, on trouve 3 qui correspond à 0011en binaire, c'est-à-dire les 4 bits de sortie.

Cette opération est répétée pour les autres sous blocs de 6 bits faisant appeler dans l'ordre les fonctions de sélection SBOX. Tous les résultats sont regroupés pour former un bloc de 32 bits. A ce nouveau bloc, on additionne le bloc de gauche et de l'itération précédente.

A la fin des itérations, les deux blocs G_{16} et D_{16} sont recollés, puis soumis à la permutation initiale inverse.

b) Construction des sous-clés

On fabrique à partir de la clé principale K de 64 bits, 16 sous-clés K_1,...,K_{16} à 48 bits, pris dans un certain ordre.
Dans un premier temps, les bits de parité de la clé sont éliminés afin d'obtenir une clé d'une longueur de 56 bits. Les bits de la clé initiale sont d'abord stockés dans une matrice. La première étape consiste en une permutation notée CP-1 dont la matrice est présentée ci-dessous.

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

Figure 3.13 Matrice de permutation PC-1

Cette matrice peut s'écrire sous la forme de deux matrices G_i et D_i où G_i est constitué des premiers 28 bits et D_i des derniers 28 bits. Notons G_0 et D_0 le résultat de cette première permutation. Ces deux blocs subissent ensuite une rotation à gauche de telle sorte que les bits en seconde position prennent la première position, ceux en troisième position prennent la seconde, etc. Les bits en première position passent en dernière position. Les deux blocs de 28 bits sont ensuite regroupés en un bloc de 56 bits. Celui-ci passe à travers une permutation, notée CP-2, fournissant en sortie un bloc de 48 bits, représentant la clé K_i.

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

Figure 3.14 Matrice de permutation PC-2

Le déchiffrement est structurellement est structurellement identique au chiffrement.

c) Cryptanalyse

Bien qu'aucune réelle faille n'ait été trouvée sur le DES, celui-ci utilise des clés de 64 bits, dont seulement 56 apportent de la sécurité (il y a un bit de contrôle par octet), ce qui ne résisterait pas plus de quelques heures à une machine récente. En 1990, l'algorithme DES a été cassé par la cryptanalyse différentielle. La meilleure attaque différentielle connue exige 2^{47} textes clairs choisis. Afin d'en relever simplement sa sécurité, le Triple-DES fut adopté en 1998.

d) Triple DES

Le Triple DES (aussi appelé 3DES) est un algorithme de chiffrement symétrique enchainant trois applications successives de l'algorithme DES sur le même bloc de données de 64 bits, avec 2 ou 3 clés DES différentes. Cette utilisation de trois chiffrements DES a été développée par Walter TUCHMAN. Elle permet d'augmenter la sécurité du DES, toutefois, il a l'inconvénient majeur de demander également plus de ressources pour le chiffrement et le déchiffrement.

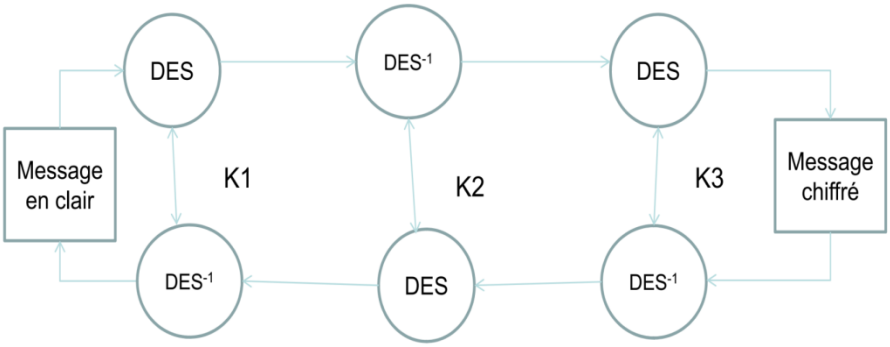


Figure 3.15 Triple DES

3.3.4 AES (Advanced Encryption Standard)

Une mise en concurrence pour AES a été lancée le 2 janvier 1997 par le NIST (National Institute od Standards ans Technologies) et le choix de la solution a eu lieu le 3 octobre 2000. C'est l'algorithme *Rijndael* développé par Joan Daemen et Vincent Rijmen de l'université catholique de Louvain qui a été retenu.
AES est la norme standard recommandée pour le chiffrement à clé secrète. Le principe de l'AES est très proche du DES mais se révèlera plus performant. C'est un système cryptographique constitué d'une suite d'opérations de permutation et de substitution mais contrairement à DES ce n'est pas un schéma de Feistel.

a) Description de l'algorithme AES

C'est un chiffrement symétrique itéré qui utilise des blocs de 128 bits, 192 bits ou 256 bits et des clés de 128 bits, 192 bits ou 256 bits. Le nombre des itérations varie entre 10, 12 et 14 où chaque ronde utilise une clé générée à partir de la clé principale.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	$k_{0,6}$	$k_{0,7}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	$k_{1,6}$	$k_{1,7}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	$k_{2,6}$	$k_{2,7}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$	$k_{3,5}$	$k_{3,6}$	$k_{3,7}$

Figure 3.16 Représentation de l'état et de la clé

L'état ou le bloc en clair est représenté par une matrice. Le nombre de ligne de cette matrice égale toujours à 4 mais le nombre de colonne égale se change selon la taille du bloc : 4 colonnes correspond à un bloc de 128 bits, 6 colonnes à un bloc de 192 bits et 8 colonnes à un bloc de 256 bits. La représentation des clés se fait de la même façon comme le montre la figure ci-dessus.

Le nombre de rondes varie suivant la taille des blocs et la taille des clés à la fois. Si par exemple la taille du bloc égale à 192et la taille de la clé égale aussi à 256 on aura 14 rondes à faire selon la figue suivante :

	$\|k\| = 128$	$\|k\| = 192$	$\|k\| = 256$
blocs 128	10	12	14
blocs 192	12	12	14
blocs 256	14	14	14

Figure 3.17 Nombre de ronde de l'AES

Dans le reste de la description, nous supposons que la taille du bloc et la taille de la clé sont égalent à 128 (16 octets) donc le chiffrement du bloc nécessite 10 rondes et 10 sous-clés.

La matrice en entrée est d'abord modifiée par un ou exclusif avec la matrice principale de la clé. Les 16 octets en sortie vont passer par quatre transformations de ronde :

1. **SubBytes (substitution)**: passage dans une S-box
2. **ShiftRows (rotation)**: décalage des lignes.
3. **MixColumns (mélange)**: mélange des colonnes (sauf dernier tour)
4. **AddRoundKey (Ajout de la sous-clé)**: XOR avec la sous-clé de ronde.

La dernière ronde (dans ce cas la dixième ronde) ne subit pas la transformation de mélange (MixColumns).

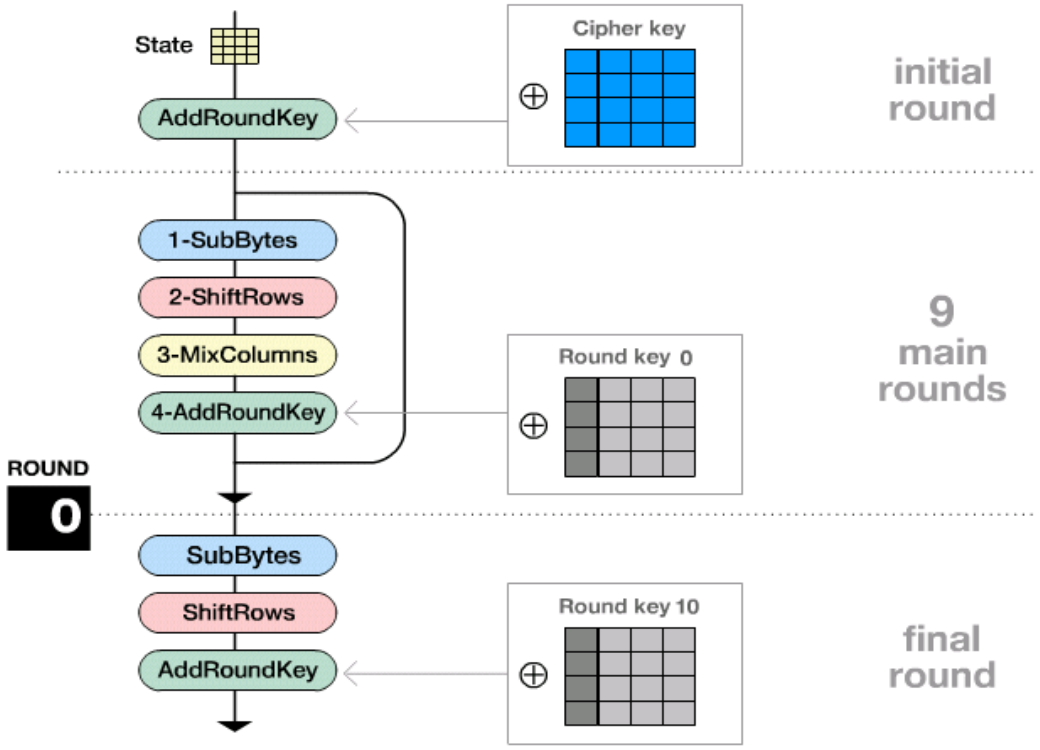


Figure 3.18 Structure générale de l'AES

Nous allons maintenant décrire les différentes transformations de ronde.

Transformation SubBytes(): Il s'agit d'une transformation non linéaire appliquée indépendamment à chacun des octets de l'état en utilisant une table de substitution (Sbox).

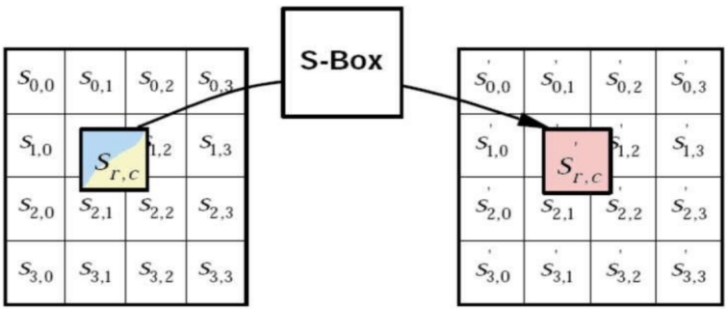


Figure 3.19 Substitution avec S-BOX

Exemple : Si la valeur à substituer égale à 95 alors le numéro 9 représente la ligne de S-BOX et le 5 représente sa colonne. L'intersection de cette ligne et cette colonne donne le résultat de la substitution **2a** comme il est montré dans la figure suivante :

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

Figure 3.20 S-BOX

Il est à noter que SubBytes() est une transformation bijective sur le corps GF(28).

