TABLE DE MATIERES

TABLE DE MATIERES ..... **I**
NOTATIONS ..... **V**
INTRODUCTION GENERALE **1**
CHAPITRE 1 : GENERALITES SUR LA CRYPTOGRAPHIE **3**
1. Terminologies ***3***
2. Historiques ***3***
3. La cryptographie du point de vue général ***5***
4. Les techniques de cryptographie ***5***
4.1 Le chiffrement par substitution : **6**
4.2 Le chiffrement par transposition : **6**
4.3 Le chiffrement par bloc : **6**
4.3.1 Le mode ECB ***6***
4.3.2 Le mode CBC ***7***
4.3.3 Le chiffrement par bloc avec itération ***7***
4.4 Le chiffrement par flot : **8**
4.4.1 Le chiffrement de Vernam ***8***
4.4.2 Chiffrement avec un registre à décalage à rétroaction linéaire ***8***
4.4.3 Chiffrement à registre à LFSR combinées : ***9***
5. Implémentation de la cryptographie dans le modèle OSI ***11***
5.1 Le système OSI **11**
5.1.1 La couche physique : ***11***
5.1.2 La couche liaison de donnée : ***11***
5.1.3 La couche réseau : ***12***
5.1.4 La couche transport : ***12***
5.1.5 Les couches hautes : session, présentation et application. ***12***
5.1.6 La couche session. ***12***
5.1.7 La couche présentation. ***12***
5.1.8 La couche application. ***12***
5.2 La cryptographie dans le modèle OSI. **13**
5.2.1 Le chiffrement lien par lien. ***13***
5.2.2 Le chiffrement de bout en bout. ***14***
6. La cryptanalyse ***14***
6.1 Attaques des fonctions de chiffrement. **14**
6.2 Classement des attaques en fonction des données dont dispose le cryptanalyste **14**
6.3 Attaques sur les algorithmes symétriques. **15**
6.3.1 Attaques au niveau des clés. ***15***
6.3.2 La cryptanalyse différentiel ***15***
6.3.3 La cryptanalyse linéaire. ***16***
6.4 Attaques sur les algorithmes asymétriques. **16**

6.4.1 Attaques au niveau des clés. *16*
6.4.2 L'attaque à texte clair deviné et problème de la faible entropie *16*
6.4.3 Attaque à texte chiffré choisi. *16*
6.4.4 Attaque temporelle. *17*
6.5 Attaques des générateurs pseudo-aléatoires. **17**
CHAPITRE 2 THEORIES DE NOMBRES ET LA CRYPTOGRAPHIE. **18**
1. Elément de théorie des nombres ***18***
1.1 Entropie. **18**
1.2 Quantité d'information. **18**
1.3 Système cryptographique et théorie de l'information. **19**
2. Elément mathématique pour la cryptographie ***20***
2.1 Les nombres premiers. **21**
2.2 Congruence dans □ **21**
2.3 Ensemble quotient $\mathbf{Z}\big|n\mathbf{Z} = \mathbf{Z}_n$ **21**
2.3.1 Divisibilités dans $\mathbf{Z}$ *22*
2.3.2 Plus petit commun multiple (ppcm) et plus grand commun diviseur (pgcd). *22*
2.4 Algorithme d'Euclide. **23**
2.5 Algorithme d'Euclide étendu. **23**
2.6 Exponentiation modulo n. **24**
2.7 Théorème de Fermat et d'Euler. **25**
2.8 Système de congruence. Théorème chinois. **25**
2.9 Décomposition d'un entier en facteurs premiers. **26**
2.10 Résidu quadratique **26**
2.11 Symbole de Legendre. **27**
2.12 Logarithme discret. **27**
3. La cryptographie classique ***28***
4. La cryptographie moderne ***28***
CHAPITRE 3 : LES ALGORITHMES DE CRYPTAGES **29**
1. Définitions ***29***
1.1 Chiffrement à clé privé **29**
1.2 Chiffrement à clé publique **29**
2. Quelque Cryptosystème à clé privée ***30***
2.1 Le DES et son successeur. **30**
2.1.1 Historique. *30*
2.1.2 Schéma général. *30*
2.1.3 La fonction f *32*
2.1.4 Les boites-S *33*
2.1.5 La diversification de la clé *33*
2.2 Le triple DES **34**
2.3 L'IDEA **35**

2.3.1 Algorithme. *35*
2.3.2 Déchiffrement : *36*
2.4 L'AES ou Rijndael **37**
2.4.1 Déchiffrement : *39*
3. Chiffrement à clé publique *39*
3.1 RSA **39**
3.1.1 Fonctionnement. *39*
3.1.2 Algorithme : *40*
3.3 Cryptosystème d'El Gamal. **41**
3.4 PGP **42**
3.4.1 Chiffrement. *42*
3.4.2 Déchiffrement. *42*
CHAPITRE 4 : LA CRYPTOGRAPHIE ET LA SECURITE DES RESEAUX DE TELECOMMUNICATIONS. **43**
1. Définition des vulnérabilités et des attaques de sécurité de la téléphonie *43*
2. Réseaux traditionnels analogique et numérique *44*
2.1 Architecture du réseau. **44**
2.2 Les Vulnérabilités de la téléphonie classique. **44**
2.3 La sécurité dans les réseaux téléphoniques traditionnels. **45**
2.3.1 Réseau Téléphonique commuté (RTC). *45*
2.3.2 Réseau numérique (RNIS). *46*
2.3.3 Sécurisation à travers le réseau de signalisation *46*
3. Réseau mobile GSM/GPRS *47*
3.1 Introduction. **47**
3.2 Architecture du réseau. **47**
3.3 Vulnérabilités et failles du réseau GSM/GPRS **49**
3.4 La sécurité dans le réseau GSM/GPRS **50**
4. Réseaux UMTS *52*
4.1 Introduction **52**
4.2 Architecture du réseau. **52**
4.3 Vulnérabilités dans le réseau UMTS **53**
4.4 La sécurité dans le réseau UMTS. **54**
CHAPITRE 5 : CONCEPTION D'UN CRYPTOSYSTEME CLIENT/SERVEUR BASEE DE RSA **55**
1. Notion d'Architecture client/serveur *55*
1.1 Client/Serveur à client passif (1-tiers) **55**
1.2 Client/Serveur de donnée (2-tiers) **55**
1.3 Client/Serveur distribuées (3-tiers). **56**
2. Les choix dans la réalisation *56*
2.1 Choix de l'architecture Client/Serveur. **56**
2.2 Choix de l'algorithme de chiffrement. **56**
2.3 Choix du langage de programmation **57**

3. Conception et réalisation *57*
3.1 La classe InetAdress **58**
3.2 Utilisation du protocole TCP **58**
3.2.1 La classe SocketServer *59*
3.2.2 La classe Socket *59*
3.3 Réalisation du cryptosystème ou client. **60**
3.3.1 Description de JCE : *62*
3.3.2 La classe Provider *62*
3.3.3 La clé d'encryptage *62*
3.3.4 La classe Cipher *62*
3.4 Realisation du serveur. **64**
4. Présentation du cryptosystème Client/Serveur à base de RSA. *65*
4.1 La partie Client. **65**
4.1.1 Le mode cryptosystème simple. *65*
4.1.2 Le mode client du serveur *67*
4.2 La partie serveur. **70**
CONCLUSION GENERALE **73**
ANNEXES **75**
BIBLIOGRAPHIE **84**

NOTATIONS

a,b,c,d,e,r,p,n,u,v,q	: Des entiers
C	: Message chiffré
{c_1, c_2, …, c_i}	: Suite binaire de message chiffré
f	: Une fonction
E	: Une expérience aléatoire
H	: Entropie
K	: Clé de chiffrement
{k1, k2,...,ki}	: Suite binaire de clé
M	: Message clair
{m1, m2,…,mi}	: Suite binaire de message clair
pi	: Probabilité i
AES	: Advance Encryption Standard
AKA	: Authentification and Key Agreement
ATM	: Asynchronous Transfer Mode
AUC	: Authentification Center
BSS	: Base Station Subsystem
BTS	: Base Transceiver Station
BSC	: Base Station Controller
CBC	: Cipher Block Chaining
CRC	: Cyclic Redundancy Check
DES	: Data Encryption Standard
DSA	: Digital Signature Algorithm
DSS	: Digital Signature Standard
ECB	: Electronic CodeBook
EIR	: Equipment Identity Register
ETSI	: European Telecommunication Standardization Institute

GEA : GPRS Encryption Algorithm
GGSN : Gateway GPRS Support Node
GSM : Global System for Mobile Communications
GPRS : General Packet Radio Service

HLR : Home Location Register

IDE : Interface Development Environment
IMEI : International Mobile Equipment Identity
IMSI : International Mobile Subscriber Identity
ISO : International Standard Organisation
IP : Internet Protocol
JDK : Java Development Kit
JCE : Java Cryptography Extension

LFSR : Linear Feedback Shift Register

MAC : Message Authentification Code
MD5 : Message Digest 5
MS : Mobile Station
MSC : Mobile services Switching Center

NAT : Network Address Translation
NBS : National Bureau of Standard
NIST : National Institute of Standard and Technology
NSA : National Security Agency

OMC : Operation and Management Center
OSI : Open System Interconnection

PC : Personal Computer
PIM : Protocol Independent Multicast
PGP : Pretty Good Privacy
PLNM : Public Land Mobile Network

PSTN	: Public Switched Telephone Network
RSA	: Rivest Shamir Adelman
RTC	: Réseau Téléphonique Commuté
RNIS	: Réseau Numérique à Intégration de Service
RAND	: Nombre aléatoire
RNC	: Radio Network Controller
SGSN	: Serving GPRS Support Node
SHS	: Secure Hash Standard
SHA	: Secure Hash Algorithm
SS7	: Signalling System 7
STP	: Signalling Transfer Point
TCP	: Transport Control Protocol
TMSI	: Temporary Mobile Subscriber Identity
UMTS	: Universal Mobile Telecommunications System
USIM	: UMTS SIM
VLR	: Visitor Location Register
VPN	: Virtual Private Network
W-CDMA	: Wideband – Code – Division Multiple Access
addRoundKey	: Fonction d'addition de la clé de tour
MixCuloms	: Fonction de brouillages des colonnes
ShiftRows	: Fonction de décalage ligne
SubBytes	: Fonction de transformation non linéaire
$\mathbb{N}$	: Ensemble des entiers naturelles
$\mathbb{Z}$	: Ensemble des entiers relatifs
$\mathbb{Z}/n\mathbb{Z}$	: Ensemble des classes d'équivalence modulo *n*
$\oplus$	: Ou exclusif

INTRODUCTION GENERALE

La communication a toujours constitué l'un des piliers importants dans la vie des hommes. Le développement, les recherches, l'acquisition des nouvelles connaissances font parties de l'implication immédiate de la communication. Le besoin d'envoyer des messages sécurisés est aussi ancien que la communication elle-même. Dans notre monde moderne, qui utilise différents types de moyens de communication, le besoin de confidentialité est plus important que jamais. Ces besoins ont donnés naissance à la science des secrets que nous appelons cryptologie.

Comptenu de ceci, la cryptographie est devenue une composante essentielle de la sécurisation des systèmes de communication. Réservée au domaine militaire pendant plusieurs années, la cryptographie est utilisée actuellement dans différents secteurs d'activités. Elle fait partie intégrante de la sécurité des réseaux privés ou publics.

L'existence de différents types de réseaux, comme le réseau informatique, le réseau téléphonique qu'on considère comme réseau de télécommunication, dans le monde entraîne le besoin d'interconnexion. Cela va diminuer la résistance de ces réseaux faces à des attaques ou à des menaces. D'où la nécessité d'implémenter des solutions offrant plus de sécurité. Nous faisons allusion ici à la cryptographie, plus précisément à des solutions cryptographiques.

Les informations, les services qu'offrent un opérateur, ou une entreprise sont des ressources très sensibles, et qui méritent d'êtres sécurisées. C'est pourquoi l'implémentation des méthodes de chiffrement dans le réseau est inévitable. Même dans les réseaux informatiques, on peut envisager des solutions logicielles, des cryptosystèmes capables de chiffrer et déchiffrer les données, avant toute circulation dans le réseau. En effet, le développement de ces systèmes est devenu un très grand investissement rentable. Des grandes sociétés sont créées depuis, comme « RSA Security », « P.G.P ».

Aussi avons-nous l'idée de concevoir un de ces logiciels. D'où le titre de ce mémoire : « CONCEPTION D'UN CRYPTOSYSTEME CLIENT/SERVEUR BASEE SUR RSA », qui offre la sécurisation des informations, ainsi que de leurs transferts jusqu'à l'arrivée de l'information à la destination finale, en utilisant un chiffrement à clé publique RSA.

RSA est un algorithme de cryptage à clé publique largement utilisé pour assurer la sécurité de différents domaines. C'est aussi un algorithme difficile à casser. Même son prédécesseur (PGP), lui-même l'utilise pour le chiffrement de sa clé privée. Il est à savoir que PGP est un algorithme de chiffrement hybride.

Pour démontrer nos travaux de recherche, notre travail est divisé en cinq chapitres :

1. Généralités sur la cryptographie : qui est une chapitre introductive permettant au lecteur de se familiariser avec la cryptographie. Ceci va de l'historique jusqu'à son utilisation actuelle et des menaces possibles.
2. La cryptographie et la théorie des nombres : Ce chapitre est plutôt mathématique, où nous avons développés les notions de bases dont on a besoin pour pouvoir expliquer les algorithmes de chiffrement.
3. Les principaux algorithmes de chiffrement : Ce chapitre montre les différentes possibilités d'algorithmes cryptographiques.
4. La cryptographie et la sécurité des réseaux de télécommunication : Ce chapitre nous parle de l'utilisation de la cryptographie dans des réseaux de télécommunications.
5. Conception d'un Cryptosystème client/serveur basée sur RSA : Ce chapitre est la concrétisation de nos recherches.

CHAPITRE 1 : GENERALITES SUR LA CRYPTOGRAPHIE

1. Terminologies [1] [4]

La cryptologie ou encore science du secret présente deux visages complémentaires : la cryptographie et la cryptanalyse. La cryptographie, étymologiquement écriture secrète, est devenue par extension l'étude de cet art ; il s'agit de déterminer les manières les moins faillibles possibles de chiffrer des messages susceptibles d'être interceptés lors de leur transmission. La cryptanalyse en est son contrepoint : alors que pour le destinataire légitime du cryptogramme, il s'agit de le déchiffrer pour prendre connaissance du contenu du message, l'attaquant qu'est le cryptanalyse cherche à décrypter un message chiffré, c'est-à-dire à connaître le contenu du message sans posséder les codes ou les clés nécessaires à son déchiffrement. Ces deux aspects sont indissociables. En effet, la conception de systèmes cryptographiques sûrs ne saurait se faire sans connaissance sur les cryptanalyses antérieurs d'autres systèmes.

2. Historiques [2] [3]

L'encryptage, et par conséquent le décryptage, est un concept très ancien, puisqu'il a été utilisé par les plus nautiques générations.

- En 1900 ans avant Jésus Christ, les scribes Egyptiens utilisent les hiéroglyphes n'étant pas standard.
- En 500 ans avant Jésus Christ, des scribes hébreux emploient Atbash, un simple algorithme de chiffrement de substitution utilisant l'alphabet renversé, pour la transcription du livre de Jeremy.
- En 487 ans avant Jésus Christ, des Grecs utilisent scytales, un grand bâton de chiffre sur lequel on enroulait une longue et mince bande de cuivre auquel on enroulait une longue et mince bande cuire sur laquelle se trouvait des informations.
- En 50 ans avant Jésus Christ, Julius César utilise une simple substitution dans l'alphabet pour des communications gouvernementales.
- Entre 0 et 400 ans, le Kamasoutra de Vatsayana liste la cryptographie comme étant le $44^{ième}$ et le $45^{ième}$ art dans une liste totalisée de 64 arts.
- En 200 ans, le Papyrus de Leyde utilise un algorithme de chiffrement pour cacher les parties importantes de certaines recettes (magique).