Transformation ShiftRows(): Elle correspond à une permutation cyclique des octets sur les lignes de l'état. Le décalage des octets correspond à l'indice i de la ligne considérée (0<=i< 4)

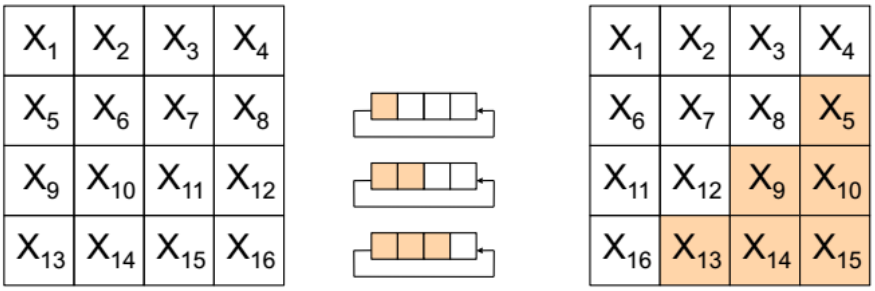


Figure 3.21 Rotation

ShiftRows() est une transformation linéaire.

Transformation MixColumns() : Cette transformation est appliquée à un état colonne après colonne.

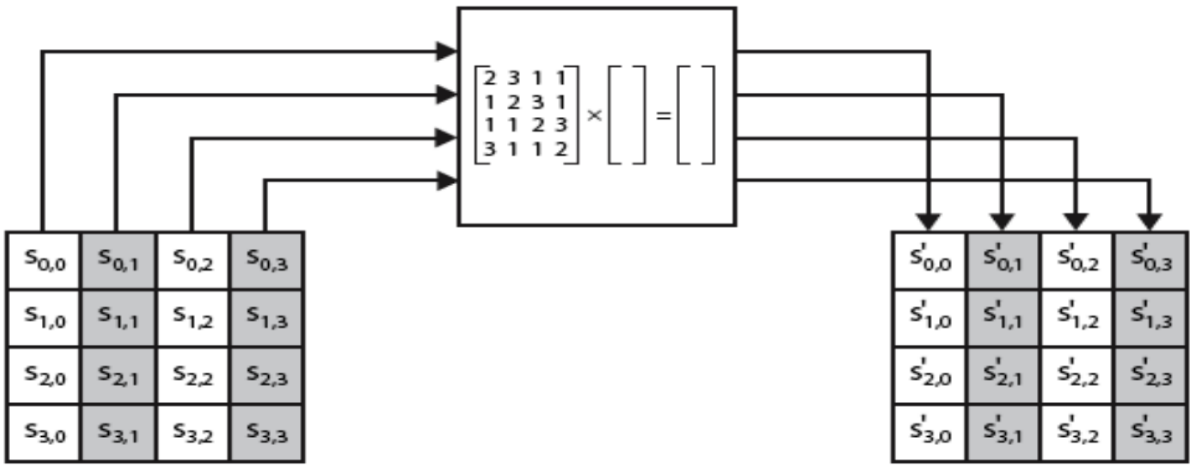


Figure 3.22 MixColumns()

C'est une transformation linéaire : un produit matriciel utilisant les 4 octets d'une colonne. Les colonnes sont traitées comme des polynômes dans GF(28) et multipliées modulo x4 + 1 avec les polynômes fixes donnés figure suivante :

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$
$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$
$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

Transformation AddRoundKey(): C'est l'ajout de la clé de ronde (ou de la clé lors de la ronde initiale) à l'état considéré (l'addition est un ou exclusif). Un XOR (au niveau bit) est appliqué entre chacun des octets de l'état et de la clé de ronde.

b) Construction des sous-clés

La clé doit être modifiée avant de passer à l'étape de la transformation AddRoundKey. La première colonne de la clé une fois modifiée va correspondre à la dernière colonne, décalée du bas vers le haut. Ensuite, elle va être modifiée avec la S-BOX, puis une opération XOR va être réalisée entre le résultat obtenu, la première colonne de la clé d'origine et la colonne x (x correspond au numéro De la ronde) de la matrice Rcon (tableau fixe de constantes de tours). Ensuite, les trois autres colonnes restantes seront des opérations XOR entre la colonne de la clé d'origine et la dernière colonne ajoutée à la nouvelle clé.

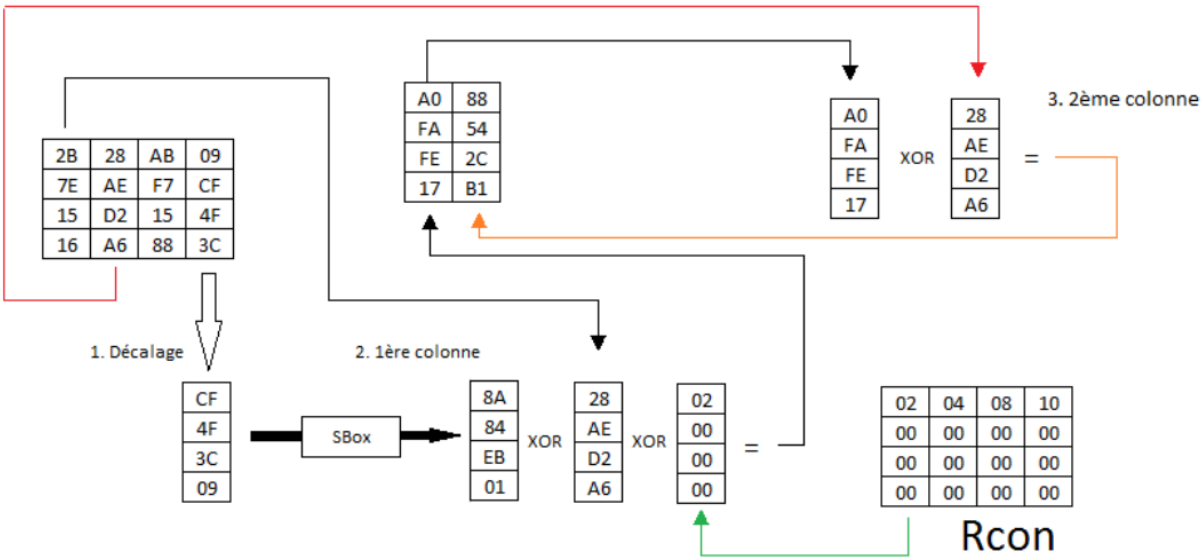


Figure3.23 Schéma de fonctionnement de l'étape KeyExpansion

c) Déchiffrement AES

Les clefs de ronde sont utilisées dans l'ordre inverse de celui du chiffrement. On notera que l'ordre des transformations diffère de celui du Cipher. Le déchiffrement consiste à appliquer dans l'ordre inverse du chiffrement les transformations inverses correspondantes.

InvShiftRows(): est l'inverse de la transformation ShiftRows.

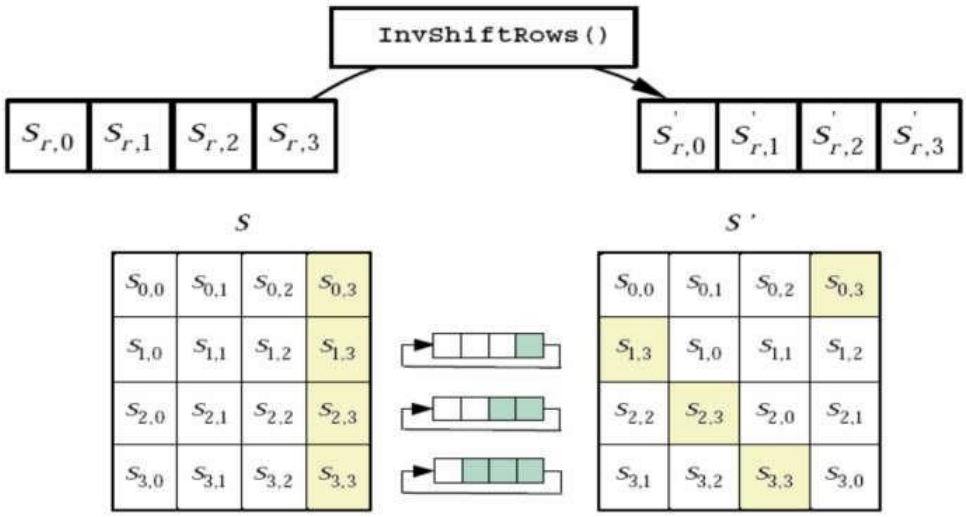


Figure 3.24 InvShiftRows ()

InvSubBytes() est l'inverse de la transformation SubBytes().

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

Figure 3.25 InvSubBytes()

InvMixColumns():

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$
$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$
$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

d) Caractéristiques et points forts de l'AES

Le choix de cet algorithme répond à de nombreux critères que nous pouvons citer :

- Sécurité et résistance contre une éventuelle cryptanalyse.
- Puissance de calcul qui entraine une grande rapidité de traitement.
- Possibilité d'être implémenté sur des ressources limitées.
- Possibilité d'implémenter AES aussi bien sous forme logicielle que matérielle

Si l'on se réfère à ces critères, on constate 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.

e) Cryptanalyse

L'AES peut être menacé par les attaques suivantes :

Attaques par dictionnaires: En ce qui concerne l'AES, il y a 2^{128} clés possibilités (dans la version minimale) pour cryptanalyser AES, chose qui est théoriquement possibles mais quasi impossible à réaliser pratiquement.

Attaques par cryptanalyse différentielle: L'attaquant doit choisir des textes clairs présentant une différence fixe, et calculer les chiffrés (en ayant accès au système) et Ieurs différences puis assigne des probabilités à certains types de clés. Plus le nombre d'essais augmente, plus la probabilité de découvrir la bonne clé devient grande. Dans le cas du DES simple, cette attaque nécessite 2^{47} textes clairs et 2^{47} chiffrements pour retrouver la clé. Néanmoins, les textes clairs doivent être soigneusement choisis. L'AES est lui résistant à ce type d'attaque.

Attaques par cryptanalyse linéaire: Pour ce type d'attaques, on utilise des approximations linéaires pour décrire les opérations conduisant au résultat chiffré. Comme précédemment, plus le nombre d'essais augmente, plus la probabilité de découvrir la bonne clé augmente. Cette attaque est actuellement la plus performante puisqu'elle ne nécessite que 2^{43} textes clairs et 2^{43} chiffrements pour retrouver une clé DES (simple). L'AES est lui résistant à ce type d'attaque.

3.4 Problèmes de chiffrement symétrique

Distribution des clés : Les algorithmes de chiffrement symétrique se fondent sur une même clé pour chiffrer et déchiffrer un message. L'un des problèmes de cette technique est que la clé, qui doit rester totalement confidentielle, doit être transmise au correspondant de façon sûre.

Gestion des clés : La mise en œuvre peut s'avérer difficile, surtout avec un grand nombre des intervenants car il faut autant de clés que des intervenants. Si N intervenants veulent s'échanger des informations, chaque intervenant doit avoir une clé différente avec chacun des autres intervenants donc le nombre de clés est N*(N-1)/2.

3.5 Exercices

Exercice 3.1

1. Quel est l'avantage et l'inconvénient d'un chiffrement symétrique ?
2. Combien de clés sont nécessaires pour que cinq personnes puissent communiquer via un chiffrement symétrique?

Exercice 3.2

Soit E_k une fonction de chiffrement binaire par bloc de taille fixe (4 bits) tel que:
A tout message en clair m_i on associe un chiffré $c_i = E_k(m_i)$

Le tableau suivant donne la correspondance entre les messages m_i et leurs chiffrés c_i :

m_i	0000	0001	0010	0011	0100	0101	0110	0111
c_i	0001	1001	0000	1000	0011	1011	0010	1010

m_i	1000	1001	1010	1011	1100	1101	1110	1111
c_i	0101	1101	0100	1100	0111	1111	0110	1110

1. Donner le chiffré du message *M* suivant : *M*= 10110001, avec les modes d'opérations :
   - ECB
   - CBC (valeur initiale IV= 1010)
   - CFB (valeur initiale IV= 1010)
   - OFB (valeur initiale IV= 1010)

Exercice 3.3

On considère un chiffrement de Feistel à deux rondes sur des chaînes de 8 bits avec deux fonctions *f*1 et *f*2. On pose: $f1(x) := x \oplus 1011$ et $f2(x) := \bar{x} \oplus 0101$ pour toute chaîne *a* de 4 bits.

- Calculer le chiffré de M =11010011 à travers le schéma de Feistel.

Exercice 3.4

L'algorithme MiniDES est un chiffrement par bloc suivant un schéma de Feistel. Il chiffre des messages de 16 bits en un autre bloc de 16 bits avec une clé de longueur 12 bits. Il manipule des clés individuelles de de rondes 12 bits.

- Calculer le résultat de la première ronde du message M = A0E0 avec la clé de ronde K1= 07E

Exercice 3.5

1. L'algorithme AES est un chiffrement par bloc itératif qui repose sur 4 opérations, les quelles ?
2. Donner le résultat de l'opération « SubBytes » de : 3d (code hexadécimal)
3. Quel est le résultat de l'opération « ShiftRows » sur le bloc suivant ?

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

4. Calculer dans le corps AES, les quantités suivantes (code hexadécimal) :
   - 66 + fa
   - 02 * bf
   - 11* de + 02 * bf

On donne :

- Le polynôme de Rinjdael : $R(x)=x^8+x^4+x^3+x+1$
- La table de substitution est fournie dans le cours AES.

Exercice 3.6

Soit la clé MiniAES de 16 bits donnée en hexadécimal par k=B2EA

- Construire les deux premières clés de ronde suivant les formules suivantes :

$K^{(1)} = SubByte(K^{(0)}) \oplus \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ et $K^{(2)} = SubByte(K^{(1)}) \oplus \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$

La transformation *SubByte* est présentée par le tableau suivant :

Input	Output		Input	Output
0000	1110		1000	0011
0001	0100		1001	1010
0010	1101		1010	0110
0011	0001		1011	1100
0100	0010		1100	0101
0101	1111		1101	1001
0110	1011		1110	0000
0111	1000		1111	0111

Chapitre 4: Cryptographie à clé publique

4.1 Introduction

C'est en 1976 que Whitfield Diffie et Martin Hellman, de l'Université Stanford, proposent un principe de chiffrement entièrement nouveau : ***la cryptographie à clé publique, ou asymétrique.***

4.2 Principe de la cryptographie

Un système cryptographie à clé publique est en fait basé sur **deux clés** :

a) **Clé publique** pour chaque intervenant
- Cette clé peut être connue de tous. Par exemple, disponible dans un répertoire accessible publiquement.
- Toute personne connaissant cette clé peut envoyer un message chiffré au propriétaire de cette clé.
- Les clés publiques doivent être distribuées de façon authentifiée.

b) **Clé privée** pour chaque intervenant
- Doit demeurer confidentielle.
- Liée (mathématiquement) à la clé publique correspondante.
- Permet de déchiffrer tout message chiffré avec la clé publique correspondante

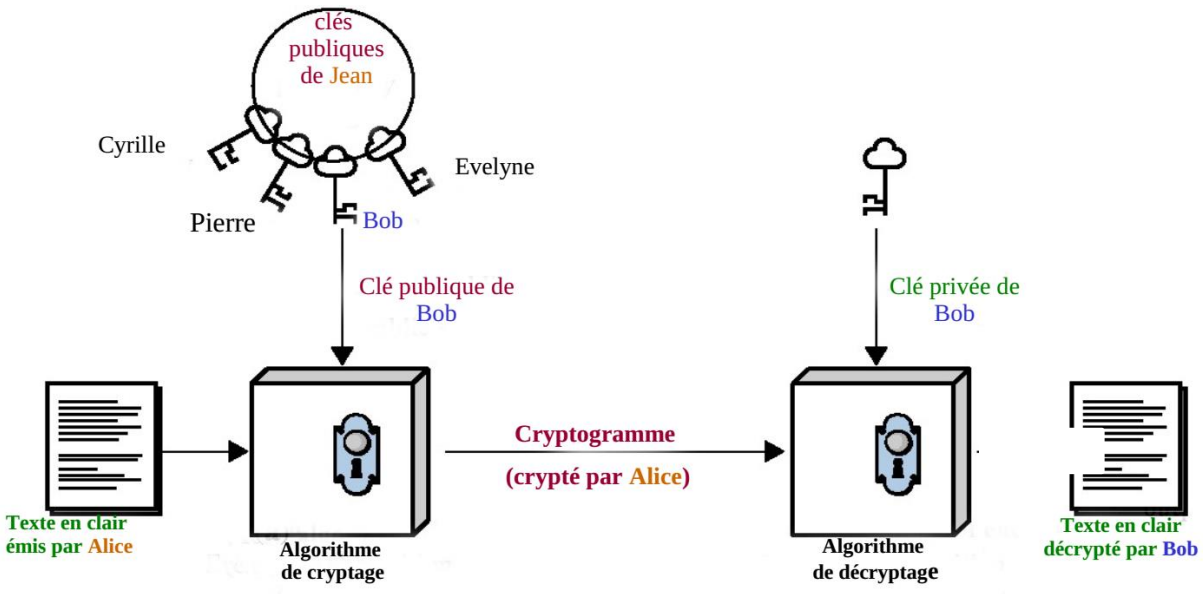


Figure 4.1 Chiffrement à clé publique

Exemple:

Alice doit recevoir un message de Bob, mais elle ne fait pas confiance au facteur qui pourrait ouvrir sa lettre. Comment peut-elle être sûre de recevoir ce message sans qu'il soit lu ?
Alice va d'abord envoyer à Bob un cadenas ouvert, dont elle seule possède la clé. Ensuite, Bob va placer son message dans une boîte, qu'il fermera à l'aide de ce cadenas, avant de l'envoyer à Alice. Le facteur ne pourra donc pas ouvrir la boîte, puisque seule Alice possède la clé !

4.3 Fonction à sens unique :

Etant donnée une fonction f, il est possible connaissant x de calculer «facilement» f (x) ; mais connaissant un élément de l'ensemble image de f , il est «difficile» ou impossible de trouver son antécédent.

Dans le cadre de la cryptographie, posséder une fonction à sens unique qui joue le rôle de chiffrement n'a que peu de sens. En effet, il est indispensable de trouver un moyen efficace afin de pouvoir déchiffrer les messages chiffrés. On parle alors de fonction à sens unique avec trappe secrète.

Prenons par exemple le cas de la fonction f (non bijective) suivante :

$f : x \rightarrow x^3 \pmod{100}$

- Connaissant x, trouver y = f (x) est facile, cela nécessite deux multiplications et deux divisions.
- Connaissant y image par f d'un élément x (y = f (x)), retrouver x est difficile.

Tentons de résoudre le problème suivant : trouver x tel que $x^3 \equiv 11 \pmod{100}$.

On peut pour cela :

- soit faire une recherche exhaustive, c'est-à-dire essayer successivement 1, 2, 3, ..., 99, on trouve alors : $71^3 = 357\,911 \equiv 11 \pmod{100}$,
- soit utiliser la trappe secrète : $y \rightarrow y^7 \pmod{100}$ qui fournit directement le résultat ! $11^7 = 19\,487\,171 \equiv 71 \pmod{100}$.

La morale est la suivante : le problème est dur à résoudre, sauf pour ceux qui connaissent la trappe secrète.

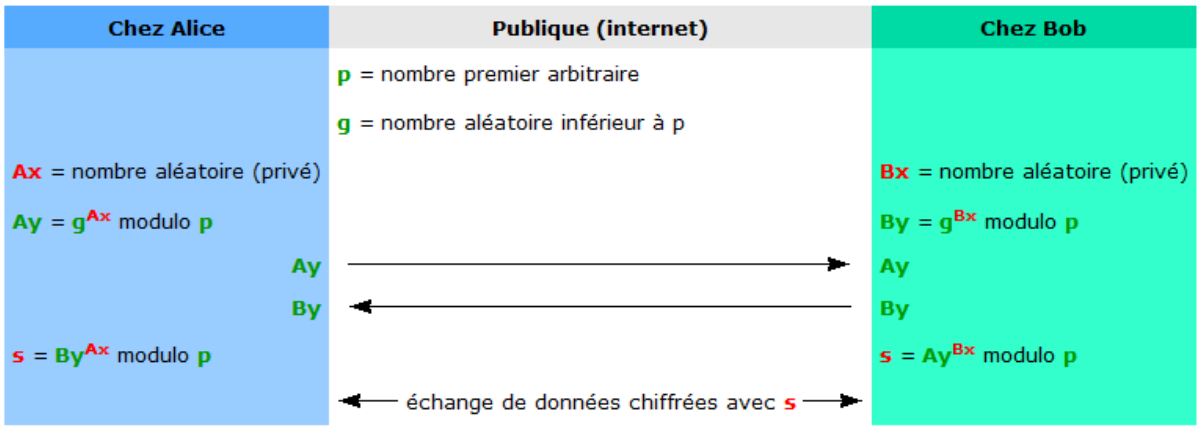
Comment imaginer une fonction qui soit à sens unique pour tout le monde sauf pour son créateur qui peut l'inverser grâce à la connaissance d'une information particulière (la clé) ?