- En 1918, Gilbert Vernam, mathématicien américain, invente le « on time cipher », qui est encore aujourd'hui l'algorithme de chiffrement le plus sécurisé mais impraticable.
- En 1919, l'américain Black Chambers, un organisme installé à New York, a pour rôle de déchiffrer les codes venant du Japon.
- En 1923, Dr Arthur Scherbius, Hollandais résidant en Allemagne, met au point « ENIGMA » qui sert à encoder les messages. Mais le prix très élevé de la machine a fait un échec.
- En 1925, la marine de guerre Allemande reprend le projet d'ENIGMA en le confiant aux chiffriestelles, service de chiffrement.
- En 1937, ENIGMA M3 est adapté par la Wehrmacht, l'armée Allemande.
- En 1939, début de la seconde guerre mondiale, 12 000 scientifiques et mathématiciens anglais, polonais et français travaillent à solutionner ENIGMA, ainsi que des milliers d'autres messages chiffrés. Et finalement l'équipe d'Alan Turing réussi à le résoudre.
- En 1970, IBM développe « LUCIFER ».
- En 1976, IBM publie une algorithme basé sur LUCIFER et devient le DES (Data Encryptions Standards).
- En 1976, Whitfield a et Martin Hellman introduit l'idée d'un système à clé publique.
- En 1978, l'algorithme de chiffrement à clé publique « RSA » est publié par Ronald L. Revest, A. Shamir et Léonard M. Adleman.
- En 1984, l'algorithme ROT 13 est utilisé dans le Usenet et devient le premier exemple d'utilisation de clé publique.
- En 1987, le RC4 est développé par Ronald L. Revest pour la RSA security et sera gardé secret jusqu'à 1994, où l'algorithme est rendu publique anonymement dans une liste de distribution de « cipher punk »
- En 1990, première publication des résultats expérimentaux de la cryptographie quantique par Charles H. Benêt et Gilles Brassard.
- En 1991, Phil Zimmermann rend disponible sa première version de PGP
- En 1992, l' IDEA est inventé en Suisse par Huejeya Laj et James Massey.
- En 1993, Bruce Scheiner conçoit Blowfish, et Don Choppers Schmith crée Seal.
- En 1994, Ronald L Revest, déjà auteur de RC1, RC4 publie RC5 et peu après, un standard régissant les signatures numériques voit le jour : « DSA ».
- En 1995, le NIST développe le « SHA »

- En 1998, l'algorithme Rijndael est finalisé et soumis au Nist pour devenir le nouveau standard de l'avancé : l'AES. Seize autres algorithmes font parties du groupes dont Mars, RC6, Serpent, et Two Fish
- En 2000, Rijndael devient l'AES standard de chiffrement avancée.
- Depuis 2000, des nouvelles versions de PGP apparaissent. Actuellement nous en sommes à la version 8.

3. La cryptographie du point de vue général [2] [3] [4]

La cryptographie, connue pour ses applications militaires et diplomatiques. Elle couvre en fait des domaines plus étendus que simple confidentialité des données assurée par le chiffrement qu'elle évoque en premier lieu. Dans un contexte plus général de protection de l'information contre des tentatives d'accès non autorisé à des données, on peut classer les principales fonctionnalités offertes par la cryptographie comme suit :

- le chiffrement, qui est utilisé pour protéger le contenu d'un message contre sa divulgation à toute personne ne possédant pas le secret lui permettant de le lire en clair ;
- l'identification, qui permet de s'assurer de l'identité de la personne dont est issu le message reçu ;
- l'authentification, qui assure que les données reçues n'ont subi aucune modification depuis leur émission première ;
- la signature, qui comme une signature manuscrite protège la provenance, l'intégrité et garantit la non répudiation d'un message.

Des applications cryptographiques permettent de combiner plusieurs de ces fonctionnalités ci-dessus .Nous nous intéressons dans notre étude qu'au chiffrement.

4. Les techniques de cryptographie [1] [2] [3] [6]

Il existe quatre techniques fondamentales qu'on utilise en cryptographie

- le chiffrement par substitution
- le chiffrement par transposition
- le chiffrement par bloc
- le chiffrement par flot

Les deux premières méthodes avaient utilisés par les anciens systèmes cryptographiques, mais que l'on utilise toujours de nos jours

Les deux dernières sont utilisées par les algorithmes de cryptographie moderne.

4.1 Le chiffrement par substitution :

Cette méthode consiste à remplacer les lettres ou les mots par un autre symbole. Cela présuppose de choisir un ensemble de symbole qui joueront le rôle du substitut. Les chiffres de substitutions peuvent être classés en quatre groupes ; chacun ayant des sous groupes, des variations, des variations avec d'autres types de chiffrement :

- substitution mono-alphabétique ;
- substitution poly-alphabétique ;
- substitution poly-grammique ;
- substitution Homophonique.

4.2 Le chiffrement par transposition :

Les méthodes de chiffrement par transposition consistent à réarranger les données à crypter de telle façon à les rendre incompréhensibles, donc à construire une anagramme. Il s'agit généralement de réarranger géométriquement les données pour les rendre visuellement inexploitables

4.3 Le chiffrement par bloc :

Ce procédé consiste à diviser en bloc, dont la taille est fixée et dépendante de la longueur de la clé, le message. Puis l'algorithme est appliqué successivement à chacun de ces blocs.

Les algorithmes de chiffrement par blocs peuvent être utilisées suivant différentes modes, dont les deux principaux sont le mode ECB (initiale de « Electronic CodeBook ») ou Carnet de Codage Electronique et le mode CBC (initiale de « Cipher Block Chaining ») ou mode de chiffrement par chaînage.

4.3.1 Le mode ECB

Quand on parle de chiffrement par bloc la méthode la plus évidente est le chiffrement en mode ECB. Il consiste à chiffrer les blocs construits indépendamment des uns et des autres. Ce mode a pour avantage de sa rapidité de chiffrement car il permet le chiffrement en parallèle des différents blocs. Son inconvénient est qu'un bloc est toujours chiffré en un même bloc, ce qui facilite la tâche d'un cryptanalyste.

4.3.2 Le mode CBC

La solution aux problèmes posée par le mode ECB est d'utiliser une technique dite de chaînage, dans laquelle chaque bloc du cryptogramme dépend non seulement du bloc correspondant, mais aussi des blocs précédents

Dans ce mode chaque bloc est combiné par un *ou exclusif* avec le bloc chiffré précédent avant d'être chiffré. Le premier bloc, quand à lui, combiné avec un bloc appelé *vecteur d'initialisation*. L'utilisation d'un vecteur d'initialisation différent pour chaque message permet de s'assurer que deux messages identiques (ou dont les premiers blocs sont identiques) donneront des cryptogrammes différents.

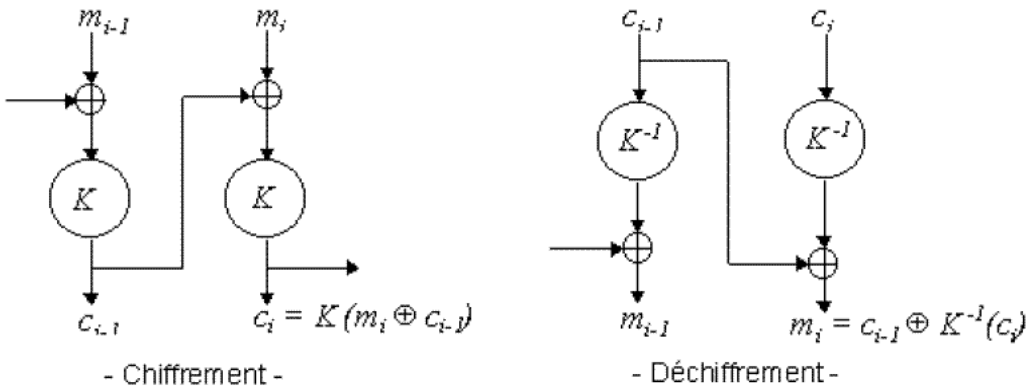

Figure 1.01 : le mode CBC

L'avantage de ce mode réside dans le masquage par chaînage de la structure des blocs. Une autre avantage est de ses capacités d'auto réparateur, c'est-à-dire, si un bit a été modifié en cours de route, seul le bloc correspondant et un bit du suivant seront endommagés. Son inconvénient majeur est du risque de propagation d'erreur importante. En effet, une erreur d'un bit dans un bloc affecte tous les blocs chiffrés suivants.

Il existe encore d'autre types de chiffrements par bloc mais ces deux sont les plus couramment utilisées

4.3.3 Le chiffrement par bloc avec itération

Un algorithme de chiffrement par bloc avec itération est un algorithme qui chiffre les blocs par un processus comportant plusieurs rondes. Dans chaque ronde, la même transformation est appliquée au bloc, en utilisant un sous clé dérivée de la clé de chiffrement. En général, un nombre de rondes plus élevées garantit une meilleure sécurité, au détriment des performances.

4.4 Le chiffrement par flot :

Ce procédé consiste à chiffrer le message de bit après bit à temps réels. Cette méthode est adaptée au chiffrage de l'information de type synchrone comme par exemple la parole avec le G.S.M. Le principe est d'effectuer un chiffrement de Vernam en utilisant une clé pseudo aléatoire.

4.4.1 Le chiffrement de Vernam

Le chiffrement de Vernam est un système de chiffrement daté de 1917, appelé aussi masque jetable ou « on time paid », et qui a servi de chiffrer les messages télégraphiques.

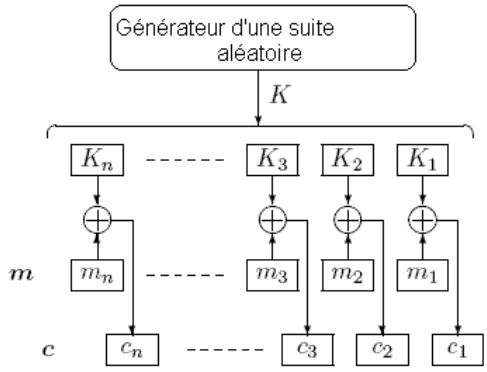

Figure 1.02 : Chiffrement de Vernam

Les messages claires et chiffrés sont des suites de bits de longueur *n*. La clé K est une suite binaire de même longueur que le message claire. Le chiffrement consiste à ajouter bit après bit par un *ou exclusif* les données avec la clé.

4.4.2 Chiffrement avec un registre à décalage à rétroaction linéaire

En pratique réaliser le chiffrement de Vernam est quasiment impossible, un générateur d'une suite aléatoire est très difficile à concevoir, alors on emploi les générateurs pseudo aléatoire. Une manière de produire une suite pseudo aléatoire est le registre à décalage à rétroaction linéaire.

Le registre à décalage à rétroaction linéaire (LFSR pour Linear Feedback Shift Register) est un composant élémentaire bien adapté pour des implémentations matérielles. En outre les suites binaires produites par de tels composants possèdent (sous certaines conditions) une période élevée et de bonnes propriétés statistiques.

Un registre à décalage à rétroaction linéaire de longueur L est constitué de L bascules reliées par une fonction de rétroaction linéaire. La figure 1.03 illustre le principe d'un tel composant.

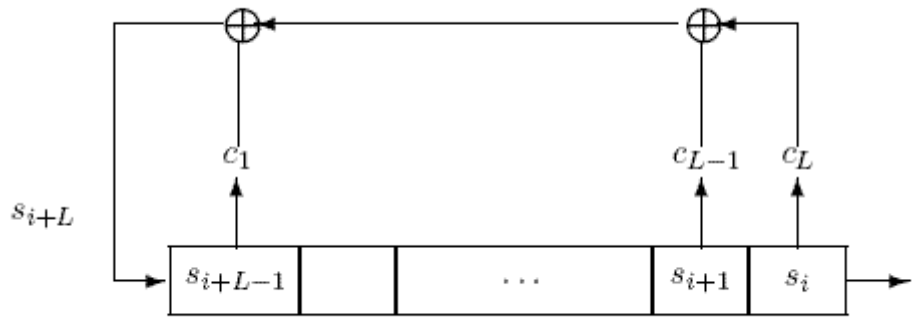

Figure 1.03 : Principe d'un registre à décalage à rétroaction linéaire

A chaque cycle d'horloge, les bits du registre sont décalés vers la sortie produisant ainsi le bit le plus ancien du registre. La bascule libérée reçoit alors un nouveau bit calculé grâce à la relation de rétroaction :

$$S_{i+L} = c_1 S_{i+L-1} \oplus c_2 S_{i+L-2} \oplus ... \oplus c_L S_i . \qquad (1.01)$$

La suite (S) produite par une telle récurrence linéaire est ultimement périodique, c'est-à-dire qu'il existe une pré-période T_0 telle que la suite $(S_n)_{n \geq T_0}$ est périodique (on a T_0=0 si c_L=1).

A toute suite récurrente linéaire on associe un polynôme f_s que l'on appelle polynôme de rétroaction :

$$f_s \in \mathrm{F}_2[X], \quad f_s(X) = 1 + c_1 X + c_2 X^2 + ... + c_L X^L . \qquad (1.02)$$

La même suite peut être générée à partir de différentes polynômes de rétroaction, c'est pourquoi on définie le polynôme de rétroaction minimal de la suite comme le polynôme de plus bas degré permettant d'engendrer cette suite. Le degré du polynôme de rétroaction minimal de la suite s correspond à la complexité linéaire de (S). On le noté $\Lambda(s)$. Lorsque le polynôme de rétroaction minimal est primitif et que l'état initial (S_0, S_1,…, S_{L-1}) est non nul, la période de la suite (S) est maximale et égale à $2^{\Lambda(s)}$-1. Les suites périodiques engendrées par des polynômes de rétroaction primitifs sont appelées m-séquences ou suites ML (de longueur maximale).

4.4.3 Chiffrement à registre à LFSR combinées :

Le chiffrement par un seul LFSR présente le désavantage de facile à décrypter par les cryptanalyste. La connaissance du polynôme de rétroaction (qui est en général fixé pour une

implémentation matérielle) permet de produire, à partir de l'observation de L bits consécutifs de la suite S, tous les bits ultérieurs. Le cas où le polynôme de rétroaction est inconnu (déterminé à partir de la clé par exemple) ne garantit pas une meilleure sécurité puisque la linéarité de la récurrence permet de retrouver les L coefficients du polynôme de rétroaction grâce à la résolution d'un système linéaire obtenu par l'observation de 2L éléments de la suite.

En effet, pour produire une suite chiffrante de complexité linéaire élevée et de grande période en utilisant les LFSR de taille raisonnable pour leur simplicité d'implémentation, il est impératif d'introduire une composante non-linéaire au système. Cette composante peut être obtenu par combinaisons de n LFSR par une fonction booléenne à n variables ou un seul LFSR dont on choisit n bascules avec des espacements soigneusement déterminés, que l'on filtre avec une fonction booléenne à n variables.

Pour que la suite obtenue en sortie de la fonction booléenne f ait une complexité linéaire significativement plus grand que celle d'une des composant d'entrée de f , il est nécessaire que le degré de f soit élevé. Les deux systèmes sont représentés par les schémas

Si on note z_t le bit de suite chiffrante à l'instant t, on a alors :

$$z_t = f\left(x_t^1, x_t^2, ..., x_t^n\right)$$

Où $x_t^1, x_t^2, ..., x_t^n$ correspondent soit aux sorties de n LFSR à l'instant t pour le schéma :

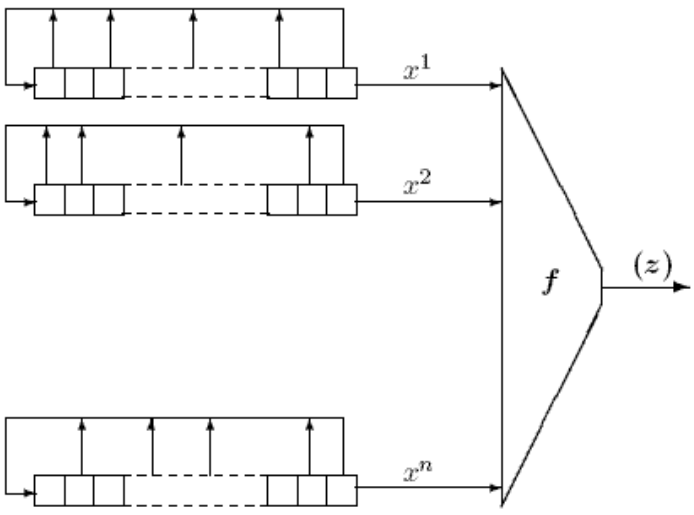

Figure 1.04 : LFSR combinées

5. Implémentation de la cryptographie dans le modèle OSI [10] [11] [12]

5.1 Le système OSI

L'ISO a spécifié un modèle d'architecture des logiciels pour les réseaux de télécommunications où les fonctionnalités réseaux sont réparties dans sept couches logicielles. Ce modèle s'appel OSI. Chaque couche possède son rôle et remplie des fonctions bien définies et limitées.

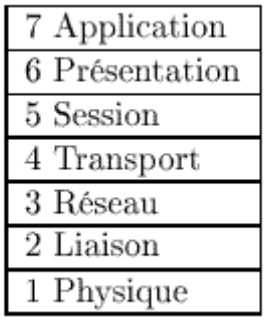

Figure 1.05 : Les sept couches du modèle OSI de l'ISO

5.1.1 La couche physique :

La couche physique fournit les moyens mécaniques, électriques, fonctionnels et procéduraux nécessaires à l'activation, au maintien et à la désactivation des connexions physiques destinées à la transmission de bits entre deux entités de liaison de données.
Sa fonction principale est le transport des informations.

5.1.2 La couche liaison de donnée :

La couche liaison de données fournit les moyens fonctionnels et procéduraux nécessaires à l'établissement, au maintient et à la libération des connexions de liaison de données entre entités du réseau. Elle détecte et corrige, si possible, les erreur dues au support physique et signale à la couche réseau les erreurs irrécupérables. Elle supervise le fonctionnement de la transmission et définit la structure syntaxique des messages, la manière d'enchaîner les échanges selon un protocole normalisé ou non.

5.1.3 La couche réseau :

La couche réseau assure toutes les fonctionnalités de relais et d'amélioration de services entre entité de réseau, à savoir : l'adressage, le routage, le contrôle de flux et la détection et correction d'erreurs non réglées par la couche liaison de donnée

5.1.4 La couche transport :

La couche transport assure un transfert de données transparents entre entités de session et en les déchargeant des détails d'exécution. Elle a pour rôle d'optimiser l'utilisation des services de réseau disponibles afin d'assurer au moindre coût les performances requises par la couche session.

C'est la première couche à résider sur les systèmes d'extrémité. Elle permet aux deux applications de chaque extrémité de dialoguer directement indépendamment de la nature des sous réseaux traversés et comme si le réseau n'existait pas.

5.1.5 Les couches hautes : session, présentation et application.

Les couches session, présentation et application constituent les couches hautes du modèle OSI et offrent des services orientés vers les utilisateurs alors que les couches basses sont concernées pas la communication fiable de bout en bout. Elles considèrent que la couche transport fournit un canal fiable de communication et ajoutent des caractéristiques supplémentaires pour les applications.

5.1.6 La couche session.

La couche session fournit aux entités de la couche présentation les moyens d'organiser et synchroniser les dialogues et les échanges de données.

5.1.7 La couche présentation.

La couche présentation s'occupe de la syntaxe et de la sémantique des informations transportées en changeant notamment de la représentation des données.

5.1.8 La couche application.

La couche application donne au processus d'application le moyen d'accéder à l'environnement OSI et fournit tous les services directement utilisables par l'application, à savoir :

- Le transfert d'informations
- L'allocation de ressources
- L'intégrité et la cohérence des données accédées
- La synchronisation des applications coopérantes

La figure 1.06 montre ce qui se passe entre la communication des couches :

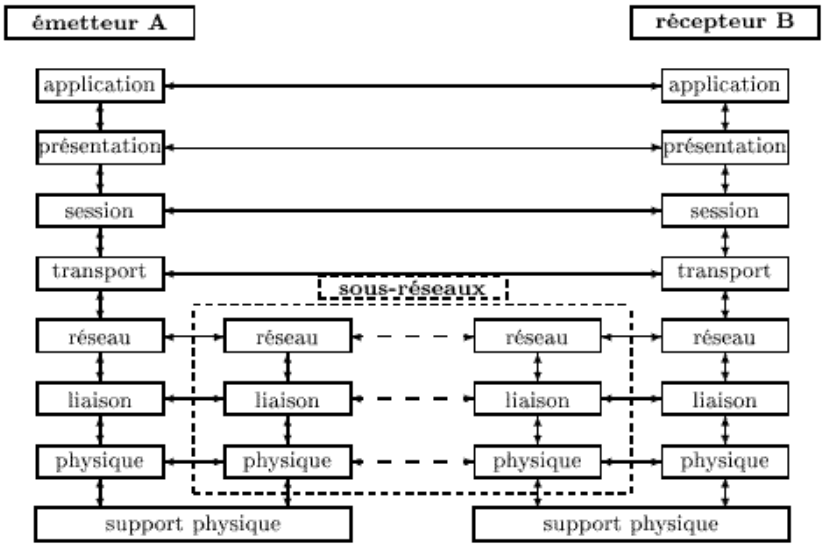

Figure 1.06 : Communication entre couches.

5.2 La cryptographie dans le modèle OSI.

L implémentation de la cryptographie dans le modèle OSI, peut s'effectuer en deux types : dans les couches basses ou dans les couches hautes :

- Si le chiffrement a eu lieu dans les couches basses on parle de chiffrement lien par lien, les données qui passe par le lien sera chiffré.
- Si le chiffrement a eu lieu dans les couches hautes, on parle de chiffrement de bout en bout. Les données restent chiffrées jusqu'à ce que le destinataire le déchiffre.

5.2.1 Le chiffrement lien par lien.

L'endroit le plus simple pour effectuer le chiffrement se trouve dans la couche physique. Le dispositif de chiffrement est placé à cet endroit. Il chiffre ensuite toutes les informations qui passe entre lui et l'autre dispositif reverse.

La sécurité de cette méthode réside sur le fait que les données qui circulent dans le réseau sont cryptées. Comme toutes les informations sont cryptées, l'interception des données ne corrompre personne (on ne connaît ni le contenu ni sa destination).

5.2.2 Le chiffrement de bout en bout.

Dans ce cas on place le dispositif de chiffrement entre la couche réseau et la couche transport. Les données chiffrées sont les unités de transport

6. La cryptanalyse [1] [2] [3] [5] [6]

Si le but de la cryptographie est d'élaborer des méthodes de protection, le but de la cryptanalyse est au contraire de casser ces protections. Une tentative de cryptanalyse d'un système est appelée une attaque, et elle peut conduire à différents résultats :

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

6.1 Attaques des fonctions de chiffrement.

Les fonctions de chiffrement sont supposées rendre impossible le décryptement, c'est-à-dire la récupération d'un message clair sans clé. A fortiori, elles doivent protéger le secret des clés.

6.2 Classement des attaques en fonction des données dont dispose le cryptanalyste

On distingue plusieurs types d'attaques suivant les informations que peut obtenir le cryptanalyste :

- L'attaque à texte chiffré seulement, où le cryptanalyste ne connaît qu'un ensemble de textes chiffrés ; il peut soit retrouver seulement les textes en clair, soit retrouver la clé. En pratique il est très souvent possible de deviner certaines propriétés du texte en clair, ce qui permet de valider ou non le décryptement.
- L'attaque à texte en clair connu, où le cryptanalyste connaît non seulement les textes chiffrés, mais aussi les textes en clair correspondants ; son but est alors de retrouver la clé. Du fait

de la présence, dans la plupart des textes chiffrés, de parties connues, ce type d'attaque est très courant.

– L'attaque à texte en clair choisie, où le cryptanalyste peut, de plus, choisir des textes en clair à chiffrer et donc utiliser des textes apportant plus d'informations sur la clé. Si le cryptanalyste peut de plus adapter ses choix en fonction des textes chiffrés précédents, on parle aussi d'attaque adaptive.

– L'attaque à texte chiffré choisi, qui est l'inverse de la précédente : le cryptanalyse peut choisir des textes chiffrés pour lesquels il connaîtra le texte en clair correspondant ; sa tâche est alors de retrouver le clé. Ce type d'attaques est principalement utilisé contre les systèmes à cl é publique, pour retrouver la clé privée.

6.3 Attaques sur les algorithmes symétriques.

6.3.1 Attaques au niveau des clés.

L'attaque la plus simple sur le plan conceptuel est l'attaque exhaustive ou attaque en force brute, ce qui consiste à essayer toutes les clés possibles. Avec une clé de 56 bits par exemple, cela nécessite 2^{56} tests.

Si la clé recherchée est en fait un mot de passe ou si elle dérive d'un mot de passe, on a des chances de trouver la clé en testant des mots susceptibles d'avoir été choisis comme mot de passe. On parle d'attaque par dictionnaire, car le cryptanalyste se constitue un dictionnaire de mots à tester. Ce type d'attaque est bien sûr beaucoup plus rapide qu'une attaque exhaustive.

6.3.2 La cryptanalyse différentiel

La cryptanalyse différentielle est un type d'attaque qui peut être utilisé contre les algorithmes de chiffrements par blocs itératifs. Elle utilise une attaque à texte en clair choisie et base sur l'observation de l'évolution des différences entre deux textes lorsqu'ils sont chiffrés avec la même clé. En analysant ces différences entre paires de textes, il est possible d'attribuer des probabilités à chaque clé possible. A force d'analyser des paires de textes, on finit soit par trouver la clé recherchée, soit par réduire suffisamment le nombre de clés possibles pour pouvoir mener une attaque exhaustive rapide. La cryptanalyse différentielle peut également être utilisée pour attaquer d'autres types d'algorithmes comme les fonctions de hachage.

6.3.3 La cryptanalyse linéaire.

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

6.4 Attaques sur les algorithmes asymétriques.

6.4.1 Attaques au niveau des clés.

Avec les algorithmes asymétriques, le problème n'est pas de trouver la bonne clé par attaque exhaustive, mais de dériver la clé secrète à partir de la clé publique. La plupart des algorithmes asymétriques reposant sur des problèmes mathématiques difficiles à résoudre, cela revient généralement à résoudre ce problème. C'est pourquoi les paramètres qui influencent la difficulté de résolution du problème sont les éléments déterminant la sécurité. Généralement, cela se traduit par la nécessité d'utiliser de grands nombres, la taille de ces nombres ayant une répercussion sur la longueur des clés. Cela explique que les clés utilisées par la cryptographie à clé publique soient généralement bien plus longues que celles utilisées par la cryptographie à clé secrète.

6.4.2 L'attaque à texte clair deviné et problème de la faible entropie

Un point faible des algorithmes à clé publique est le caractère publique de la clé de chiffrement : le cryptanalyste ayant connaissance de cette clé, il peut mener une attaque à texte en clair deviné, qui consiste à tenter de deviner le texte en clair et à le chiffrer pour vérifier son exactitude. Cette particularité implique donc une restriction sur l'utilisation des algorithmes asymétriques : il faut absolument éviter de les utiliser avec un ensemble de textes en clair possibles restreint (c'est-à-dire de faible entropie). En effet, si tel était le cas, un attaquant pourrait aisément se constituer une liste exhaustive des textes en clairs possibles et des textes chiffrés correspondants. Il n'aurait alors aucun mal à retrouver le texte en clair correspondant à un cryptogramme donné.

6.4.3 Attaque à texte chiffré choisi.

D'autre part, la plupart des algorithmes asymétriques sont sensibles aux attaques à texte chiffré choisi. Il convient donc, si l'on utilise le même algorithme pour le chiffrement et pour la signature, de s'assurer que les protocoles employés, ne permettent pas à un adversaire de faire

signer n’importe quel texte. Mieux, on peut avoir recours à des couples distincts pour ces deux opérations.

6.4.4 Attaque temporelle.

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

6.5 Attaques des générateurs pseudo-aléatoires.

Les générateurs pseudo aléatoires sont supposés être imprévisibles. Un modèle d’attaque courant consiste à observer les aléas engendrés pendant un certain temps pour deviner la suite, voire l’état du générateur.

CHAPITRE 2 THEORIES DE NOMBRES ET LA CRYPTOGRAPHIE.

1. Elément de théorie des nombres [1] [3]

La théorie de l'information est due à Shannon. Ses liens avec la cryptographie ont été développés par Hellman.

Cette notion permet de mesurer la sécurité apportée par un système de cryptographie et ainsi de permettre de reconnaître lequel des systèmes cryptographiques est plus sur et adéquat.

1.1 Entropie.

L'entropie est une notion de la physique, plus précisément dans la thermodynamique. Elle consiste en une mesure de l'état de désordre d'un système d'atomes ou molécules. L'entropie augmente lorsque le système évolue vers un état de plus grand désordre, et elle diminue si le système évolue vers un état plus ordonné.

Définition de Shannon. Si E est une expérience aléatoire qui admet $e_1, e_2, ..., e_k$ comme résultats possibles, avec les probabilités respectives $p_1, p_2, ..., p_k$, alors l'entropie de E notée H(E), est donné par la formule :

$$H(E) := p_1 \log_2(\frac{1}{p_1}) + p_2 \log_2(\frac{1}{p_2}) + ... + p_k \log_2(\frac{1}{p_k}) \qquad (2.01)$$

1.2 Quantité d'information.

L'entropie d'un texte peut mesurer la densité d'information de ce texte. L'idée est ici que plus un texte est dense en contenu, moins il est possible de le remplacer avec un texte plus cours. Imaginons que nous avons condensé au maximum l'information contenue dans un texte sous la forme d'une suite de bit. Le nombre de ces bits serait alors considéré comme donnant la quantité d'information contenu dans le texte.

Pour illustrer que l'entropie mesure la quantité d'information, considérons l'expérience qui consiste à deviner une boule au hasard dans un boulier contenant des boules numérotées de 1 à 16. On cherche à savoir le nombre minimum de questions qui devrait être posées, à une personne, pour arriver à déterminer le numéro X de la boule cachée. On suppose ici que les seules réponses possibles sont oui ou non.

Ainsi chaque réponse nous apporte au plus un bit d'information. L'entropie de E est

$$H(E) = \log_2 16 = 4 \qquad (2.02)$$

On interprète cette valeur comme la quantité d'information nécessaire pour déterminer à coût sur le résultat. Il faut donc 4 bits d'information pour déterminer le résultat.

1.3 Système cryptographique et théorie de l'information.

Nous allons maintenant aborder la cryptanalyse du point de vue de la théorie de l'information. Afin d'étudier les qualités du système cryptographique étudié, on s'imagine que l'entropie de messages est une expérience aléatoire, qui fait intervenir trois variables aléatoires M, K et C. La première M donne le message à envoyer, K correspond à la clé de chiffrement choisit et C le message chiffré résultant. On peut dire alors :

- Le résultat de la variable aléatoire M est l'un des messages clairs possibles

$$\{m_1, m_2, ..., m_3\}$$

Qui est choisit par l'envoyeur avec une certaine probabilité P(M=m_i)=p_i.

- La variable aléatoire K donne l'une des clés possible :

$$\{k_1, k_2, ..., k_s\}$$

Selon le système choisie, et q_j est la probabilité que celui-ci soit k_j

- C donne le message chiffré résultant, et la probabilité d'obtenir le message chiffré c se calcul comme :

$$P(c) = \sum_{fk_i(m_j)=e} p_j q_j \qquad (2.03)$$

Ici la somme s'effectue sur tous les couples (k_i, m_j) constitués d'une clé k_i et d'un message clair m_j qui donne le message codé c.

On supposera que l'envoyeur et le receveur choisissent une clé avec un mécanisme aléatoire qui est indépendant du choix du message à coder. Il en résulte que

$$H(k \mid M) = H(K) \qquad (2.04)$$

D'autre part, lorsqu'on connaît la clé, on peut récupérer le message clair à partir du message chiffré. En conséquence, la quantité d'information

$$H(M) + H(K) \qquad (2.05)$$

globalement obtenu lorsqu'on connaît la valeur des deux variables aléatoires indépendantes M et K ; est donc égal à la somme

$$H(c) + H(K \mid C) \qquad (2.06)$$

De la quantité d'information, $H(C)$, qui est obtenu lorsqu'on apprend la valeur de C, et de la quantité d'information supplémentaire, $H(K|C)$, obtenu lorsqu'on découvre la clé. En formule, on a :

$$H(C)+H(K|C)=H(K)+H(M). \qquad (2.07)$$

Sous une forme un peut cachée, cette identité ne fait qu'affirmer que la connaissance du message chiffré ne contribue aucune information supplémentaire lorsqu'on connaît déjà le message clair et la clé. Elle nous sera utile.

Lorsqu'on peut réussir à récupérer la clé à partir de la simple analyse d'un message chiffré, on en conclut qu'aucune information nouvelle n'est obtenue si nous procure effectivement la clé. Du point de vue de l'entropie, cela correspond à la situation :

$$H(K|C)=0$$

Puisque l'entropie $H(K|C)$ donne l'incertitude sur la valeur de la clé, quand on connaît le texte codé.

Il est naturel de considérer que la quantité d'information transmise est l'entropie $H(M)$ du message en clair. Autrement dit, quelque soit son approche, un cryptanalyste doit absolument obtenir au moins $H(M)$ bits d'information pour récupérer toute l'information contenue dans le message M. Nous allons supposer qu'il connaît, au départ, que le message chiffré C.

Un système cryptographique est parfaitement sûr si la connaissance du message chiffré n'apporte aucune information sur le message clair. En terme de la fonction d'entropie, ceci revient simplement à dire qu'un système cryptographique est parfait si

$$H(M|C)=H(M) \qquad (2.08)$$

Soulignons que l'entropie $H(K|C)$ mesure la quantité d'information moyenne qu'on peut obtenir sur la clé secrète quand on connaît le texte codé, et $H(K)$ est la quantité d'information minimum qui est nécessaire pour retrouver K. D'une autre façon, $H(M|C)$ est la quantité d'information moyenne qu'on peut obtenir sur le message en clair, à partir de la connaissance du message chiffré correspondant.

2. Elément mathématique pour la cryptographie [1] [3] [14] [15]

Le but de ce paragraphe c'est de donner les quelques notions de base d'arithmétique, qui sont nécessaire quand on parle de cryptographie. Tout en bien précisant que ce ne sont que des notions, mais à l'heure actuelle les prés requis mathématiques pour faire la cryptographie sérieusement sont d'un autre niveau.

2.1 Les nombres premiers.

On ne considère que l'ensemble des entiers naturels

Définitions 2.01 Un nombre premier p est un nombre différent de 1 qui n'admet pas d'autres diviseurs que 1 et lui-même.

Un nombre qui n'est pas premier est appelé nombre composé.

Théorème 2.01 : Tout nombre admet au moins un facteur premier.