Ce sont Diffie et Hellman qui ont les premiers donné une réponse à cette question.

4.4 Le protocole de Diffie et Hellman

Le but de l'algorithme Diffie-Hellman est de créer un secret commun aux personnes qui veulent communiquer et d'utiliser ce secret pour chiffrer les données échangées.

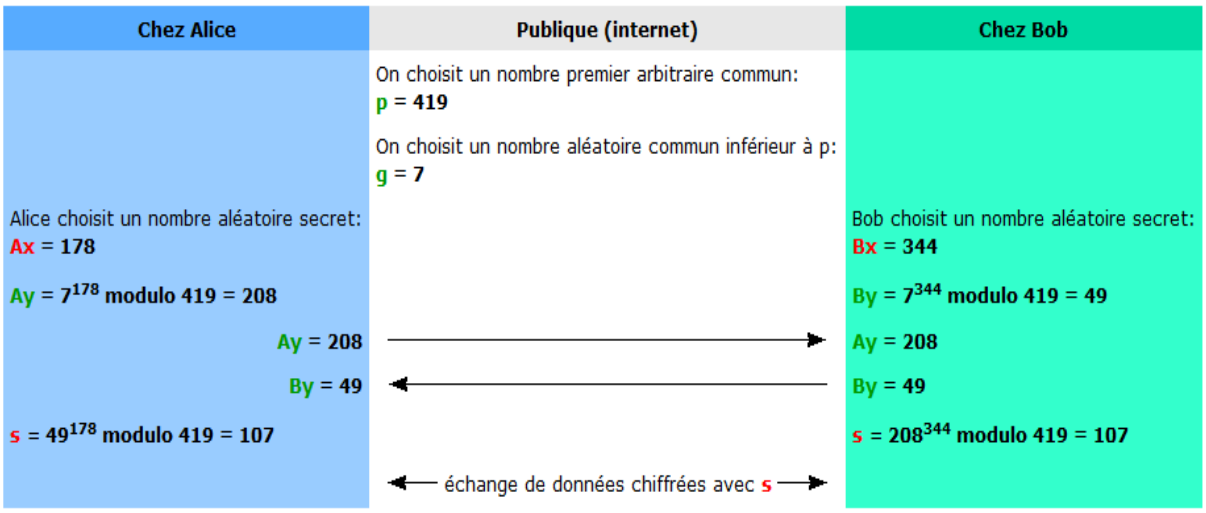
Imaginons qu'Alice et Bob veuillent communiquer.



Tout ce qui est en vert est publique (diffusé sur internet). Tout ce qui est en rouge est privé.

Un espion sera incapable de calculer **s** à partir de **p** et **g**, car il ne connait ni le nombre aléatoire **Ax** choisi par Alice, ni le nombre aléatoire **Bx** choisi par Bob. **Ay** et **By** échangés entre Alice et Bob ne l'aideront pas non plus à calculer **s**.

Exemple :



Alice et Bob ont calculé le même secret commun: **107**. On se sert de 107 pour chiffrer les données échangées (Dans la pratique, on utilise des nombres beaucoup plus grands).

4.5 Le système RSA

Le système de chiffrement RSA a été inventé par trois mathématiciens : **R**ivest, **S**hamir et **A**dleman, en 1977 (On retrouve le sigle RSA dans les noms des inventeurs) pour répondre

aux concepts de Diffie-Hellman. La sécurité repose sur la difficulté de factoriser un nombre qui est le produit de deux nombre premiers très grands. Les clés publique et privée sont générées à partir de deux nombres premiers très grands (plus de 100 chiffres)

4.5.1 Principe

a) Créer une paire de clés
- Soit deux grands nombres premiers **p** et **q**
- Soit **n = p x q**.
- Prendre un nombre **e** qui n'a aucun facteur en commun avec φ (N) = **(p-1)(q-1)**.
- Calculer **d** tel que : **e d mod (p-1)(q-1) = 1**
- Le couple **(e,n)** constitue **la clé publique** et le couple **(d,n)** constitue **la clé privée**.

b) Chiffrement de M

$$C = E_{e,N}(M) = M^e \bmod N$$

c) Déchiffrement de C

$$M = D_{d,N}(C) = C^d \bmod N$$

Exemple:

Prenons comme exemple p = 7, q = 13. Pour calculer n, il faut multiplier p et q, ce qui nous donne n = 91. Pour calculer φ(n), il faut faire (7-1) * (13-1) ce qui donne 72.

Maintenant, il faut choisir l'exposant e, qui doit être premier avec φ(n). Prenons donc par exemple e = 5 (PGCD (5, 72) = 1 donc premiers entre eux). Voici donc la clé publique e = 5 et n = 91.

Pour calculer d, il suffit de faire 5^{-1} mod 72, ce qui donne 29. Voici donc La clé privée d=29 et n=91.

Pour vérifier que d est correct, il suffit de calculer e * d mod φ(n) = 1. Dans cet exemple cela revient à calculer 5 * 29 mod 72 = 1

Soit le message à crypter M = SECRET. Pour chiffrer M il faut d'abord le transformer en une suite de chiffres, par exemple en remplaçant les lettres par des chiffres selon la position de la lettre dans l'alphabet.

S	E	C	R	E	T
19	05	03	18	05	20

En appliquant la formule de chiffrement sur chaque lettre comme suit :

$C_1 = M_1^e \bmod n = 19^5 \bmod 91 = 80$
$C_2 = M_2^e \bmod n = 5^5 \bmod 91 = 31$
$C_3 = M_3^e \bmod n = 3^5 \bmod 91 = 61$

$C_4 = M_4^e \bmod n = 18^5 \bmod 91 = 44$
$C_5 = M_5^e \bmod n = 5^5 \bmod 91 = 31$
$C_6 = M_6^e \bmod n = 20^5 \bmod 91 = 76$

Alors : C= 803161443176

Pour déchiffrer C il suffit d'utiliser que la clé privée (d, N)= (29,91) dans la formule de déchiffrement.

4.5.2 Discussion

Bien que l'algorithme soit simple, il est très coûteux en termes de ressources.

Par exemple, une carte à puce peut calculer :

- Une itération AES ou DES en 1 à 5 millisecondes (voir nanosecondes avec un accélérateur matériel)
- Un déchiffrement RSA (utilisant la version CRT et un accélérateur matériel) en 100-300 millisecondes !!!

Personne n'a encore trouvé comment:

- Calculer **d** avec l'aide de **e** sans avoir une connaissance de la factorisation de **N**
- Déchiffrer le cryptogramme sans la connaissance de la clé privée **d**.

En pratique, RSA n'est pas utilisé pour le chiffrement des messages, mais plutôt pour l'échange sécuritaire de clés de sessions pour le chiffrement symétrique.

4.5.3 Système hybride

- Le message est chiffré rapidement grâce à une clé secrète DES ou AES, qui ne sert que pour un message. C'est une clé de session.
- La clé de session est chiffrée grâce à un algorithme de cryptographie à clé publique tel que RSA.

Problématique : Comment Bob peut être sûr que le message provient bien d'Alice ?

4.6 La signature numérique

La signature numérique ou électronique est un mécanisme qui permet d'authentifier un message, autrement dit de prouver qu'un message provient bien d'un expéditeur donné, à l'instar d'une signature sur un document papier.

Elle permet d'assurer l'intégrité du message ainsi que la non-répudiation, c'est-à-dire qu'elle permet de vérifier l'origine du message.

Il est possible de combiner le chiffrement et la signature numérique et, par conséquent, d'assurer la confidentialité et l'authentification.

4.7 Protocole de signature RSA

La signature RSA est la notion duale du chiffrement RSA.

La signature du message M:

$$Signature = D_{d.N}(M) = M^d \bmod N$$

Puisque le signataire est le seul possédant sa clé privée, il est le seul pouvant signer.

La vérification de la signature:

$$Message = E_{e,N}(Signature) = Signature^e \bmod N$$

Puisque tous les intervenants peuvent obtenir la clé publique, ils peuvent tous vérifier la signature.

4.8 Robustesse du chiffrement asymétrique

La robustesse des cryptosystèmes à clés publiques repose sur deux piliers :

- La confidentialité de la clé privée de celui qui l'utilise ; en effet la divulgation de cette clé privée réduit à néant la protection offerte par le système.
- Les résultats de la théorie des nombres, ou plutôt l'absence de tels résultats, qui nous dit que la factorisation de très grands nombres est un problème difficile, ainsi d'ailleurs que le problème du logarithme discret (ici, le terme difficile doit être entendu comme insoluble en pratique). C'est-à-dire que tous ces systèmes sont à la merci d'un progrès inattendu de la théorie mathématique, qui viendrait par exemple offrir aux cryptanalystes un nouvel algorithme de factorisation rapide.

Comme pour le chiffrement symétrique, les nombres choisis comme bases du système doivent être suffisamment grands pour décourager les attaques par force brute, et la réalisation des programmes doit être correcte.

4.9 Fonction de hachage

Il s'agit de la troisième grande famille d'algorithmes utilisés en cryptographie. Une fonction de hachage *h* est une fonction qui, à partir d'un message clair de longueur quelconque doit être transformé en un message de longueur fixe inférieure à celle de départ. Le message réduit portera le nom de "Haché" ou de "Condensé". L'intérêt est d'utiliser ce condensé comme empreinte digitale du message original afin que ce dernier soit identifié de manière univoque.