Théorème 2.02 : L'ensemble des nombres premiers est infini.

Théorème 2.03 : Tout entier peut se décomposer en produit de facteurs premiers.

Le problème de la factorisation en nombres premiers est un problème difficile qui nous le verrons a permis de mettre en place des systèmes cryptographiques presque inviolable. Le deuxième théorème s'interprète sur le plan cryptographique comme l'existence de grands nombres premiers. Ainsi en 1963, le plus grand nombre premier était 2^{86243}-1, soit un nombre de 30000 chiffres. Le record actuel est de un million de chiffres.

*2.2 Congruence dans *

Définition 2.02 : Les entiers relatifs *a* et *b* sont congru modulo n s'ils ont même reste par la division par n. Il revient au même de dire que *b-a* est multiple de n. On note :

$$a \equiv b[n] \text{ ou } a \equiv b(\text{mod } n). \qquad (2.09)$$

Théorème 2.04 : Si $x \equiv x'[n]$ et $y = y'[n]$, alors $x + y \equiv x' + y'[n]$ et $xx' \equiv yy'[n]$.

2.3 Ensemble quotient $\mathbf{Z}|n\mathbf{Z} = \mathbf{Z}_n$

Proposition 2.01 : La relation de congruence est une relation d'équivalence sur $\mathbf{Z}$.

Définition 2.03 : L'ensemble $\bar{x}$ des éléments congrus à x de $\mathbf{Z}$ est dit classe d'équivalence de x modulo n. L'ensemble des classes d'équivalence modulo n est noté $\mathbf{Z}|n\mathbf{Z} = \mathbf{Z}_n$.

Définition 2.04 : on définit deux lois internes sur $\mathbf{Z}_n$ par :

$$\overline{xy} = \overline{yx} \text{ et } \bar{x} + \bar{y} = \overline{x + y}. \qquad (2.10)$$

Théorème 2.05 : L'ensemble quotient $\mathbf{Z}_n$ muni des lois précédentes est un anneau.

2.3.1 Divisibilités dans **Z**

Définition 2.05 **:** Soient a et b deux éléments de avec a≠0, « a divise b » que l'on note $a|b$ s'il existe un $k \in$ telle que : b=ka. On dit aussi que b est multiple de a.

Théorème 2.06 : pour a et b éléments de **Z** , il existe un couple unique *(q,r)* de $\mathbf{Z}^2$ tel que :

$$a = bq + \text{r avec } 0 \le r < |b| \qquad (2.11)$$

On dit que q est le quotient et r le reste de la division euclidienne de a pour b.

a : la dividente.

b : le diviseur.

2.3.2 Plus petit commun multiple (ppcm) et plus grand commun diviseur (pgcd).

Définition 2.06 : soient *a* et *b* deux éléments de **Z** .

On dit que *m* est le ppcm de *a* et *b* si et seulement si :

- *a* divise *m* et *b* divise *m* : $a|m$ et $b|m$
- Si *a* divise *m'* et *b* divise *m'* alors *m* divise *m'* : $a|m'$ et $b|m' \Rightarrow m|m'$

On dit que *d* est le plus grand commun diviseur de *a* et *b* et noté *d = pgcd(a,b)* si :

- *d* divise *a* et *d* divise *b* : $d|a$ et $d|b$
- Si *d'* divise *a* et *d'* divise *b* alors *d'* divise *d* : $d'|a$ et $d'|b \Rightarrow d'|d$

Remarque : On trouve pour ces relations d'ordre $p\gcd(a,b) = \sup\{a,b\}$ et $ppcm(a,b) = \inf\{a,b\}$.

Définition 2.07 : Les éléments a et b de **Z** sont premiers entre eux, si et seulement si, ils n'ont pour diviseur commun que 1 et -1.

Théorème 2.07 : Les propriétés suivantes sont équivalentes

(i) *a* et *b* sont premiers entre eux.

(ii) *pgcd (a,b)=1*.

(iii) $\exists u, v \in \mathbf{Z}$; $au + bv = 1$.

(iv) $\forall z \in \mathbf{Z}\ \exists x, y \in \mathbf{Z}$; $ax + by = z$

Théorème 2.08 : $p\gcd(ac,bc) = |c|\, p\gcd(a,b)$. Ainsi $ppgd(a,b) = p\gcd(a,(b+na))$.

Théorème 2.09 : *a* et *b* sont premiers entre eux, si et seulement si, $\forall x \in \mathbf{Z}\ a|bx \Rightarrow a|x$.

Corollaire 2.01 : Si *p* est premier et divise un produit de facteur, alors il divise un des facteurs. Si *p* est premier et divise a^n, alors il divise *a*.

Théorème 2.10 : Tout entier relatif est produit de facteurs premiers et sa décomposition en facteurs premiers est unique à l'ordre prés des facteurs. On dit que $\mathbf{Z}$ est factoriel.

Corollaire 2.02 : On a :

$$\left| ab \right| = p\gcd(a,b)\, ppcm(a,b) \qquad (2.12)$$

2.4 Algorithme d'Euclide.

Cet algorithme a été établi et baptisé ainsi par Etienne Bézout, il permet de calculer le pgcd de deux entiers a et b en effectuant un nombre fini de divisions euclidiennes.

Soient a et b deux entiers positifs, on pose $r_0 = a$ et $r_1 = b$, et tant que $r_i > 0$ on effectue les divisions euclidiennes successives suivantes.

$$\begin{cases} r_0 = r_1 q_1 + r_2, & \text{où} \quad 0 \le r_2 < r_1 \\ r_1 = r_2 q_2 + r_3, & \text{où} \quad 0 \le \mathrm{r}_3 < r_2 \\ \ldots \\ r_{k-2} = r_{k-1} q_{k-1} + r_k, & \text{où} \quad 0 \le \mathrm{r}_k < r_{k-1} \\ r_{k-1} = r_k q_k + r_{k+1}, & \text{où} \quad 0 \le \mathrm{r}_{k+1} w < r_k \end{cases} \qquad (2.13)$$

Pour chaque $k \ge 0$, on a $p\gcd(a,b) = p\gcd(r_k, r_{k+1})$. La suite des restes $(r_1, r_2, r_3, \ldots)$ étant une suite strictement décroissante d'entiers positifs, on obtient nécessairement un reste nul au bout d'un nombre fini de divisions.

Soit r_n le dernier reste non nul. On a $r_{n+1} = 0$, ce qui signifie que

$$p\gcd(a,b) = p\gcd(r_n, r_{n+1}) = p\gcd(r_n, 0) = r_n . \qquad (2.14)$$

D'où l'algorithme d'Euclide : On effectue les divisions euclidiennes successives décrites ci-dessus jusqu'à obtenir un reste nul, le pgcd de *a* et *b* est le dernier reste non nul.

2.5 Algorithme d'Euclide étendu.

L'algorithme d'Euclide étendu permet de déterminer $d = p\gcd(a,b)$ ainsi que deux entier u et v vérifiant.

$$d = au + bv \qquad (2.15)$$

Reprenons la suite (2.13) des divisions euclidiennes et à chaque étape $k \ge 0$, calculons deux entiers u_k et v v_k tels que

$$r_k = au_k + bv_k . \qquad (2.16)$$

Un coup d'œil à (2.16) montre que $u_0 = 1,\ v_0 = 0,\ u_1 = 0$ et $v_1 = 1$.

On en déduit que pour $k \geq 0$, on a $\begin{cases} u_{k+1} = u_{k-1} - u_k q_k \\ v_{k+1} = v_{k-1} - v_k q_k \end{cases}$

Si $d = r_n$ est le dernier non nul, on a $u = u_n$ et $v = v_n$.

D'où l'algorithme d'Euclide étendu :

Soient deux entiers positifs a et b. Pour déterminer $d = p\gcd(a,b)$, ainsi que deux entiers u et v tels que $d = au + bv$,

On écrit $\begin{cases} r_0 = a, \\ u_0 = 1, \\ v_0 = 0, \end{cases}$ $\begin{cases} r_1 = b, \\ u_1 = 0, \\ v_1 = 1, \end{cases}$ et $\forall k \geq 1,$ $\begin{cases} r_{k+1} = r_{k-1} - r_k q_k, \\ u_{k+1} = u_{k-1} - u_k q_k, \\ v_{k+1} = v_{k-1} - v_k q_k, \end{cases}$

Jusqu'à obtenir un reste nul.

Si r_n est le dernier reste non nul, on a $\begin{cases} d = r_n, \\ u = u_n, \\ v = v_n, \end{cases}$

2.6 Exponentiation modulo n.

Le but est ici de calculer l'expression : $a^e \bmod n$ avec *a=2,3,....* Quand on le calcul on constate que les résultats apparaissent de façon très aléatoirement dans l'ensemble des entier modulo *n.* Pour calculer de grandes puissances modulo *n*, on procède comme suit. On observe d'abord que les règles de calculs pour les exposants sont aussi valables modulos *n.* En particulier, on a les identités :

$$a^{2k} \equiv (a^2)^k \pmod{n}, \qquad \text{et} \qquad a^{2k+1} \equiv (a^2)^k \cdot a \pmod{n}, \qquad (2.17)$$

et on peut choisir de remplacer a^2 par son reste modulo *n*, chaque fois que cela permet de simplifier le calcul.

Exemple : calculons 2^{1024} modulo 23. on a :

$$2^{1024} = (2^2)^{512} = (4^2)^{256} = (16^2)^{128},$$

et donc

$$2^{1024} = 3^{128} \pmod{23},$$

puisque $16^2 = 256$ et $256 \equiv 3 \pmod{23}$. On peut continuer notre calcul en remarquant que

$$3^{128} = (3^2)^{64} = (9^2)^{32},$$

et, puisque $81 \equiv 12 \pmod{23}$, on a maintenant

$$2^{1024} \equiv 12^{32} \pmod{23}.$$

Le reste du calcul donne

$$\begin{aligned} 2^{1024} &\equiv (12^2)^{16} \pmod{23} \\ &\equiv (6^2)^8 \pmod{23} \\ &\equiv (13^2)^4 \pmod{23} \\ &\equiv (8^2)^2 \pmod{23} \\ &\equiv 18^2 \pmod{23} \\ &\equiv 2 \pmod{23} \end{aligned}$$

Lorsqu'en cours de route l'exposant est impair, on utilise la seconde formule de (2.17).

2.7 Théorème de Fermat et d'Euler.

Théorème 2.11 : Petit théorème de Fermat (Pierre de Fermat, 1601-1665) Etant donné un nombre premier p et un entier $a \in \mathbf{Z}$, on a

$$a^p \equiv a \pmod{p}. \qquad (2.18)$$

Définition 2.08 : Indicatrice d'Euler Soit n un entier positif, l'indicatrice d'Euler de n, notée $\varphi(n)$, est définie comme étant égal au nombre des entiers k vérifiant

$$(1 \le k \le n) \quad et \quad (p \gcd(k,n) = 1).$$

Notons que pour tout entier positif n, on a pgcd (1,n) = 1, ce qui fait que $\varphi(n) \ge 1$.

Théorème 2.12 : (Euler) Soit n un entier positif, et soit $a \in \mathbf{Z}$ un entier premier avec n, alors $a^{\varphi(n)} \equiv 1 \pmod{n}$.

2.8 Système de congruence. Théorème chinois.

Un système de congruences est un système de la forme :

$$(S): \quad \begin{cases} x \equiv a_1 \pmod{m_1}, \\ x \equiv a_2 \pmod{m_2}, \\ \ldots \quad\quad \ldots \\ x \equiv a_k \pmod{m_k}. \end{cases} \qquad (2.19)$$

Où les a_i, et les m_i sont des entiers donnés. Résoudre le système (S) consiste à déterminer tous les entiers $x \in \mathbf{Z}$ vérifiant le système.

Lorsque les entiers m_i sont premiers entre eux deux à deux, nous allons voir que le système (S) admet toujours des solutions. Ce résultat connu sous le nom de Théorème chinois parce que les Chinois en utilisaient des cas particuliers pour déterminer les dates de certains événements astronomiques.

Théorème 2.13 : (Théorème chinois) Soit m et n deux entiers premiers entre eux.

Pour tout couple d'entiers $(a,b)\in \mathbf{Z}^2$, il existe $c\in \mathbf{Z}$ que l'on ait l'équivalence :

$$(SC) : \begin{cases} x \equiv a \pmod{m}, \\ x \equiv b \pmod{n}. \end{cases} \quad \Leftrightarrow \quad (x \equiv c \pmod{mn}). \qquad (2.20)$$

Le système (SC) admet donc une infinité de solutions dans $\mathbf{Z}$, ce sont les entiers de la forme $x = c + kmn,\ où\ k \in \mathbf{Z}$. Deux solutions distinctes sont congrus modulos *mn*.

2.9 Décomposition d'un entier en facteurs premiers.

Proposition 2.01 : Soit p un nombre premier. Si p divise un produit $q_1 q_2 ... q_n$ de n entiers, il existe au mois un indice $i \in \{1, 2, ..., n\}$ tel que *p* divise q_i

Corollaire 2.03 : Soit p un nombre premier. Si p divise un produit $p_1 p_2 ... p_n$ de n nombres premiers, il existe un indice $i \in \{1, 2, ..., n\}$ tel que $p = p_i$.

Théorème 2.14 : (Théorème fondamental de l'arithmétique) Tout entier $a > 1$ s'écrit de façon unique :

$$a = p_1^{\alpha 1} p_2^{\alpha 2} ... p_n^{\alpha n}, \qquad (2.21)$$

Où $\begin{cases} \text{les entiers } p_i \text{ sont premiers et vérifient } p_1 < p_2 < ... < p_n, \\ \text{les entiers } \alpha_i \text{ sont positifs.} \end{cases}$

L'égalité (2.21) constitue la décomposition de l'entier a en facteurs premiers.

2.10 Résidu quadratique

Soit $n \geq 2$ un entier, a un entier inférieur à n. On dit que a est un résidu quadratique modulo n si $a = b^2$ pour une classe b. C'est-à-dire :

$$b^2 \equiv a \pmod{n} \qquad (2.22)$$

Dressons par exemple les listes des résidus quadratiques modulo quelques nombres premiers :

2 : 1

3 : 1

5 : 1, 4

7 : 1, 2, 4

11 : 1, 3, 4, 5, 9

13 : 1, 3, 4, 9, 10, 12

17 : 1, 2, 4, 8, 9, 13, 15, 16

19 : 1, 4, 5, 6, 7, 9, 11, 16, 17

23 : 1, 2, 3, 4, 6, 8, 9, 12, 13, 16, 18

29 : 1, 4, 5, 6, 7, 9, 13, 16, 20, 22, 23, 24, 25, 28

Notons qu'il y a $(n-1)/p$ résidus quadratiques pour chacun de ces nombres premiers et que -1 est un résidu quadratique pour 5, 13, 29.

2.11 Symbole de Legendre.

Définition 2.09 : Soit p un nombre premier et a un entier. On pose :

$$\left(\frac{a}{p}\right)=\begin{cases}0 & \text{si } p \text{ divise } a\\ 1 & \text{si } a \text{ est résidu quadratique modulo } p\\ -1 & \text{si } a \text{ est non résidu quadratique modulo } p\end{cases} \qquad (2.23)$$

On appelle symbole de Legendre l'expression $\left(\frac{a}{p}\right)$

2.12 Logarithme discret.

Le calcul du nombre x qui fait en sort que $y=b^x$, pour y et b donnés, correspond à trouver le logarithme, $\log_b(y)$. Dans le cas des entiers modulos un nombre premier, on peut considérer la même problème ; parce que les puissances successives b^x parcourent toutes les entiers $(\neq 0)$modulo p, lorsque x parcourt tous les entiers modulo $p-1$. Ainsi, avec $p=11$, on a

y	1	2	3	4	5	6	7	8	9	10
$\log_6(y)$	0	9	2	8	6	1	3	7	4	5

Tableau 2.01 : Logarithme discret de quelque entier

On obtient ainsi la notion de logarithme discret. Ici, le mot discret sert à distinguer de la notion usuelle qu'on qualifie souvent de continue. Pour p un nombre premier, et b entre 2 et $p-2$, on peut reformuler le théorème a comme

$$y \equiv b^x (\text{mod p}) \qquad (2.24)$$

Exactement lorsque

$$x \equiv \log_b(x) (\text{mod } p-1). \qquad (2.25)$$

3. La cryptographie classique [2] [3]

Quand on parle de cryptographie ce qui vient tout de suite à l'esprit c'est la cryptographie mise en œuvre par les anciens civilisations.

4. La cryptographie moderne [2] [3]

Les méthodes simples d'encryptage utilisent des tables, qui doivent alors se trouver dans les mains des deux partenaires : l'expéditeur et le destinataire. Ceci pose un sérieux problème qui est celui de l'acheminement et stockage sécurisé de cette table.