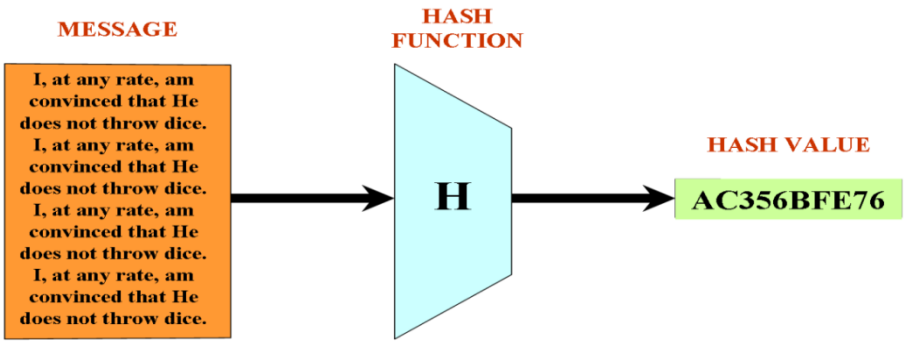


Figure 4.2 Fonction de hachage

4.9.1 Principe

- Une fonction de hachage calcule l'empreinte y d'un message x. Cette fonction doit être une fonction à sens unique.
- Elle doit aussi être très sensible pour qu'une petite modification du message entraîne une grande modification de l'empreinte.
- En envoyant le message accompagné de son empreinte, le destinataire peut s'assurer de l'intégrité du message en recalculant le résumé à l'arrivée et en le comparant à celui reçu.
- Si les deux résumés sont différents, cela signifie que le fichier n'est plus le même que l'original : il a été altéré ou modifié par une tierce personne

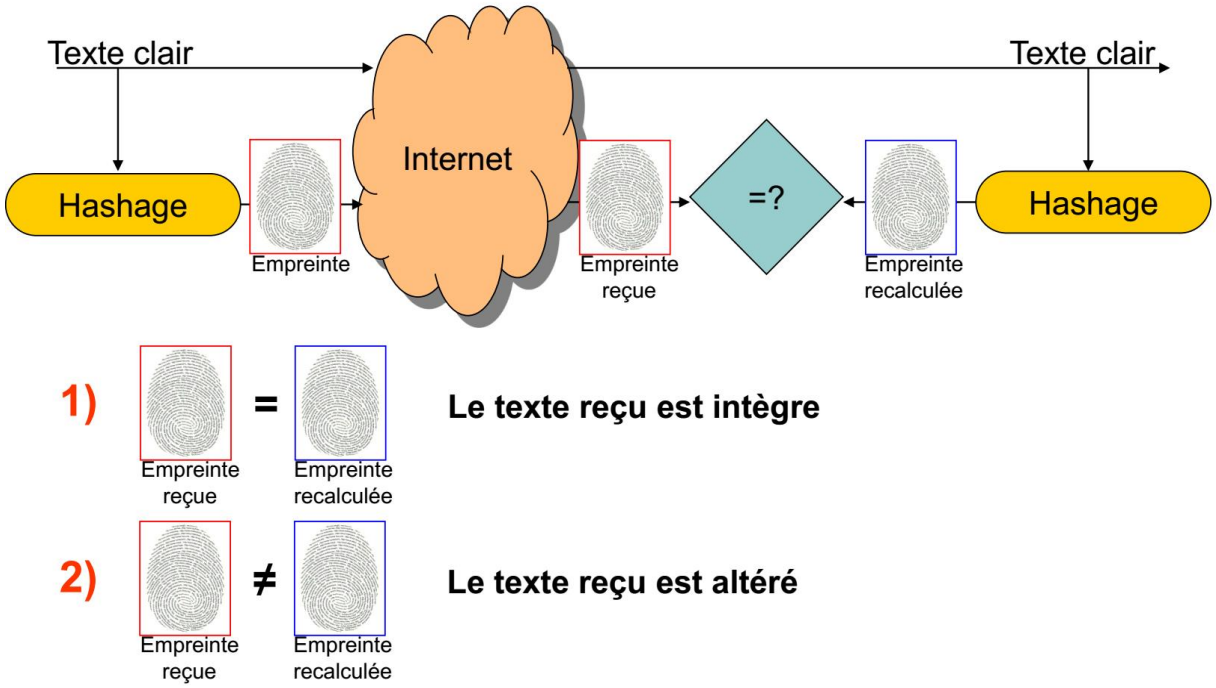


Figure 4.3 Vérification de l'intégrité

4.9.2 Efficacité de l'opération

- Généralement pour toute fonction h avec entrée x, le calcul de $h(x)$ est une opération rapide.
- Les fonctions de hash par calcul sont beaucoup plus rapides qu'un cryptage symétrique.

4.9.3 Propriétés des fonctions de hachage

Pour être un outil cryptographique efficace, la fonction d'hachage doit avoir les propriétés suivantes :

- **Résistance à la pré-image:** pour toute valeur de hachage, il devrait être difficile de trouver un messag m tel que $h = hash(m)$; cette notion est liée à la notion de fonction à sens unique ; les fonctions qui n'ont pas cette propriété sont vulnérables aux attaques de pré-image ;
- **Résistance à la seconde pré-image:** pour toute entrée m_1, il devrait être difficile de trouver une entrée différente m_2 telle que $hash(m_1) = hash(m_2)$; les fonctions qui n'ont pas cette propriété sont vulnérables aux attaques de seconde pré-image;
- **Résistance aux collisions :** il doit être difficile de trouver deux messages différents m_1 et m_2 tels que $hash(m_1) = hash(m_2)$; une telle paire de messages est appelée une collision de hachage cryptographique ; pour obtenir cette propriété, il faut une valeur de hachage au moins deux fois plus longue que celle requise pour obtenir la résistance à la pré-image ; si la clé n'est pas assez longue, une collision peut être trouvée par une attaque des anniversaires.

4.9.4 MD5 (Message Digest 5)

La fonction de hachage MD5 est une version améliorée de MD4, les deux algorithmes ont été developpé par Ron Rivest, un des créateurs de RSA. MD5 utilise des blocs de 512 bits et génère des messages chiffrés de 128 bits. Chaque bloc est découpé en 16 sous-blocs de 32 bits (A, B, C et D).

a) Principe de fonctionnement

- MD5 prend 4 tampons de 32 bits en entrée (en hexadécimal) :
- MD5 est composé de quatre rondes qui exécutent chacune 16 opérations.
- Pour chaque ronde, une seule fonction prenant 3 arguments codés sur 32 bits et renvoyant une valeur sur 32 bits est utilisée pour les 16 opérations.
- A l'issue des 4 rondes, les nouvelles valeurs de A, B, C, D sont ajoutés aux anciennes.
- L'empreinte finale de 128 bits est la concaténation du dernier jeu ABCD.

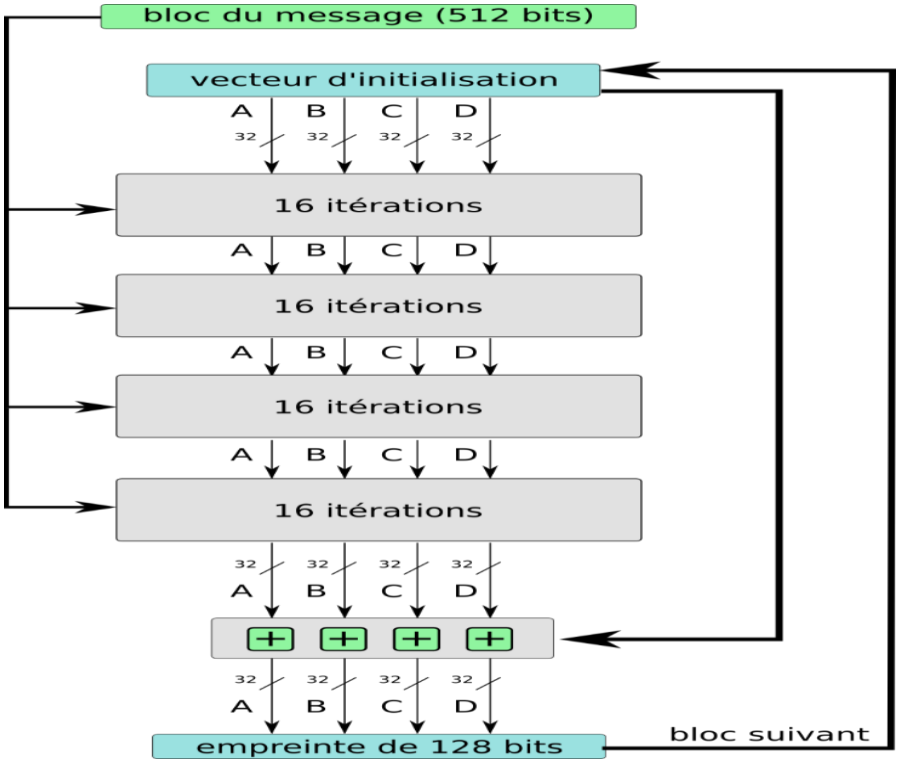


Figure 4.4 Structure générale du MD5

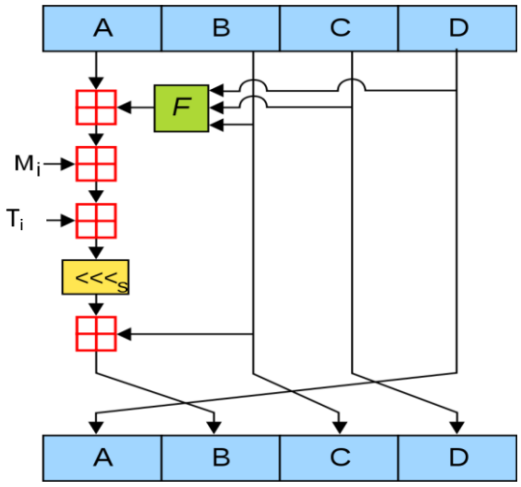


Figure 4.5 Un tour de MD5

- A chaque étape de chaque ronde une fonction non linéaire est effectuée sur 3 des variables B,C,D, le résultat est ajouté à un des 16 mots du bloc de message et à un élément du tableau T. Le résultat est décalé de *s* bits.

- Les 4 fonctions sont les suivantes :

  1. F= (B and C) or (not(B) and D)
  2. F = (D and B) or (C and not(D))
  3. F = B XOR C XOR D
  4. F = C XOR (B OR NOT(D))

- *Mj* symbolise un bloc de 32 bits provenant du message à hacher et *Ti* est une constante de 32 bits, différentes pour chaque itération.

b) Algorithme MD5

//Note : Toutes les variables sont sur 32 bits

//Définir r comme suit :

var entier [64] r, T

r[0..15] := {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}
r[16..31] := {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}
r[32..47] := {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}
r[48..63] := {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}

//MD5 utilise des sinus d'entiers pour ses constantes :

pour i **de** 0 **à** 63 **faire**

    T[i] := floor(abs(sin(i + 1)) × 2^32) // la partie entière

fin pour

//Préparation des variables :

var entier h0 := 0x01234567
var entier h1 := 0x89ABCDEF
var entier h2 := 0xFEDCBA98
var entier h3 := 0x76543210

//Préparation du message (padding) :

ajouter le bit "1" au message
ajouter le bit "0" jusqu'à ce que la taille du message en bits soit égale à 448 (mod 512)
ajouter la taille du message codée en 64-bit au message

//Découpage en blocs de 512 bits :

pour chaque bloc de 512 bits du message
subdiviser en 16 mots de 32 bits en M[j], $0 \leq j \leq 15$

    //initialiser les valeurs de hachage :

    var entier a := h0
    var entier b := h1
    var entier c := h2
    var entier d := h3

```
//Boucle principale :
pour i de 0 à 63 faire
    si 0 ≤ i ≤ 15 alors
        f := (b et c) ou ((non b) et d)
        g := i
    sinon si 16 ≤ i ≤ 31 alors
        f := (d et b) ou ((non d) et c)
        g := (5×i + 1) mod 16
    sinon si 32 ≤ i ≤ 47 alors
        f := b xor c xor d
        g := (3×i + 5) mod 16
    sinon si 48 ≤ i ≤ 63 alors
        f := c xor (b ou (non d))
        g := (7×i) mod 16
    fin si
    var entier temp := d
    d := c
    c := b
    b := ((a + f + k[i] + w[g]) leftrotate r[i]) + b
    a := temp
fin pour

//ajouter le résultat au bloc précédent :
h0 := h0 + a
h1 := h1 + b
h2 := h2 + c
h3 := h3 + d

fin pour
```

c) La cryptanalyse

MD5 est actuellement considéré comme une méthode de hachage non sûre. En réalisant une attaque par force brute, cela permet de retrouver un message original assez rapidement. La complexité du message haché est de 2^{64} calculs, or certains algorithmes permettent de réduire les calculs à 2^{30} opérations.

4.10 Protocole de signature à clé publique et fonction de hachage.

Les algorithmes à clé publique sont trop lents pour signer de longs documents. Pour gagner du temps les protocoles de signature numérique sont souvent réalisés avec des fonctions de hachage à sens unique.

Au lieu de signer le document Alice signe l'empreinte du document en suivant le protocole suivant:

1. Alice calcule à l'aide de la fonction de hachage à sens unique, l'empreinte du document.
2. Alice chiffre à l'aide de l'algorithme de signature numérique, cette empreinte avec sa clé privée, signant par la même occasion le document.
3. Alice envoie le document et l'empreinte signée à Bob (à l'aide de la clé publique de Bob).
4. Bob calcule, à l'aide de la fonction de hachage à sens unique, l'empreinte du document qu'Alice lui a envoyé. Ensuite à l'aide de l'algorithme de signature numérique, il déchiffre l'empreinte signée avec la clef publique d'Alice. La signature est valide si l'empreinte de la signature est la même que l'empreinte qu'il a produite.

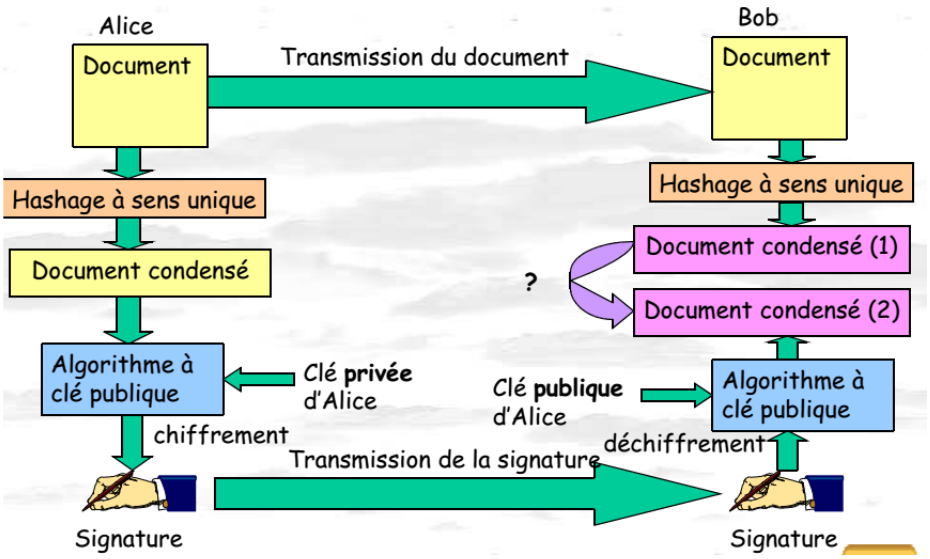


Figure 4.6 Signature à clé publique et hachage

Avantage de ce procédé:

- Rapidité de la transmission et de la comparaison des empreintes car une empreinte ne comporte que 160 bits ou au plus 512.
- Confidentialité car la signature peut être gardée à part du message. On peut donc vérifier l'existence du document sans stocker son contenu.

4.11 Infrastructure des systèmes à clef publique

Les infrastructures des systèmes à clef publique ou PKI (Public Key Infrastructure) consistent en toutes les dispositions techniques et organisationnelles nécessaires pour gérer un système cryptographique à clef publique.

On a un ensemble d'interlocuteurs en réseau qui se sont mis d'accord sur un cryptosystème à clef publique (par exemple RSA), sur une fonction de hachage et sur un protocole de signature. On suppose que chacun d'entre eux dispose d'une paire (clé publique, clé secrète),

(clé de chiffrement, clé de déchiffrement) et que chacun d'entre eux est capable de chiffrer, déchiffrer et signer.

Du fait que les clés publiques ne sont pas confidentielles, il n'est pas nécessaire de les crypter pour les transmettre. Mais il est très important et même vital pour la sécurité des transmissions de s'assurer de l'authenticité des clés ainsi transmises.

En effet si Alice désire transmettre sa clé publique à Bob, n'importe quel opposant, Martin par exemple, peut intercepter le message la contenant. Martin peut ensuite envoyer un message à Bob en se faisant passer pour Alice et contenant sa propre clé publique et donnant comme adresse de retour sa propre adresse. Ainsi il est capable de lire les messages cryptés que Bob envoie à Alice avec la soi-disant clé publique d'Alice que Bob croit posséder. Une fois qu'il les a décryptés et lus il peut les envoyer à Alice dont il possède la clé publique en les modifiant s'il l'estime nécessaire.

Il faut donc d'une certaine manière faire un lien entre chacun des participants au réseau et sa clé publique. Pour cela on utilise **les certificats**.

Un certificat consiste en une clef publique et une identité digitale (par exemple une suite de symboles contenant le nom du propriétaire de la clef, de la même manière qu'on met une étiquette sur une clé ordinaire), le tout étant cacheté à l'aide de la signature digitale d'une personne ou d'une organisation en laquelle on a confiance et appelée un Tiers de Confiance (Trusted Third Party ou TTP) ou encore **Certification Authority**.

Pratiquement on peut par exemple concaténer, mettre bout à bout, la clé publique, le nom de son possesseur et signer le message obtenu à l'aide de la clef privée du Tiers de Confiance (il faut que personne ne puisse usurper l'identité du Tiers de Confiance).

Il existe plusieurs modèles de réseau avec Tiers de Confiance, avec deux modèles extrêmes, le modèle hiérarchique qui repose sur des TTP distincts des utilisateurs et le modèle distribué où chaque utilisateur est son propre autorité de certification. Un système très employé de système hiérarchique de Tiers de Confiance est le modèle X.509

Il y a aussi des systèmes non hiérarchiques de PKI fondé sur des chaînes de certificats et une distribution de clefs décentralisées.

4.10.1 Certificat

Un certificat est un élément d'information qui prouve l'identité du propriétaire d'une clé publique. À l'instar d'un passeport, un certificat est une preuve reconnue de l'identité d'une personne. Les certificats sont signés et transmis de façon sécuritaire par un tiers de confiance appelé autorité de certification (AC).

Le rôle de l'AC est de s'assurer de la validité de la correspondance entre un nom d'une personne et une clé publique.

– L'AC émet des certificats X.509 aux personnes qu'elle a pu authentifier.

Une personne faisant confiance à une AC devrait pouvoir identifier toutes les personnes authentifiées par cette AC.

4.10.2 Structure du certificat X.509v3

Un certificat contient notamment :

Version du certificat
Numéro de série du certificat
Algo.de signature de l'AC
Nom de l'AC ayant délivré le certificat
Période de validité
Nom du propriétaire du certificat
Clé publique
Algo. à utiliser avec la clé publique
Identification de l'AC (opt)
Identification du propriétaire (opt)
Extensions (opt)
Signature de l'AC

Figure 4.7 Contenu d'un certificat

4.10.3 Certificat et vérification

Certificat de l'AC :

Distribution du certificat de l'AC à tous les intervenants.

– Certificat auto-signé c.-à-d. que l'AC signe son propre certificat.

– Le certificat est distribué de façon sécurisée (par exemple avec le système d'exploitation).

Certificat du client :

Chaque intervenant s'inscrit à l'AC afin qu'il puisse être identifié par un autre intervenant.

– L'intervenant reçoit un certificat l'identifiant signé par l'AC.

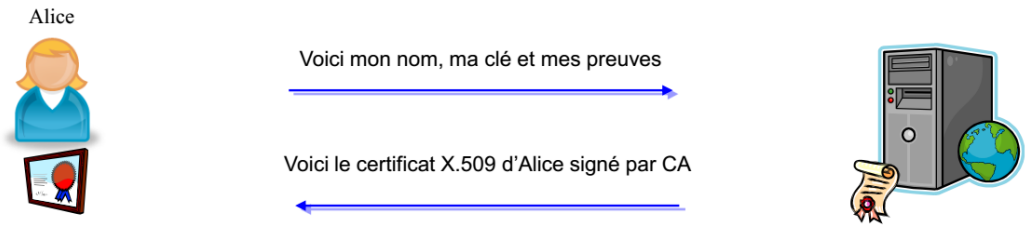


Figure 4.8 Génération du ertificat

Chaque intervenant ayant un certificat peut « prouver » son identité à tout autre intervenant ayant confiance à l'AC.

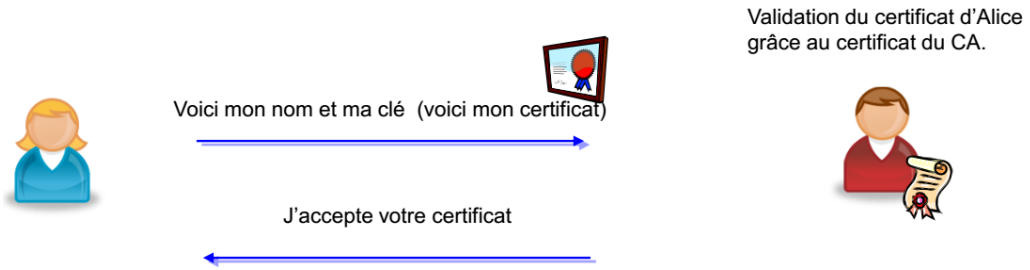


Figure 4.9 Validation du certificat par le récepteur

Bob peut posséder le certificat de plusieurs ACs afin de pouvoir identifier plusieurs intervenants.