La cryptographie moderne repose sur la principe de clé : une suite de caractère en général hexadécimal, qui permet de chiffrer le contenu du message ou une signature et de le déchiffrer. Dans l'absolu, la longueur de la clé mesurée en nombre de bit définit la force de l'algorithme de chiffrement. Le principal avantage du chiffrement à clé est que l'algorithme peut être connu de tout le monde mais seul qui possède ou connaît la clé pourra lire le contenu du message. Il y a deux types de chiffrement moderne : le chiffrement à clé privée et le chiffrement à clé publique.

Dans la suite, on s'intéresse plutôt au chiffrement moderne qu'au chiffrement classique.

CHAPITRE 3 : LES ALGORITHMES DE CRYPTAGES

Comme on venait de décrire, il y a deux grands types d'algorithme de chiffrement moderne

- Le chiffrement à clé privée
- Le chiffrement à clé publique

Au tout début, il n'y a que le chiffrement à clé privée mais il y a le problème de distribution de clé. Alors le nouveau concept est né.

1. Définitions [1] [2] [3] [4] [5]

1.1 Chiffrement à clé privé

Le chiffrement à clé privé consiste à utiliser la même clé pour le chiffrement et le déchiffrement.

Par analogie c'est le principe d'une serrure d'une porte : tous les utilisateurs autorisées ont une clé identique.

Son avantage réside dans la rapidité par rapport au chiffrement asymétrique car il utilise une clé plus réduite.

Son principal inconvénient est la transmission au préalable de la clé au correspondant.

1.2 Chiffrement à clé publique

Les problèmes de distribution des clés sont résolus par la cryptographie à clé publique ou cryptographie asymétrique. Ce concept a été introduit par Whitfield a et Martin Hellman en 1975.

La cryptographie à clé publique est un procédé asymétrique utilisant une paire de clés asymétrique associé : une clé publique qui crypte des données et une clé privée ou secrète correspondante pour le décryptage. Vous pouvez ainsi publier votre clé publique tout en conservant votre clé privée secrète. Il est basé sur une méthode mathématique garantissant un encryptage facile et rapide, et un décryptage difficile. S'il fallait aussi un analogie, considérons que l'on crypte le message avec un cadenas (clé publique) que le détenteur de la clé privée peut ouvrir pour lire le cryptogramme. Il est impossible de retrouver la clé privée à partir de la clé publique.

2. Quelque Cryptosystème à clé privée [1] [2] [3] [4] [5] [6] [8]

2.1 Le DES et son successeur.

2.1.1 Historique.

Jusque dans les années 1970, seuls les militaires possédaient des algorithmes à clé secrète fiables. Devant l'émergence des besoins civils, le NBS (National Bureau of Standards) lança le 15 mai 1973 un appel d'offre dans le Federal Register (l'équivalent du journal Officiel américain) pour a création d'un système cryptographique. Le cahier des charges était le suivant :

- l'algorithme repose sur une clé relativement petite, qui sert à la fois au chiffrement et au déchiffrement.
- l'algorithme doit être facile à implémenter, logiciellement et matériellement, et doit être très rapide.
- Le chiffrement doit avoir un haut niveau de sûreté, uniquement lié à la clé, et nom à la confidentialité de l'algorithme.

Les efforts conjoint d'IBM, qui propose Lucifer (fin 1974), et de la NSA (National Security Agency) conduisent à l'élaboration du DES (Data Encryption Standard), l'algorithme de chiffrement le plus utilisé au monde durant le dernier quart du XX^{ième} Siècle.

2.1.2 Schéma général.

Le DES utilise une clé K de 56 bits, pour chiffrer des blocs de 64 bits, les blocs chiffrés obtenus ayant aussi 64 bits. Le bloc de texte clair subit d'abord une permutation initiale. Puis on itère 16 fois une procédure identique décrite ci-après, où la moitié droite est recopiée telle qu'elle à gauche, et la moitié gauche est transmise à droite en subissant au passage une modification dépendante de la clé. A la fin, on inverse les moitiés gauches et droites (ou bien, comme sur le schéma, on supprime le croisement de la dernière étape), et on applique l'inverse de la permutation initiale pour obtenir le bloc chiffré. Le schéma général de DES est donc le suivant (on a seulement représenté quelques-unes des 16 étapes) :

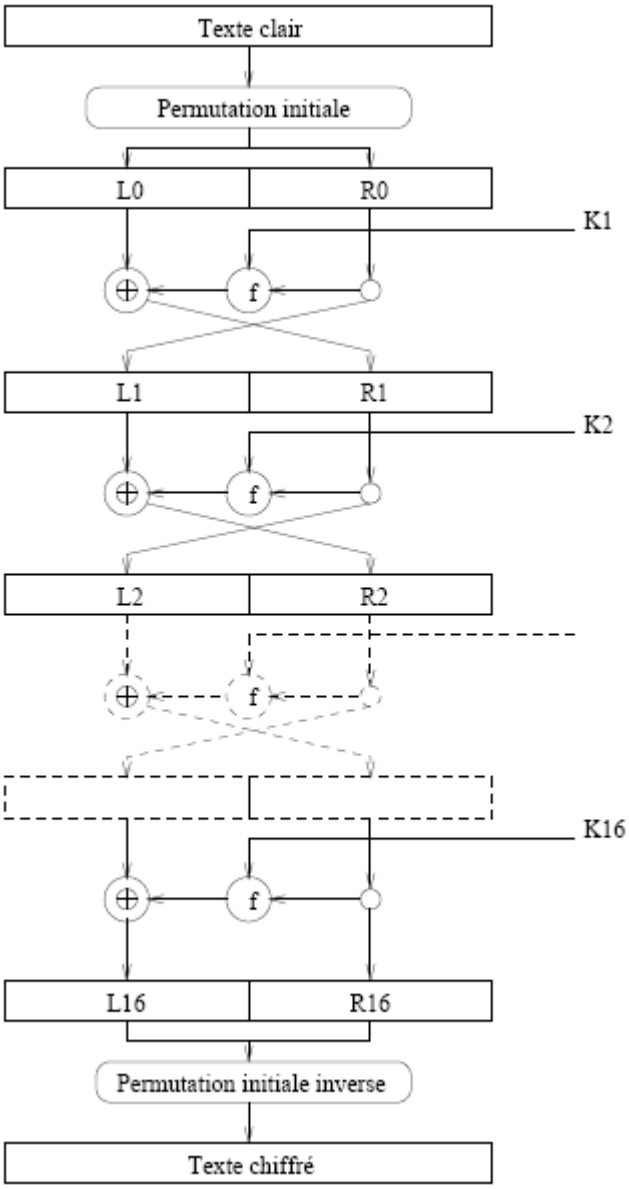

Figure 3.01 : DES : schéma général

Le tableau suivant montre la permutation initiale et son inverse :

Permutation initiale

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

Permutation initiale inverse

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

Figure 3.02 : La permutation initiale et son inverse

La permutation initiale et son inverse sont décrits par la figure 3.02. Les tableaux se lisent de gauche à droite et de haut en bas, le n-ième nombre est la position avant permutation du bit qui se trouve en n-ième position après permutation.

Après la permutation initiale, le message est séparé en deux moitiés de 32 bits, désignées par L_0 et R_0. A chaque itération de la procédure, on détermine deux groupes de 32 bits L_i et R_i en fonction de L_i-1 et R_{i-1} obtenus précédemment. Pour cela, on utilise une clé intermédiaire K_i de 48 bits, calculée à partir de K et on applique les formules suivantes :

$$L_i = R_{i-1} \qquad \text{et} \qquad R_i = L_{i-1} \oplus f(R_{i-1}, K_i). \qquad (3.01)$$

2.1.3 La fonction f

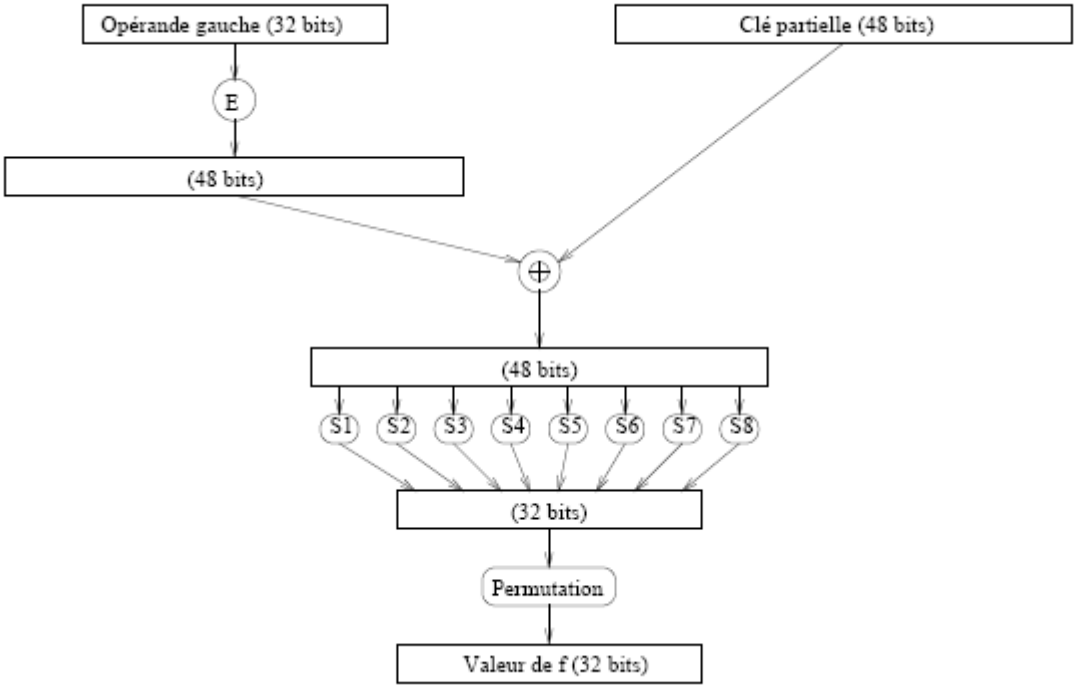

Figure 3.03 : Schéma de la fonction f

La fonction f est schématisée par la figure 3.03. Tout d'abord, l'argument de gauche qui possède 32 bits est expansé en 48 bits en redoublant certains bits. Cette expansion est précisée par la figure 3.02. Ensuite, on calcule le ou exclusif de cet argument expansé avec le deuxième argument (qui n'est autre que la clé K_i). Le résultat possède $48 = 8 \times 6$ bits et est transformé en une chaîne de $32 = 8 \times 4$ bits en utilisant des dispositifs appelés boite-S qui calculent un bloc de 4 bits à partir d'un bloc de 6 bits. Enfin on applique la permutation décrite par la figure à ces 32 bits pour obtenir la valeur de f

2.1.4 Les boites-S

Il y a huit boites-S différentes. On les représentes par deux tableaux à 4 lignes et 16 colonnes. Les premiers et derniers bits de l'entrée déterminent une ligne du tableau, les autres bits déterminent une colonne. La valeur numérique trouvée à cet endroit indique la valeur des quatre bits de sortie.

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

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

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

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

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

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

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

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

Figure 3.04 : Les boites-S

2.1.5 La diversification de la clé

Le principe de la diversification de la clé est schématisé par la figure 3.03. On applique une permutation PC_i à K. Puis, à chacune des 16 étapes, chaque moitié de la chaîne de 56 bits obtenu subit une rotation à gauche, d'un cran aux étapes 1, 2, 9, 16, et deux crans aux autres étapes. A chacune de ces étapes, on obtient une clé partielle de 48 bits en appliquant la règle d'extraction PC_2. La permutation PC_1 et la règle PC_2 sont détaillées par la figure 3.06. Les 56 bits de K y sont numérotés de 1 à 64 en évitant les multiples de 8, puisque dans la pratique, ces positions multiples de 8 sont utilisées pour insérer des bits de parité (la convention utilisée est fréquemment celle de la parité impaire).

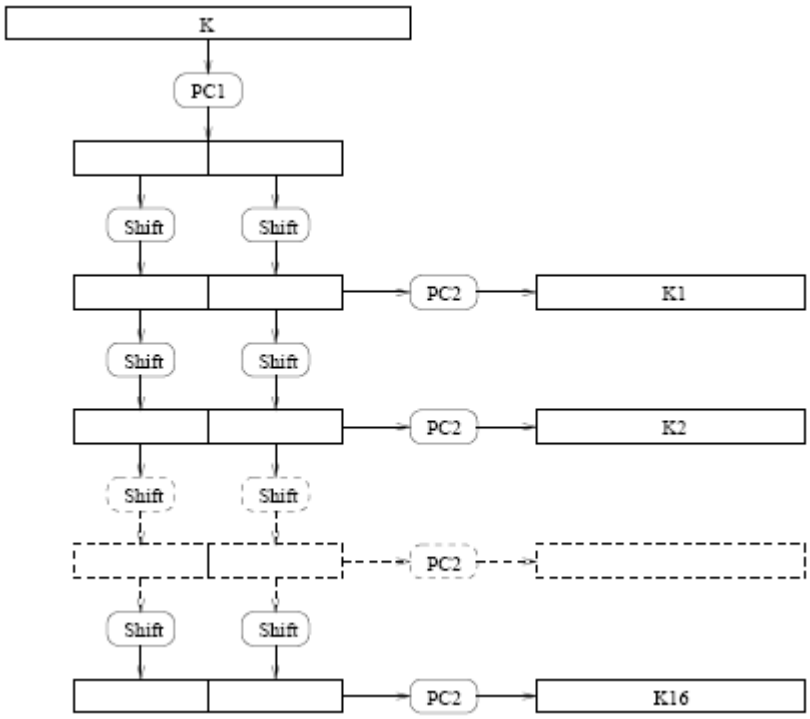

Figure 3.05 : diversité de la clé

Permutation PC_1

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

Règle d'extraction PC_2

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

Figure 3.06 : Permutation de la diversité

2.2 Le triple DES

Puisque la faiblesse de DES est la faible longueur de sa clé, il est naturel de chercher à combiner plusieurs chiffrements pour obtenir un système de chiffrement global avec une clé plus longue.

La première idée qui vient à l'esprit est de composer deux chiffrements DES avec des clés différentes. Mais on peut alors monter contre ce « double-DES » une attaque à message clair dite « par le milieu » parce qu'elle s'appuie sur le message intermédiaire (inconnu) apparaissant entre les deux chiffrements DES successifs. Cette attaque consiste à construire la liste des messages intermédiaires possibles en chiffrant par DES un clair avec les 2^{56} clés possibles. En déchiffrant

par DES le chiffré correspondant avec des clés différentes, on obtient une autre liste de messages intermédiaires possibles et le véritable message intermédiaire est dans l'intersection des deux listes. Le coût en mémoire de cette attaque est très important mais son coût en temps n'est pas significativement plus élevé que l'attaque exhaustive sur DES.

Triple-DES consiste à composer deux chiffrements DES de même clé séparées par un déchiffrement DES avec une autre clé. Plus précisément :

$$Triple - DES_{K_1,K_2} = DES_{K_1} \circ DES_{K_2}^{-1} \circ DES_{K_1} \qquad (3.02)$$

Cette façon de faire est préférée à trois chiffrements parce qu'elle généralise DES qui se trouve être le cas particulier où $K_1 = K_2$. Bien sûr, le déchiffrement est

$$Triple - DES_{K_1,K_2}^{-1} = DES_{K_1}^{-1} \circ DES_{K_2} \circ DES_{K_1}^{-1} \qquad (3.03)$$

Une clé triple DES est donc composée de deux clés DES et fait 112 bits ce qui met Triple DES largement hors de portée d'une attaque exhaustive. On peut aussi concevoir une variante à trois clés DES différentes mais elle reste vulnérable à une attaque de coût en 2^{112}s'appuyant sur l'un des deux messages intermédiaires.

2.3 L'IDEA

Développé à Jurich en Suisse par Xuejia Lai et James Massey en 1992, L'International Data Encryption Algorithm (IDEA) a été proposé pour remplacer le DES. Il utilise une clé de 128 bits, réalise un chiffrement par blocs de 64 bits, en opérant 8 rondes d'une même fonction. La description de cette fonction est un peu compliquée ; elle utilise seulement trois opérations :

- Ou exclusif
- Addition module 2^{16}
- Multiplication modulo 2^{16}

IDEA est particulièrement adapté aux réalisations logicielles. Il est aussi efficace même sur des processeurs de 16 bits.

2.3.1 Algorithme.

Le bloc de 64 bits en entré est tout d'abord divisé en quatre blocs de 16 bits : X1, X2, X3, X4. La clé K est, divisée en 8 blocs de 16 bits, puis décalée circulairement sur la gauche de 25 bits, et redivisée, et ainsi de suite jusqu'à obtenir 52 clés. Ces clés formeront 8 groupes de 6 clés (un

groupe par ronde) : K1, K2, K3, K4, K5, K6, un groupe de 4 clés pour la ronde finale : K1, K2, K3, K4.