Vérification :

En disposant d'un certificat au lieu d'une clé publique, le destinataire peut maintenant vérifier un certain nombre d'aspects au sujet de l'émetteur pour **s'assurer que le certificat est valide et qu'il appartient bien à la personne à qui il est censé appartenir.**

Il peut notamment :

- comparer l'identité du propriétaire;
- vérifier que le certificat est toujours valide;
- vérifier que le certificat a été signé par une AC de confiance;
- vérifier la signature du certificat de l'émetteur pour s'assurer que ce dernier n'a pas été altéré.

Notez que les certificats sont signés par une AC, ce qui signifie qu'ils ne peuvent être altérés. La signature de l'AC peut, à son tour, être vérifiée à l'aide du certificat de cette AC.

Afin de garantir l'intégrité du message chiffré, la signature du message se base principalement sur la signature du condensé de ce message qui obtenue en appliquant une fonction de hachage.

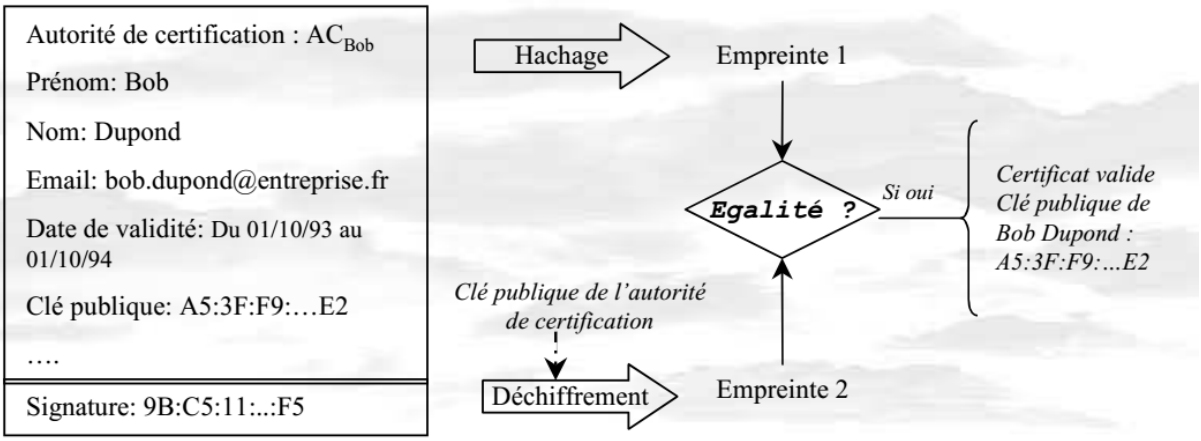


Figure 4.10 Vérification du certificat

4.11 Exercices

Exercice 4.1

1. Quel est le but de l'algorithme de DiffieHellman ?
2. Dans la cryptographie asymétrique chaque entité possède une paire de clés.
   a. Quelle clé utilise Alice pour chiffrer un message destiné à Bob?
   b. Quelle clé utilise Bob pour déchiffrer le message reçu ?

Exercice 4.2

Expliquer à travers un simple exemple comment obtenir une paire de clé (publique et privé) dans un système RSA.

Exercice 4.3

Bob choisit comme nombre premier p = 17 et q = 19, comme exposant e = 5. Alice et lui se fixent un protocole RSA dans lequel les messages sont des nombres en base 10 que l'on code par bloc de 2 chiffres. Alice veut envoyer le message "462739".

1. Donnez la clé privée de Bob.
2. Ecrivez le message chiffré qu'Alice envoie à Bob.
3. Déchiffrez le message qu'a reçu Bob et vérifiez que c'est bien celui qu'a envoyé Alice.

Exercice 4.4

Admettons qu'Alice choisisse e_{Alice}= 5, d_{Alice}=29, N_{Alice}=91, de son côté, Bob a choisi e_{Bob}=3, d_{Bob}=7 n=33.

Alice veut faire verser à Charles la somme de 111 euros. Le message en clair est donc m = 111.

1. Quelle est la signature S qu'Alice calcule avec sa clé privée ?

2. Quel est le message signé codé qu'elle envoie à Bob.
3. Comment Bob procède pour trouver le message en clair ?
4. Comment Bob vérifie-t-il l'authenticité et l'intégrité du message reçu ?

Exercice 4.5

Proposer un protocole qui permettrait à Alice d'envoyer un très gros fichier à Bob, en utilisant à la fois la cryptographie symétrique et asymétrique. Quel est l'objectif de ce protocole ?

Exercice 4.6

On considère l'application suivante du hachage cryptographique. On suppose que h est une fonction de hachage cryptographique que tout le monde (Alice, le serveur, l'espion, …) connaissent.

- Alice choisit un nombre g
- Elle calcule h(g), h(h(g)), …, h(h(h(h(h(h(h(h(h(g))))))))))=h^{11}(g)
- Elle transmet1 h^{11}(g) au serveur qui le stocke tel quel.

L'espion, de son côté, peut espionner tout ce qui passe sur le réseau. Il souhaite se connecter à distance au serveur en se faisant passer pour Alice.

1. Pour s'authentifier à distance, Alice transmet h^{10}(g) au serveur. Est-ce un authentification solide ?
2. Pour s'authentifier à distance une deuxième fois, Alice transmet h^{10}(g) au serveur. Est-ce un authentification solide ?
3. Proposez un algorithme d'authentification sûr qui s'appuie sur les résultats de la question 1 et qui résiste à un espion qui peut lire2 tout ce qui passe sur le réseau. La seule fonction cryptographique que votre algorithme est autorisé à utiliser est le fonction h.

Exercice 4.7

A quel problème répond une infrastructure de gestion de clefs (PKI) ? Décrivez sa structure.

Conclusion générale

Le présent polycopié a abordé des notions de base en cryptographie, il a mis l'accent sur les principales méthodes de la cryptographie classique. Le polycopié a aussi présenté l'essentiel sur la cryptographie moderne par la description des algorithmes les plus important pour le chiffrement symétrique à savoir DES et AES ainsi que pour le chiffrement asymétrique en abordant le système RSA et la signature numérique. Une étude sur les attaques et la cryptanalyse des algorithmes a été discuté. Enfin, la fonction de hachage et le certificat ont été présentés avec une description de leurs fonctionnements.

Corrigés des exercices

Exercice 2.1

1. Chiffrement de César : avec un décalage des lettres d'alphabet avec 3 positions, les lettres du message en clair correspondent au texte chiffré suivant : ERQMRXU
2. Chiffrement par décalage (généralisation de code de César) : En remplaçant les lettre du message par leurs rang dans l'alphabet et en appliquant pour chacune la formule de chiffrement :E = (x+k) mod 26, donc le message chiffré est JXKL.
3. Il n'y a que 26 décalages possibles pour le chiffrement par décalage. Un ordinateur ou un humain (suffisamment patient) peut très facilement tester ces 26 possibilités et est sûr de décrypter le cryptogramme.

Exercice 2.2

1. Dans la langue française c'est la lettre A et E qui sont les plus fréquentes et puisque dans le texte chiffre c'est les lettres K et O qui apparaissent beaucoup donc on peut essayer les combinaisons suivantes :
   - La lettre A est remplacé par la lettre K et la lettre E est remplacé par la lettre O
   - La lettre A est remplacé par la lettre O et la lettre E est remplacé par la lettre K

   Dans le chiffrement par décalage toutes les lettres sont décalés avec le même nombre de position donc en calculant le nombre de décalage dans les deux combinaisons, on constate que la première combinaison est correcte car le nombre de décalage entre les lettres A et K est le même que celui des lettre E et O donc la clé K=10.

2. Pour déchiffrer le texte il suffit d'appliquer la formule de déchiffrement : D (y) = (y - k) mod 26

Exercice 2.4

Pour remplacer le A, on dispose des 26 lettres de l'alphabet. Pour le B il n'y a plus que 25 possibilités, et ainsi de suite. Il y a donc $26 \times 25 \times 24 \times \cdots \times 2 \times 1$ possibilités de cryptage par permutation.

Exercice 2.5

1. Chiffrement Vigenère : Nous répétant autant de fois le mot clé selon la longueur du message puis nous faisons l'intersection entre chaque lettre du message en clair et la lettre qui lui correspond du mot clé en utilisant le carré de vigenère.

Message	C	H	I	F	F	R	E	D	E	V	I	G	E	N	E	R	E
Clé	B	A	C	H	E	L	I	E	R	B	A	C	H	E	L	I	E
Chiffré	D	H	K	M	J	C	M	H	V	W	I	I	L	R	P	Z	I

2. Vigénère apporte un plus car une même lettre peut être chiffrée différemment dans le texte ce qui complique l'analyse statistique.

Exercice 2.6

1. Chiffrement affine : $E_{(a,b)}(x) = ax+b \bmod 26$ *tel que (a,b)=(7,3).* Alors me texte chiffré est : RCF
2. Déchiffrement affine : $D(y) = a^{-1}(y-b) \bmod 26$. a^{-1}est l'inverse modulo *de a (mod 26) tel que* a. $a^{-1}=1 \bmod 26$. L'inverse modulo de 7^{-1} est 15. Alors le message en clair est : CODE

Exercice 2.7

Chiffrement de Hill :

$$\begin{pmatrix} C_k \\ C_{k+1} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} P_k \\ P_{k+1} \end{pmatrix} (\mathrm{mod} 26)$$

Les lettres en clair : P_1 et P_2 et les lettre chiffrés : C_1 et C_2

$$C_1 \equiv aP_1 + bP_2 (\mathrm{mod} 26)$$

$$C_2 \equiv cP_1 + dP_2 (\mathrm{mod} 26)$$

Chiffrement de DZ avec la clé : $\begin{bmatrix} 3 & 2 \\ 1 & 3 \end{bmatrix}$, tel que D→3 et Z→25

On a : 3x3+2x25 =59 mod26= 7

1x3+3x25= 78 mod26=0, donc DZ est chiffré par HA.

Exercice 2.8

1. Chiffrement par transposition simple : écrire le message ligne par ligne et le lire colonne par colonne dans une matrice 5x5

T	U	E	R	L
E	R	O	I	D
E	M	A	I	N
A	M	I	N	U
I	T	X	X	X

Compléter les cases vides par la lettre X. Le texte chiffré est :

TEEAIURMMTEOAIXRIINXLDNUX

2. Chiffrement par transposition complexe : Chaque lettre du mot clé correspond à une colonne puis numéroter les colonnes selon le rang de chaque lettre du mot clé. La lecture commence par la colonne qui correspond au plus petit rang (ordre croissant).

1	**5**	**3**	**4**	**2**
C	**R**	**I**	**M**	**E**
T	U	E	R	L
E	R	O	I	D
E	M	A	I	N
A	M	I	N	U
I	T	X	X	X

Le texte chiffré est : TEEAILDNUXEOAIXRIINXURMMT

3. Déchiffrement par transposition complexe : Il faut diviser le texte chiffré en bloc de 6 selon le nombre de ligne de la matrice. Ensuite il faut numéroter ces blocs de 1 à 5 puis remplir les colonnes selon l'ordre donné 3-2-5-1-4 et lire la matrice ligne par ligne.

C	H	I	F	F
R	E	M	E	N
T	P	A	R	T
R	A	N	S	P
O	S	I	T	I
O	N	X	X	X

Le message en clair est : CHIFFREMENT PAR TRANSPOSITION

Exercice 3.1

1. Avantage : le chiffrement symétrique est rapide.

   Inconvénient : La distribution des clés n'est pas sécurisée ainsi que la gestion des clés.

2. Le nombre de clés = N (N-1)/2 tel que N est le nombre des intervenants.

   N=5, alors on trouve **10 clés**.

Exercice 3.2

M= 10110001 donc m_1=1011, m_2=0001

a) Mode ECB

$c_i=E_k(m_i)$

donc : $c_1 = E_k(m_1) = E_k(1011) = 1100$

$c_2 = E_k(m_2) = E_k(0001) = 1001$

b) Mode CBC (c_0=1010)

$c_i= (m_i \oplus c_{i-1})$

donc : $c_1= E_k(m_1 \text{xor } c_0)$

$= E_k[(1011) \oplus (1010)]$

$= E_k (0001)$

$= 1001$

$C_2= E_k(m_2 \oplus c_1)$

$= Ek[(0001) \square (1001)]$

$= Ek(1000)$

$= 0101$

c) Mode CFB (c_0=1010)

$c_i= m_i \oplus E_K (c_{i-1})$

donc : $c_1 = m_1 \oplus E_k(c_0)$

$= (1011) \oplus E_k(1010)$

$= (1011) \oplus (0100)$

$= 1111$

$c_2 = m_2 \oplus E_k(c_1)$

$= (0001) \oplus E_k(1111)$

$= (0001) \oplus (1110)$

$= 1111$

d) Mode OFB (c_0=1010)

$Z_0=c_0$; $Z_i= E_k (Z_{i-1})$, $c_i=m_i \oplus Z_i$

donc : $Z_0=c_0=1010$

$Z_1= E_k (Z_0)= E_k (1010)=0100$

$c_1 = m_1 \oplus Z_1$

$= (1011) \oplus (0100)$

$= 1111$

$Z_2 = E_k (Z_1) = E_k (0100) = 0011$

$c_2 = m_2 \oplus Z_2$

$= (0001) \oplus (0011)$

$= 0010$

Exercice 3.3

On divise le message M en 2 blocs, on aura : $G_0 = 1101$ et $D0 = 0011$

- Première ronde : on cherche G_1 et D_1

  $G_1 = D_0 = 0011$

  $D_1 = G_0 \oplus f_1(D_0) = G_0 \oplus (D_0 \oplus 1011) = 1101 \oplus (0011 \oplus 1011) = 0101$

- Deuxième ronde : on cherche G_2 et D_2

  $G_2 = D_1 = 0101$

  $D_2 = G_1 \oplus f_2(D_1) = G_1 \oplus (\overline{D1} \oplus 0101) = 0011 \oplus (1010 \oplus 0101) = 1100$

On reconstitue les deux parties G_2 et D_2 ; le message chiffré C= 01011100

Exercice 3.4

1. Convertir la valeur du message et la clé en binaire :

M=101000011100000 (16 bits) et K=000001111110 (12 bits)

2. Construire La sous-clé K_1 :

On divise la clé principale Ken 2 blocs et on effectue un décalage cyclique sur les deux parties puis on applique la permutation PC sur tout le bloc reconstitué, donc on aura :

$K_1 = 110011010100$

3. Chiffrement :

- Permutation initiale : PI (M)= 1000110000001100
- Première ronde :

  $G_1 = D_0 = 00001100$

  $D_1 = G_0 \oplus f_1(D_0, K_1)$; on doit calculer la fonction de confusion $f_1(D_0, K_1)$:

- Expansion: $E(D_0)$= 000001011000
- XOR K_1: $E(D_0) \oplus K_1$=110010001100
- Substitution: On divise le résultat précèdent en bloc de 6 bits :

  B_1=110010 : Le premier et le dernier bit c'est 10 et ça donne 2 en décimal, 1001 c'est les bits au milieu et c'est 9 en décimal. L'intersection de 2 et 9 dans S-BOX donne 12 donc1100 en binaire.

  On fait la même chose avec B_2=001100 : on fait l »intersection de 0 et 6, on trouve 3 dans le S-BOX qui donne 0011 en binaire.

- Permutation : on recolle B_1 et B_2 pour appliquer la permutation $P(B_1B_2)$= 11010001 (le résultat de la fonction de confusion)

On avait : $D_1 = G_0 \oplus f_1(D_0, K_1)$ alors D_1=10001100⊕11010001= 010111101

Le résultat de la première ronde : **G_1= 00001100** et **D_1=010111101**

Exercice 3.5

1. Les transformations de ronde sont : SubBytes, ShiftRows, MixColumns, AddRoundKey
2. Le résultat de substitution est 27
3. Le résultat de rotation est le suivant :

04	e0	48	28
cb	f8	06	66
d3	26	81	19
4c	e5	9a	7a

4. 66 + fa = 9c

   02 * bf = 65

   11* de + 02 * bf = f4

Exercice 4.1

1. L'algorithme de DiffieHellman permet à deux personnes de se mettre d'accord sur un secret commun en échangeant des messages publiques qu'un espion peut voir.
2. Alice utilise la clé publique de Bob pour le chiffrement et Bob utilise sa clé privée pour le déchiffrement.

Exercice 4.2

Exemple simplifié (avec p et q très petits). On choisit : $p = 5$ et $q = 11$

Ce qui implique : $\boldsymbol{n} = p \times q = 5 \times 11 = \boldsymbol{55}$

$\Phi(n) = (p-1) \times (q-1) = 4 \times 10 = 40$

On peut choisir ***e = 7*** (7 est premier avec 40)

Et on calcule *d* tel que : $d.e = 1 \bmod \Phi(n) \Rightarrow d.7 = 1 \bmod 40 \Rightarrow$ ***d = 23***

On obtient donc : Clé publique : (7, 55) et Clé privée : (23, 55)

Exercice 4.3

1. La clé privée de Bob : e.d mod (p-1) (q-1)=1 alors $d=5^{-1} \bmod 288$ (calcul de l'inverse modulo de 5)

   d=173

2. Chiffrement : $C= M^e \bmod n$ tel que n=p.q et (e,n) est la clé publique de Bob

   On découpe le message en bloc de 2 chiffres puis on les chiffre avec la formule précédant :

   C= 088 278 286

3. Déchiffrement : $M= C^d \bmod$ tel que n=p.q et (d,n) est la clé privée de Bob

   M= 462739

Exercice 4.5

Le protocole proposé est comme suit :

1. Générer aléatoirement une clé de taille raisonnable utilisée pour un algorithme de chiffrement symétrique ;
2. Chiffrer cette clé à l'aide d'un algorithme de chiffrement à clé publique, à l'aide de la clé publique du destinataire ;
3. Envoyer cette clé chiffrée au destinataire et le destinataire déchiffre la clé symétrique à l'aide de sa clé privée.
4. Les deux interlocuteurs disposent ensuite : une clé symétrique commune qu'ils sont seuls à connaître ; et donc, de la possibilité de communiquer en chiffrant leur données à l'aide d'un algorithme de chiffrement symétrique rapide.

Exercice 4.6

1. C'est une authentification solide car on ne peut déduire $h_{10}(g)$ à partir de $h_{11}(g)$. Alice est donc seule à pouvoir fournir $h_{10}(g)$.
2. Ce n'est pas une authentification solide car l'espion a déjà vu passer $h_{10}(g)$.

3. Le problème mis en évidence à la question précédente est lié à la réutilisation de $h_{10}(g)$. Si on ne l'avait utilisé qu'une fois, on aurait eu une authentification solide. Pb: comment s'authentifier la deuxième fois ? Réponse en fournissant $h_9(g)$. C'est le mécanisme des mots de passe jetable « OTP (One Time Password) :

   On choisit g

   On fournit $h_{11}(g)$ au serveur

   Première authentification :

   - Alice fournit $h_{10}(g) = m$ au serveur
   - le serveur valide l'authentification en vérifiant que $h(m)=h_{11}(g)$
   - le serveur mémorise $h_{10}(g)$

   Seconde authentification:

   - Alice fournit $h_9(g) = m$ au serveur
   - le serveur valide l'authentification en vérifiant que $h(m)=h_{10}(g)$
   - le serveur mémorise $h_9(g)$

   Et ainsi de suite.

   Évidemment, une fois qu'on a fourni g, il faut choisir un nouveau nombre g et transmettre $h_{11}(g)$ au serveur de façon sûre. On peut bien sur choisir des valeurs plus grandes que 11 pour éviter d'avoir à refaire ça trop souvent.

Exercice 4.7

Les algorithmes à clefs publiques sont sûrs. Le problème est d'avoir la preuve qu'une clé publique est bien la clé publique de la personne avec laquelle on veut communiquer. Une PKI résout ce problème en mettant en place une autorité de confiance qui certifiera les clefs publiques en précisant pour chaque clé de qui elle est la clé et si elle est valide.

Une PKI comprend trois éléments obligatoires :

- Une autorité d'enregistrement chargée de vérifier l'identité du possesseur de la clé publique en appliquant la politique définie par l'autorité de certification. L'utilisateur qui veut obtenir un certificat s'adresse à elle
- Une autorité de certification qui est une autorité de confiance reconnue par une communauté d'utilisateurs. Elle délivre et gère les certificats (« clefs publiques + identités » signées), maintient une liste des certificats révoqués.

- Un service de publication qui met à la disposition de la communauté les certificats et liste aussi ceux qui sont révoqués.