Etapes des 8 rondes :

- Etape1 = $X1 * K1$
- Etape2 = $X2 * K2$
- Etape4 = $X3 * K3$
- Etape5 = Etape1 $\oplus$ Etape3
- Etape6 = Etape2 $\oplus$ Etape4
- Etape7 = Etape5 * K5
- Etape8 = Etape6+Etape7
- Etape9 = Etape8 * K6
- Etape10 = Etape7 + Etape9
- Etape11 = Etape1 $\oplus$ Etape9 $\Rightarrow$ X1 de la ronde suivante
- Etape12 = Etape3 $\oplus$ Etape9 $\Rightarrow$ X3 de la ronde suivante
- Etape13 = Etape2 $\oplus$ Etape10 $\Rightarrow$ X2 de la ronde suivante
- Etape14 = Etape4 $\oplus$ Etape10 $\Rightarrow$ X4 de la ronde suivante

Pour finir, on applique une étape supplémentaire après la huitième ronde :

- C1 = X1 * K1
- C2 = X2 + K2
- C3 = X3 + K3
- C4 = X4 * K4

Les 4 blocs C1, C2, C3, C4, forment alors le message chiffré.

Les 52 sous clés générées à partir de la clé de 128 bits sont produits comme suit :

- La clé de 128 bits est divisée en 8 blocs. Ces 8 blocs sont en faite les 8 premiers sous clés utilisees dans le chiffrement.
- La clé de 128 bits est ensuite cyclement décalée de 25 positions et divisée en 8 blocs de 16 bits. Ces 8 blocs sont les 8 sous clés suivantes utilisées dans le chiffrement.
- Le cycle est répété jusqu'à ce que les 52 sous clé soient toutes générées.

2.3.2 Déchiffrement :

Le déchiffrement s'effectue essentiellement à la même manière que le chiffrement. La seule différence est que les 52 sous clés sont générées de façon inverse au chiffrement. Aussi les blocs

de textes chiffrés doivent êtres traités dans l'ordre inverse de chiffrement pour inverser parfaitement le processus de chiffrement.

2.4 L'AES ou Rijndael.

L'Advanced Encription Standard est issu d'une compétition et d'une expertise qui s'échelonne de l'appel à candidature par le NIST (National Institute of Standard and Technology) en 1997 jusqu'à la sélection finale du candidat Rijndael en octobre 2000. Tout comme son prédécesseur, ce standard a été l'occasion de formaliser des critères de conception réunissant l'état d'art des dernières connaissances en cryptographie. Trois critères principaux ont été respectés dans sa conception :

- Résistance face à toutes les attaques connues
- Rapidités du code sur la plus grande variété de plateforme possible.
- Simplicité dans la conception.

La structure globale de l'AES à 128 bits de clé est décrite à la figure 3.04
Dans l'AES, le bloc comporte 128 bits. La clé peut comporter 128, 196 ou 256 bits, ce qui influent le nombre de tours du chiffrement. La version à 128 bits, comporte 10 itérations.

L'algorithme de cadencement de clé produit des sous clés de la même taille que la clé maître. La première itération est précédée par l'addition de la sous-clé numéro 0 qui correspond à la clé-maitre et la dernière itération ne comporte pas de *MixColums* ou fonction de brouillage de colonne. La fonction itérée se compose du quadruplet de fonctions (*SubBytes*, *ShiftRows*, *MixColums*, *AddRoundKey*), cette dernière fonction étant simplement l'addition bit-à-bit de la sous clé au bloc courant. On y retrouve trois étapes, conformément aux principes fondamentaux de confusion et de diffusion énoncés par Shannon. La première étape, dite de confusion, la fonction *SubBytes*, consiste à appliquer à chacun des 16 octets de l'entrée une même permutation S. Cette fonction correspond (à une application affine près) à la fonction inverse dans le corps fini à 2^8 éléments (dans la pratique elle est mise en table) ; elle assure la résistance de l'algorithme aux attaques différentielle et linéaire

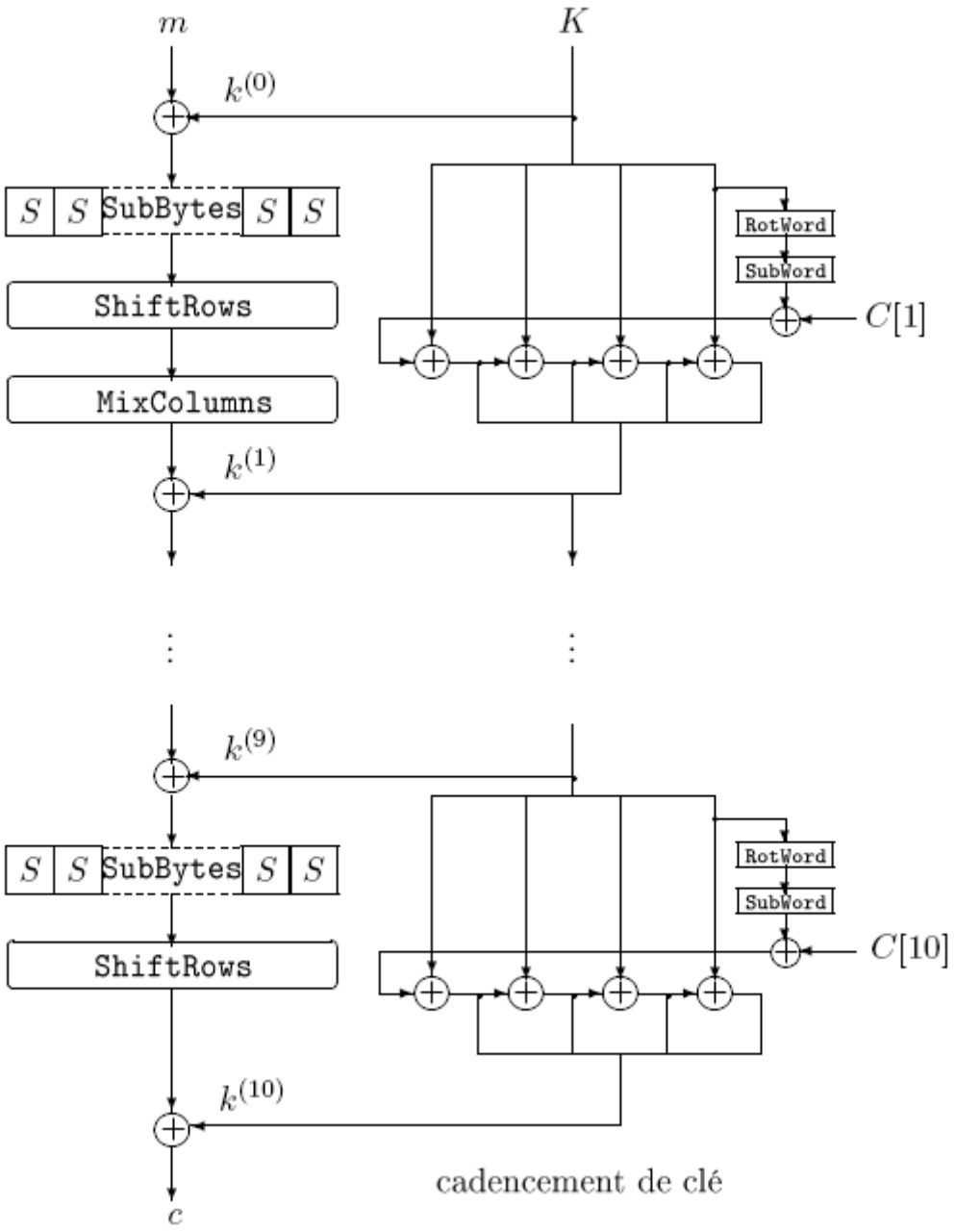

Figure 3.07 : L'AES

La phase de diffusion est composée des fonctions *ShiftRows* et *MixColums* qui représentent des opérations simples sur le corps à 2^8 éléments. Enfin on effectue un Ou exclusif bit-à-bit entre le résultat et la sous-clé de l'itération.

Les sous clés de 128 bits, numérotées de 0 à 10, sont dérivées de la clé secrète de la manière suivante : le sous-clé numéro 0 correspond à la clé secrète ; ensuite, la sous-clé numéro *i* (utilisée à la ième itération) est obtenu à partir de la sous-clé numéro *(i-1)* grâce à l'algorithme décrit à la figure 3.04. On permute de manière circulaire, par la fonction *RotWord*, les quatre

derniers octets de la clé numéro (i-1), puis on leur applique la fonction *SubWord* composée de 4 permutations S. Après avoir ajouté une constante (dépendant de i) au premier octet (les trois autres octets de la constante $C[i]$ du schéma sont nuls), on effectue une addition bit-à-bit entre les quatre octets ainsi obtenus et les quatre premiers octets de la sous-clés précédente. Les trois autres blocs de quatre octets de la clé numéro i sont ensuite simplement le résultat d'un Ou exclusif entre le bloc correspondant de la sous-clé (i-1) et le bloc précédent de la sous-clé i.

2.4.1 Déchiffrement :

Toutes les opérations réalisées lors de chiffrage sont réversibles à condition d'avoir la clé. Pour déchiffrer, on procède à l'extension de la clé de la même manière qu'un chiffrage. Les additions par Ou exclusifs lors de l'opération d'addition de la clé de la tour sont réversibles. L'opération de transformation *SubBytes* est inversée en utilisant la table S inversé. Les décalages de l'opération de décalage (*ShiftRows*) sont inversés, c'est-à-dire vers la droite. La manipulation matricielle de l'opération de brouillage (*MixColums*) nécessite une inversion de la matrice. Une fois la matrice inversée obtenue, la manipulation est la même que l'opération de brouillage des colonnes.

3. Chiffrement à clé publique [1] [2] [3] [4] [6]

3.1 RSA

En 1978, l'algorithme à clé publique de Ron Rivest, A Shamir, Leonard Adleman (RSA) apparaît. C'est un système à clé publique car l'algorithme n'est pas caché, ni la clé de chiffrement (appelé de ce fait clé publique).
Cet algorithme sert encore en 2002 à protéger les codes nucléaires de l'armée américaine et russe. Son fonctionnement est basé sur la difficulté de factoriser de grands entiers.

3.1.1 Fonctionnement.

Pour chiffrer le message l'émetteur va utiliser la clé publique que le destinataire a préalablement publiée. La clé publique est un ensemble de lettre et chiffre qui vont permettre à quiconque veut lui envoyés des messages confidentielles. Cela veut dire que tout le monde peut connaître la clé publique qui permet de cryptés les messages uniquement au destinataire qui a publié la clé mais seul ce dernier pourra déchiffrer ce qui a été codé avec sa clé publique grâce à sa clé privée.

3.1.2 Algorithme :

3.1.2.1 Génération de clé :

- Choisir deux nombres entier premier p et q, de l'ordre de 100 chiffres au minimum, pour rendre la factorisation hors de porter, même avec les meilleurs ordinateur.
- Calculer $n = p * q$
- Calculer $m = (p-1)(q-1)$
- Choisir un nombre entier e tel que $e > 2$ et $pcgd(e,n) = 1$
- Calculer d tel que $d * e * \bmod m = 1$

On prendra comme clé publique e et n et comme clé privée d et n.

3.1.2.2 Chiffrement

Connaissant la clé publique e et n, et le message à chiffrer M. On peut le chiffrer de la manière suivante :

Le message doit être remplacé par un chiffre. Ensuite on découpera le message par bloc de x longueur avec $x < n$. Le bloc B est chiffré par la formule :

$$C = B^e * \bmod(n) \qquad (3.04)$$

C'est un bloc de message chiffrer à envoyer vers le destinataire.

3.1.2.3 Déchiffrement

Pour le déchiffrement on va pratiquer quasiment de la même façon que le chiffrement mais avec la formule inverse :

$$b = C^d * \bmod(n) \qquad (3.05)$$

Ce qui permettra au destinataire de trouver le message clair.

3.1.3 Illustration.

Les auteurs du système ont fourni l'exemple suivant

$p = 47,\ q = 59$ et $n = pq = 2773$, d est premier avec $(p-1)(q-1) = 2668$ est choisi égal à 157. On trouve alors $e = 17$. En codant avec la règle : espace=00,A=01, B=02,…,Z=26 le message :

IT'S ALL GREEK TO ME

devient

M=0902 1900 0112 1200 0718 0505 1100 2015 0013 0500

Découpé en bloc de quatre chiffres. Le premier bloc M_1=0920 donne $C_1 = 920^{17} = 948[2773]$.

On répète l'opération pour les blocs suivants et on obtient à la fin :

C=0948 2342 1084 1444 2663 2390 0778 0774 0919 1655

Pour déchiffrer C_1, on calcule $948^{157}[2773]$ et on obtient bien 920, soit 09 20, c'est-à-dire IT.

Ce système est simple, facile à programmer et relativement rapide pour coder et décoder. Les algorithmes de factorisation sont beaucoup plus long que le tests de primalité ce qui rend le système RSA relativement sur et couramment utilisé.

3.2 Cryptosystème d'El Gamal.

Dans ce système, on suppose que les blocs de message clair sont numérisés dans les entiers modulo p. On commence par choisir un grand nombre premier p, et un nombre $g \pmod p$, qui sont tous deux connus de tous. L'utilisateur A choisit un grand nombre a $(\text{mod } p-1)$ qui sera sa clé secrète de déchiffrement. La clé publique de A est le nombre $g^a \pmod p$.

Pour envoyer un message m à A, l'utilisateur B choisit aléatoirement un grand nombre entier $k \pmod p$, et il l'envoie à A la paire

$$(K, M), \quad \text{où} \quad K = (q^K \bmod p), \quad \text{et} \quad M = (m \cdot g^{a-K} \bmod p). \qquad (3.06)$$

Le receveur A, qui connaît la clé secrète a, récupère le message m à partir de cette paire de la façon suivante. Il calcule d'abord $(K^{-a} \bmod p) = (g^{-a \cdot K} \bmod p)$, à partir du premier élément du couple reçu ; puis il multiplie M par ce résultat pour obtenir :

$$\begin{aligned} M \cdot g^{-aK} &\equiv (m \cdot g^{a \cdot K}) \cdot g^{-a \cdot K} \pmod p \\ &\equiv m \cdot g^{a \cdot K - a \cdot K} \pmod p \\ &\equiv m \end{aligned} \qquad (3.07)$$

Intuitivement, le message chiffré *C* envoyé à A est une version masquée de *m* obtenue par la multiplication par $g^{a \cdot K}$. Le nombre *K*, qui accompagne le message chiffré *C*, est un indice qui permet à A de retirer le masque. Cet indice $K = (g^K \bmod p)$ ne peut être utilisé que par quelqu'un qui connaît la clé *a*. Il semble que pour qu'un cryptanalyste puisse casser le cryptosystème de El Gamal, il doive retrouver la clé *a* à partir de la clé publique g^a. C'est donc dire qu'il aura trouvé une solution efficace au problème du calcul du logarithme discret.

3.3 PGP

Le PGP (Pretty Good Privacy) est un cryptosystème inventé par Phil Zimmermann en 1991. Il combine à la fois les meilleures fonctionnalités de la cryptographie à clé privée et de la cryptographie à clé publique. PGP est donc un système hybride

3.3.1 Chiffrement.

Quand un utilisateur chiffre du texte clair avec PGP, PGP compresse d'abord le texte clair. La compression de données économise le temps de transmission des données et de l'espace disque et, ce qui est important, renforce la sécurité cryptographique. La plus part des techniques de cryptanalyse exploitent les redondances trouvés dans le texte clair pour craquer le texte chiffré. La compression réduit ces redondances dans le texte clair, ce qui augmente grandement la résistance à la cryptanalyse.

PGP crée ensuite une clé de session, qui est une clé secrète qui ne sert qu'une fois. Cette clé est un nombre aléatoire généré à partir des mouvements aléatoires de votre souris et des touches du clavier sur lesquelles vous tapez. Cette clé de session fonctionne avec un algorithme de chiffrement conventionnel très sûr et rapide qui chiffre le texte clair ; le résultat est le texte chiffré. Une fois que les données sont chiffrées, la clé de session est elle-même chiffrée avec la clé publique du destinataire. Cette clé de session chiffrée par la clé publique est transmise avec le texte chiffré au destinataire.

3.3.2 Déchiffrement.

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

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

CHAPITRE 4 : LA CRYPTOGRAPHIE ET LA SECURITE DES RESEAUX DE TELECOMMUNICATIONS.

Quand on parle de réseau de télécommunication, il vient en tête tout de suite les réseaux téléphoniques. Dans ce chapitre nous allons parler de leur faille dans la sécurité et des solutions cryptographique mise en œuvre pour remédier certains de ces failles.

Jusqu'en 1990, le secteur des télécommunications était dominé par les réseaux téléphoniques fixes (jusqu'à Madagascar). Il n'y a pas encore alors d'interconnexion de différente types de réseaux, autrement dit le réseau est fermé, d'où sécurisés. Actuellement, la situation n'est plus la même, les réseaux mobiles et sans fil forment un vaste environnement qui s'étend de la transmission jusqu'aux services et ont pour objectif d'offrir des services similaires fixe (voix) et voir plus. L'interconnexion est alors inévitable le réseau devient ouvert.

1. Définition des vulnérabilités et des attaques de sécurité de la téléphonie [7] [12] [20]

Comme nous allons voir les fonctions de sécurité implantées dans les réseaux téléphoniques actuels, il est nécessaire de mentionner les menaces qui affectent la télécommunication de nos jours. Nous pouvons alors rencontrer au moins cinq failles dans le réseau téléphonique définies comme suit :

- Les écoutes téléphoniques : Un attaquant ou un espion passif écoute les données sans essayer de les changer. Nous ne pouvons pas détecter son écoute que lorsqu'il l'utilise ultérieurement.
- Tracer les déplacements d'un usager : la sécurité d'un système de télécommunication mobile n'est pas seulement reliée à la protection des informations de localisation, car un espion peut utiliser ces informations pour tracer les mouvements de l'utilisateur sur une carte.
- Modifier les données : Un espion qui a pu avoir accès au canal de données, peut non seulement écouter les informations, mais aussi modifier les données transmises.
- Rejeu : l'espion peut enregistrer l'information et la retransmettre pour son bénéfice.
- Accéder aux services avec l'identité d'autrui : l'attaquant peut jouer le rôle d'une autorité et persuader un employer de lui donner son mot de passe pour s'autoriser l'accès à la configuration et à la maintenance.

2. Réseaux traditionnels analogique et numérique [7] [12] [20]

Le premier service de télécommunication a été le service téléphonique. Lors de l'établissement d'un appel, la conversation comporte deux aspects : les signaux de signalisation et les signaux vocaux.

2.1 Architecture du réseau.

Pour relier les abonnés à tous les autres, il est nécessaire d'établir de milliard de liaison, ce qui est irréalisable. Aussi des aiguillages entre les différents niveaux de jonction du réseau sont réalisés par des commutateurs qui relient les terminaux (téléphones) entre eux. La commutation n'est pas uniquement la connexion d'une ligne appelante à un moyen banalisé de transmission, c'est aussi la réception et l'émission de signalisations et l'exécution de la logique du service de traitement des appels.

La signalisation est constitué des messages de gestion des appels, et permet la supervision des procédures telles que le traitement d'appel ou la fourniture de services avancés (rappel sur occupation, numéro universel, etc.) La signalisation sémaphore (ou SS7 S*ystème numéro 7*) définit la transmission de la signalisation sur les réseaux téléphoniques et décrit les couches et protocoles utilisés dans l'architecture du réseau. Il est indépendant du réseau d'acheminement du trafic (ou du plan usager). Le réseau sémaphore est donc équipé de commutateurs de paquets spécialisés (STP) à haut débit et à faible temps de traversée, qui ne font rien d'autre que déterminer le noeud suivant vers lequel les messages sémaphores doivent être routés et s'assurer de leur livraison dans le bon ordre.

2.2 Les Vulnérabilités de la téléphonie classique.

Nous pouvons rencontrer au moins quatre failles dans le réseau téléphonique traditionnel présentées comme suit :

- Ecoute illicite par accès physique au fils
- Vol de services
- Accès illicite, local ou à distance, au logiciel d'exploitation contrôlant le commutateur.
- Utilisation d'un émetteur de fréquence radio avec un microphone, connecté au lien physique ;
- Accès illicite aux bases de données.

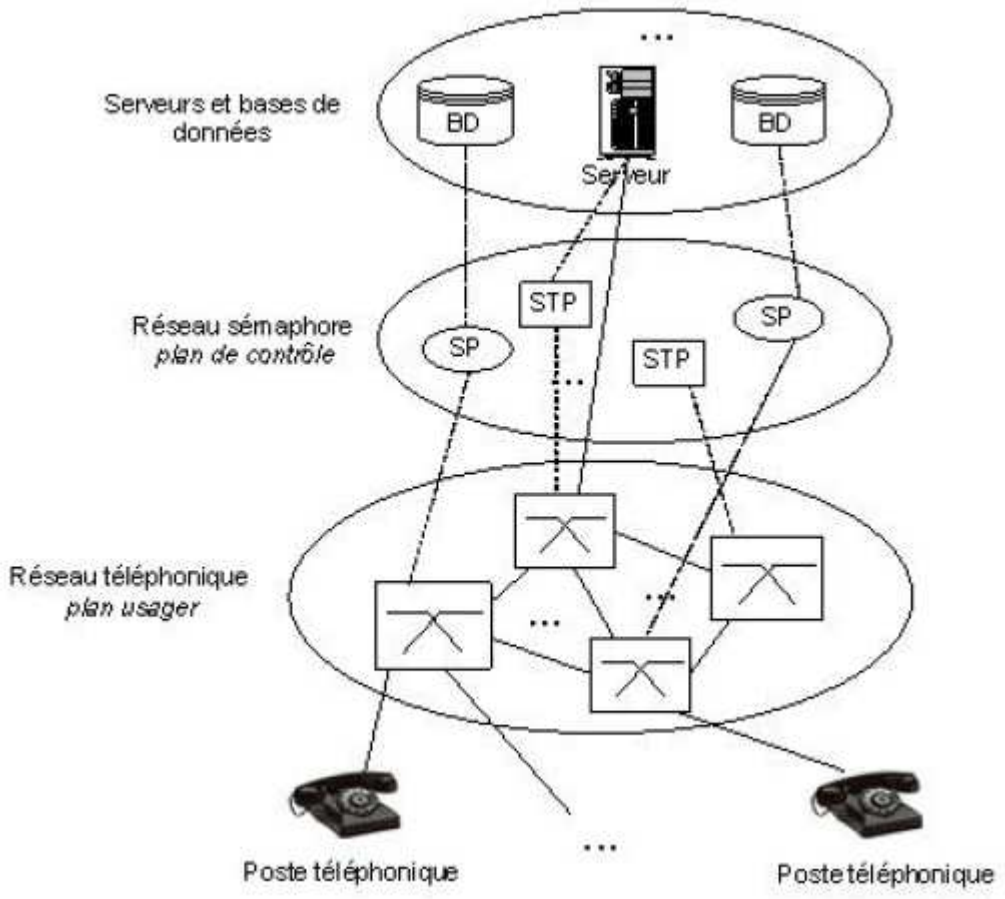

Figure 4.01 : synoptique d'un réseau de Télécommunication

2.3 La sécurité dans les réseaux téléphoniques traditionnels.

Sécuriser le réseau téléphonique revient à sécuriser aussi bien la conversation entre usager mais aussi les messages de signalisation.

La sécurisation de la téléphonie traditionnelle requiert outre l'utilisation d'équipements de cryptage pour chiffrer les conversations, une sécurisation du plan de contrôle et donc du réseau de signalisation.

Par la suite nous allons distinguer la sécurité offerte par le réseau analogique (RTC pour Réseau Téléphonique Commuté) et par le réseau numérique (RNIS pour Réseau Numérique à intégration de service).

2.3.1 Réseau Téléphonique commuté (RTC).

L'authentification dans le réseau RTC est une authentification non cryptographique assurée par l'envoi du numéro de la ligne (Caller ID) qui identifie uniquement la ligne mais pas l'utilisateur de la ligne. Les services de sécurité que l'on trouve dans le réseau téléphonique commuté sont essentiellement la confidentialité des conversations téléphoniques. Ainsi pour

contrer les failles citées dans le paragraphe précédent, la meilleure solution est d'utiliser des produits qui permettent de chiffrer les communications téléphoniques, les faxes, la transmission de fichier électronique. Le meilleur exemple que l'on peut citer est la ligne rouge établie entre le Kremlin à Moscou et la Maison Blanche à Washington.

Nous pouvons considérer trois catégories de mécanisme de sécurité :

- Brouillage simple par manipulation temporelle ou fréquentielle du signal analogique
- Brouillage par code glissant et cryptage analogique se basant sur des techniques de brouillage et de cryptage plus complexe.
- Cryptage en utilisant des algorithmes pour coder les signaux numériques par transposition de bit et/ou par opération de substitution. Cette technique s'applique à un signal vocal analogique après sa numérisation.

2.3.2 Réseau numérique (RNIS).

Le réseau numérique RNIS a été défini comme une solution globale de communication assurant les services multimédia (voix, données et vidéo). Comme le réseau traditionnel de la voix, ce réseau peut être la voie à des actes d'écoute sur les canaux de communications, d'accès aux informations critiques traversant jusqu'au réseau local. Des efforts ont été notés pour la définition des besoins de la sécurité dans ce réseau. Les services de sécurité sont similaires à ceux offerts dans le réseau RTC, sauf que le matériel pour le chiffrement du RTC n'est pas compatible avec le réseau RNIS du fait de l'architecture de ce réseau numérique. Des recherches ont été effectuées et un prototype de sécurité de niveau physique a été développé pour assurer la confidentialité d'utilisateur à utilisateur en appliquant le standard *TripleDES* comme algorithme de cryptage et une authentification de l'utilisateur.

2.3.3 Sécurisation à travers le réseau de signalisation

Des études ont été menées pour pouvoir sécuriser les communications de voix ; Une des propositions satisfaisantes a été faite à l'université de Georges Mason pour assurer la confidentialité des conversations et l'authentification des usagers. Cette proposition porte sur une sécurité de niveau applicatif et de bout en bout avec une modification minimale de l'infrastructure du réseau téléphonique public et une intégration du service de distribution des clés public aux téléphones. Ce modèle se base sur le chiffrement symétrique des signaux de voix entre les téléphones avec une clé symétrique à une seule utilisation unique. Ceci permet de prévenir les attaques par rejeu. Cette étude propose aussi des mécanismes de control d'accès des usagers et des

téléphones utilisés pour assurer les communications sécurisées. La technique d'authentification proposée est basée sur l'utilisation de cryptographie par clé publique. Elle offre au centre d'authentification la garantie de l'authenticité du téléphone à l'autre bout de la connexion.

3. Réseau mobile GSM/GPRS [7] [12] [13] [20]

3.1 Introduction.

Le réseau mobile GSM, destiné à transmettre de la voix en mode circuit, a bénéficié du système GPRS, qui utilise une technologie d'accès radio proche de celle de GSM et un système de transport de données en mode paquet avec le vecteur IP au niveau du coeur du réseau, pour faire évoluer son offre à des services de transmission de données en mode paquet et donc pouvoir offrir un service de voix sur IP. Dans un réseau de téléphonie fixe, à un numéro correspond une adresse physique fixe (une prise de téléphone), alors pour le réseau GSM/GPRS, le numéro d'un terminal mobile est une adresse logique constante à laquelle il faut associer une adresse physique qui varie suivant les déplacements de l'usager du terminal. La gestion de cette itinérance nécessite la mise en oeuvre d'une identification spécifique de l'utilisateur. De plus, l'emploi d'un canal radio rend les communications vulnérables aux écoutes et aux utilisations frauduleuses. La fraude des réseaux cellulaires, incluant le clonage et le piratage (*hijacking*) des appels de voix coûtera des millions de dollars par an. Les attaquants (*hackers*) ne volent pas uniquement des services, et violent la confidentialité des appels, mais aussi ils étaient capables de pénétrer dans les systèmes informatiques des opérateurs et de compromettre les bases de données de facturation et d'informations des appels.

3.2 Architecture du réseau.

Le GSM se différencie du RTC (ou PSTN), en introduisant des équipements spécifiques, qui n'existent pas dans le réseau RTC. Ces équipements, et plus généralement toutes les fonctions relatives à la gestion des utilisateurs mobiles, ont été regroupés dans un type de réseau particulier, appelé PLMN. Ces réseaux se raccordent aux PSTN et aux PLMN d'autres opérateurs par des passerelles spécifiées dans le standard GSM.

Dans notre étude nous limiterons sur une vue globale de l'architecture du réseau GSM/GPRS en soulignant certaines parties en relation avec la sécurité dans GSM/GPRS.

a- Le sous système radio ou BSS regroupe deux équipements concernés par la sécurités : qui sont le MS et la BTS

o MS (Mobile Station) ou terminaux mobiles : Chaque terminal est équipé d'une carte à puce, ou carte SIM (Subscriber Identity Module). Cette carte contient les informations relatives à l'abonnement de l'utilisateur, telles que ses identités, L'IMSI (International Mobile Subscriber Identity), et le TMSI (Temporary Mobile Subscriber Idenity), et ces algorithmes de chiffrement. Le terminal possède sa propre identité, l'IMEI (International Mobile Equipement Identity), qui permet de connaître, par exemple l'identité du constructeur du terminal mais aussi empêche l'utilisation de terminaux volés ou non-conformes.

o BTS (Base Tranceiver Sation) ou stations de base : les BTS forment les points d'accès au réseau GSM. Elle authentifie et chiffre le trafic entre elle et le terminal mobile.

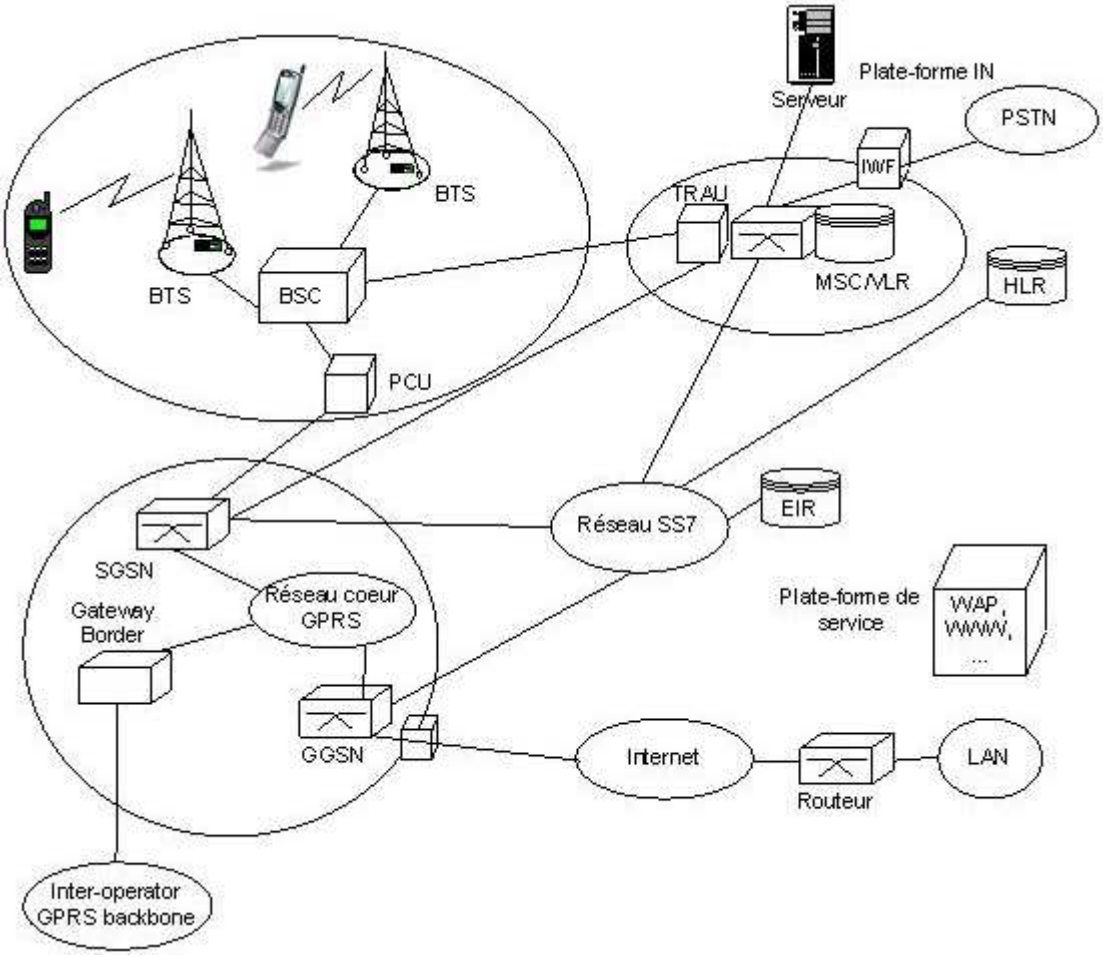

Figure 4.01 : Architecture GSM/GPRS

- BSC (Base Station Controller) ou contrôleurs de BTS : les BSC assurent le contrôle d'admission des appels, la gestion des handovers ou le contrôle de puissance est organisé par le BSC.

b- Le sous système réseau ou NSS s'occupe de l'interconnexion avec les réseaux fixes, publiques ou privés, auxquels rattacher le mobile. Il gère aussi l'établissement des communications avec les utilisateurs mobiles. Ses éléments sont :

- MSC (Mobile services Switching Center) ou commutateur de service. Qui gère toutes les communications avec les mobiles sous sa couverture.
- HLR (Home Location Register) ou base de données de localisation nominale où sont stockées toutes les informations relatives aux abonnés du PLMN.
- VLR (Visitor Location Register) ou base de données de localisation locale dans laquelle sont stockées les informations relatives aux utilisateurs d'une région particulière.
- Le sous-système réseau utilise deux autres bases de données, l'EIR qui contient la liste de tous les mobiles, identifiés par leur IMEI, autorisés à fonctionner dans le réseau, et l'AUC (*Athentification Center*), qui contient les codes PIN des cartes SIM.

c- Le sous système d'exploitation et de maintenance ou OMC assure une assistance distante des performances du réseau GSM et permet une reconfiguration à distance ainsi qu'une activité de gestion des fautes, des alarmes et des événements survenus.

L'introduction de la transmission en mode paquet dans GSM est réalisée avec le réseau GPRS (*General Packet Radio Service*) qui se greffe sur le réseau GSM existant, notamment pour la partie ressources de l'interface radio. Le réseau GPRS et le réseau GSM fonctionnent en parallèle : le premier est utilisé pour le transport des données, et le second pour les services classiques de voix. Tous deux utilisent les mêmes équipements du sous-système radio. C'est ensuite qu'ils se distinguent. Le réseau cœur du GPRS est un réseau paquet interconnecté, pouvant être relié à divers types de réseaux de données fixes – IP (*Internet Protocol*) – ou encore à d'autres réseaux GPRS, exploités par d'autres opérateurs. De nouveaux équipements, protocoles, interfaces, etc. sont intégrés au réseau GSM.

3.3 Vulnérabilités et failles du réseau GSM/GPRS

Le réseau GSM/GPRS présente quelques failles. Ceci est due peut être par sa conception, ou par technologie :

– La partie fixe du réseau présente un niveau de sécurité faible. Ce qui rend le niveau de sécurité du réseau GSM identique à celui du réseau fixe RTC ou RNIS.

– La protection n'a lieu que sur l'interface radio. Alors que les stations de base transmettent parfois leur trafic vers le MSC par micro-onde.

– L'interface radio n'est pas le seul point vulnérable dans le réseau GSM. Les données ne sont chiffrées que sur cette interface, elle est envoyée en clair à l'intérieur du réseau.

– Les liaisons entre les parties du réseau ne sont pas protégées. Sur ces liaisons des informations très importantes comme les identités des abonnées sont transmises.

– A sa conception le cassage de l'algorithme A5 n'est pas encore envisageable. Mais actuellement avec une longueur de clé de 64 bits, et qui a été démontré que la longueur effective est de 40 bits, ne présente plus une grande barrière.

3.4 La sécurité dans le réseau GSM/GPRS

L'utilisation du canal radioélectrique pour le transport d'information rend les abonnées vulnérables à la possibilité d'utilisation frauduleuse de leur compte.

Le système GSM intègre donc des fonctions de sécurité visant à protéger à la fois les abonnées et leurs opérateurs. Il assure l'authentification d'un abonné pour protéger l'accès aux services, la confidentialité de l'IMSI, des données usagers, et des informations de signalisation.
Pour mettre en oeuvre les fonctions d'authentification et de chiffrement des informations transmises sur la voie radio, GSM utilise les éléments suivants :

– Des nombres aléatoires RAND

– Une clé individuelle K_i pour l'authentification et la détermination de la clé de chiffrement $K_{c.}$

– Un algorithme A3 fournissant un condensât SRES à partir des arguments d'entrée RAND et de la clé K_i,

– Un algorithme A8 pour la détermination de la clé K_c à partir des arguments d'entrée RAND et K_i,

– Un algorithme A5 pour le chiffrement/déchiffrement des données à partir de la clé K_c.

A chaque abonné est attribué une clé Ki propre. Les algorithmes A3, A5 et A8 sont quant à eux les mêmes pour tous les abonnés d'un même réseau. Les services d'intégrité et de non rejeu ne sont pas assurés par le réseau GSM. Une non répudiation non cryptographique est similaire à celle assurée par le réseau RTC qui se base sur la fiabilité des factures de l'opérateur.

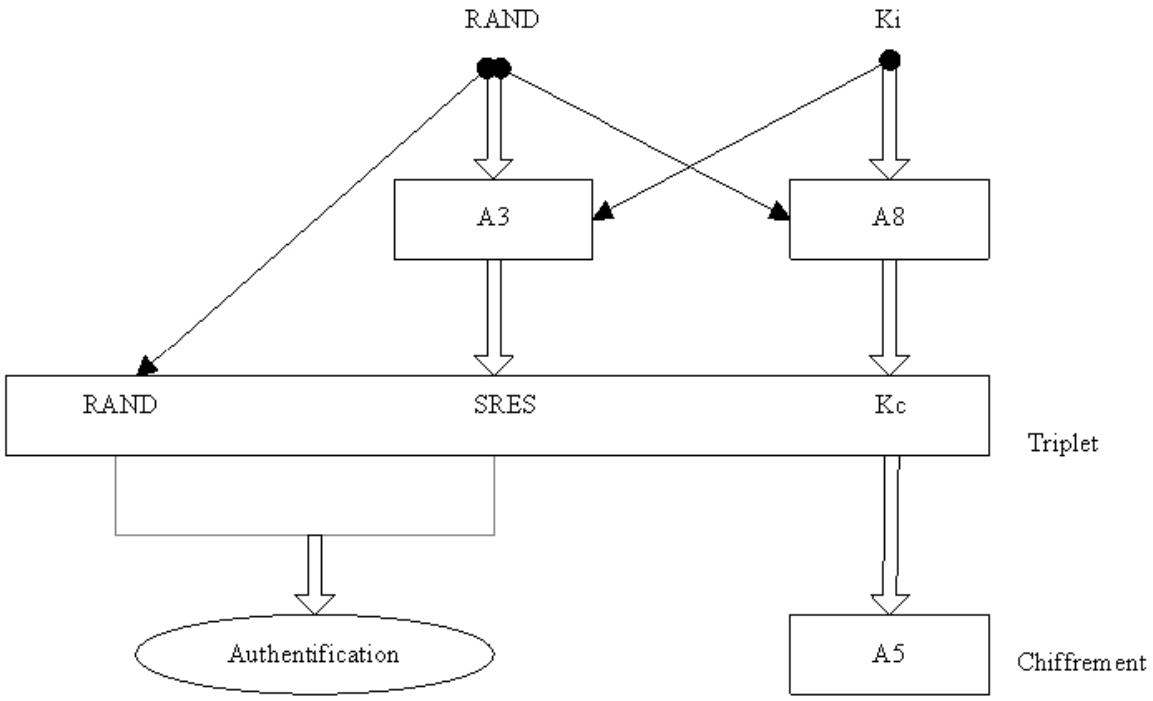

Figure 4.03 : Architecture de sécurité dans GSM

La sécurité dans le réseau GPRS (pour *General Packet Radio Service*) est très similaire au réseau GSM. Une des mesures de sécurité est l'utilisation des cartes à puce (SIM) protégée par un mot de passe (PIN). Les services de sécurité déployés dans le réseau GPRS sont :

- L'authentification : le routeur SGSN (pour *Serving GPRS Support Node*) contrôle le chiffrement et l'authentification du terminal mobile. Cette authentification est très similaire à celle offerte par le réseau GSM. Quand un abonné GPRS se connecte la première fois au réseau, le SGSN authentifie le mobile MS en utilisant la carte SIM. Il compare ces informations avec les informations d'authentification présentent dans la base de données du réseau (HLR). Durant cette phase d'authentification, un nombre aléatoire est utilisé pour générer la clé d'authentification afin de rehausser la sécurité. Cette clé n'est pas transmise sur aucune partie du réseau.
- La confidentialité : toutes les données transmises entre le mobile (MS) et le SGSN sont cryptées dans le réseau GPRS. Durant la phase d'authentification, la décision de crypter les données est prise. Le chiffrement est alors établi en générant la clé de chiffrement. L'algorithme utilisé pour le chiffrement est le GEA (*GPRS Encryption Algorithm*) qui est une version de l'algorithme A5 utilisé dans GSM.
- Le routeur GGSN (pour *Gateway GPRS Support Node*) est le point d'entrée entre le réseau Internet et le réseau GPRS. Pour sécuriser les terminaux mobiles contre les attaques en provenance de l'extérieur du réseau GPRS, des filtres de paquets (*firewalls*) GGSN sont placés pour filtrer le

trafic. Le SGSN protège l'usager contre les attaques provenant des autres mobiles. Le NAT (pour *Network Address Translation*) peut aussi être utilisé pour protéger les adresses IP.

– La protection des données s'arrête au niveau du GGSN. Pour sécuriser le trafic de bout en bout, un VPN peut être créé entre le GGSN et l'intranet des entreprises privées pour assurer la sécurisation du trafic de données et de voix.

4. Réseaux UMTS [7] [12] [21]

4.1 Introduction

L'UMTS est le réseau mobile offrant des services multimédias. L'UMTS peut être vu comme un système Internet mobile. La technologie de l'accès radio est basée sur W-CDMA. L'UMTS utilise un débit de 2Mbps pour les transferts de données.

4.2 Architecture du réseau.

L'UMTS est la troisième génération du réseau mobile standardisé par l'ETSI. C'est un réseau mobile puissant qui supporte le multimédia.
L'architecture de l'UMTS s'appuie sur la modularité. Ses éléments constituants doivent être indépendants, de façon à autoriser en théorie des mises à jour de telle ou telle partie du système sans savoir à en redéfinir la totalité.

La figure 4.04 illustre l'architecture générale de l'UMTS. L'UTRAN regroupe les stations de base, appelées NodeB dans le vocabulaire de l'UMTS, et les contrôleurs de stations de base, ou RNC (Radio Network Controller).Un RNC est l'équivalent d'un BSC dans un réseau GSM. Il contrôle les ressources radio de son domaine.

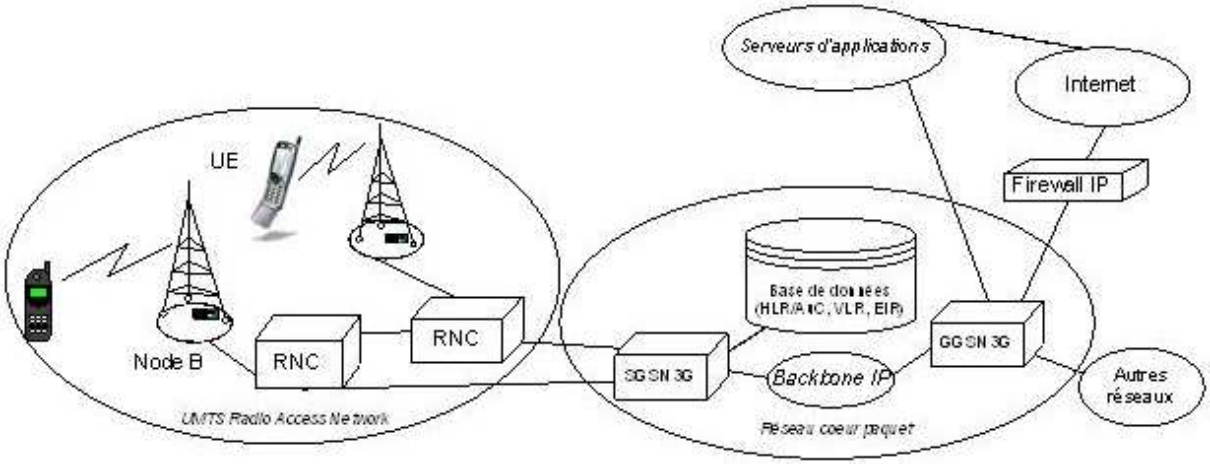

Figure 4.04 : Architecture du réseau UMTS.

Un réseau UTRAN peut inclure un ou plusieurs sous systèmes radio RNS (R*adio Network Subsystems*) ; ce dernier comprend un ou plusieurs NodeB. Le RNC de service de l'UE gère le NodeB pour le contrôle de charge et de congestion. Le rôle principal du NodeB (équivalent à la BTS dans GSM) est de réaliser le traitement des fonctions de la couche physique. Le RNC réalise les fonctions de couche 2 (*MAC*) pour les données en provenance ou vers l'interface radio ainsi que les fonctions de gestion de mobilité.

Divisé en deux parties, le réseau cœur est constitué d'un réseau cœur de type circuit (CS_Domain) et d'un réseau cœur de type paquet (PS-Domain). Le réseau cœur est composé, à l'image de celui du GSM, de commutateurs de circuits, les MSC (Mobile-service Switching Center), de passerelles vers les réseaux téléphoniques publics, les G-MSC (Gateway-MSC) et de serveurs dédiés aux SMS, les SMS-GMSC (SMS-Gateway MSC). Le réseau coeur orienté paquet est semblable à celui du GPRS. Il est composé de commutateurs paquet, les SGSNs et les GGSNs, qui relient le réseau de l'opérateur au monde extérieur, qui peut être un réseau paquet public ou un réseau paquet d'un autre opérateur. Le réseau situé entre les GGSNs et les SGSNs est un réseau paquet quelconque, le plus souvent un réseau IP. Dans la version UMTS 99, l'ATM est le protocole de transport choisi pour le réseau coeur.

Pour gérer les données relatives aux utilisateurs, telles que leur position dans le réseau, leur abonnement, etc., les bases de données introduites dans le GSM sont toujours présentes dans l'UMTS. Il s'agit, entre autres, des HLR, VLR et EIR.

4.3 Vulnérabilités dans le réseau UMTS

Les Vulnérabilités les plus connues du réseau UMTS peuvent se résumer comme suit :

- Manipulation des données transmises : des malveillants peuvent manipuler des données transmises sur toutes les interfaces du réseau UMTS.
- Manipulation des données enregistrées : des malveillants peuvent manipuler des données enregistrées dans des entités systèmes, dans des terminaux ou bien dans la carte USIM comme l'accès à l'IMEI.
- Ecoute des trafics des usagers, de la signalisation et des données de contrôle sur l'interface radio.
- Analyse de trafic : les malveillants peuvent observer le temps, le débit, la longueur, la source ou la destination des messages sur le lien radio pour accéder aux informations confidentielles.

– Durant la phase de signalisation, et pendant un laps de temps très court, les données de signalisation ne sont pas protégées et peuvent être exposées à des manipulations.

4.4 La sécurité dans le réseau UMTS.

UMTS comme GSM utilise un module d'identification qui est une care à puce appelé USIM. Dans cette USIM, les applications, les certificats, les signatures numériques, les algorithmes de chiffrement et autres types de données sont sauvegardée dans la mémoire ce cette carte. Elle est conçue sur une technologie d'intégration et de microchip très avancée, elle possède en outre une capacité de mémoire, d'interfaçage et de calcul très élevé. Ces cartes peuvent utiliser une technique dite sans contact qui ne nécessite pas l'insertion et l'extraction de la carte du terminal pour effectuer les opérations requises.

Les services de sécurité offerts par le réseau UMTS sont :

– Authentification et échange de clés (AKA) : l'authentification prouve l'identité entre l'usager et le réseau, elle est divisée en deux parties :

  - Authentification de l'usager par le réseau
  - Authentification du réseau par l'usager

La technique utilisée est appelée one-pass authentification qui permet de réduire les échanges de messages. Suite à cette procédure d'authentification, l'usager est sûr du réseau et vis versa. L'authentification est nécessaire aux autres mécanismes de sécurité comme la confidentialité et l'intégrité.

– La confidentialité : elle est assurée par le chiffrement des données des usagers et de signalisation entre l'usager et le réseau en utilisant l'identité temporaire (TMSI/P-TMSI) de l'usager à la place de son identité globale (IMSI). Le cryptage se fait entre la carte USIM et le RNC. La confidentialité de l'usager est assurée entre l'abonné et la VLR/SGSN. Si le réseau n'offre pas le service de confidentialité, l'usager est notifié et aura alors le choix d'accepter ou de refuser les connexions. Les éléments qui sont confidentiels sont l'identité de l'abonné, la localisation courante de l'usager, les informations (voix et données) de l'usager et les données de signalisation.

– L'intégrité : la protection de l'intégrité est de s'assurer que les messages entre l'usager et le réseau n'ont pas été manipulé, même si le message n'est pas confidentiel en soi. Ceci est assuré par un condensât ajouté à chaque message de signalisation. Pour cela, le réseau UMTS utilise une clé pré-partagée qui est enregistrée dans la carte USIM et le AUC pour générer le condensât. Au niveau du transport, l'intégrité est assurée par le calcul du CRC.

CHAPITRE 5 : CONCEPTION D'UN CRYPTOSYSTEME CLIENT/SERVEUR BASEE DE RSA

Dans ce chapitre, nous avons réalisés un cryptosystème Client/Serveur. Pour le réaliser, nous avons utilisés JBuilder 9 pour le développement. Ce système a deux types d'application : le cryptosystème proprement dite, qu'on considère aussi comme le client ; et le serveur de clé publique qui gère les clés publiques des clients connectés et qui achemine l'envoi du message.
Nous allons voir dans la suite les notions utiles pour la réalisation et les explications tout autour de ce système que nous avons mis au point.

1. Notion d'Architecture client/serveur [11] [16]

En général, il y a trois grands types d'architecture client/serveur. Et ils sont classés ainsi suivant les critères:

- type de donnée véhiculée,
- type de dialogue client/serveur,
- structure et localisation des applications.

1.1 Client/Serveur à client passif (1-tiers)

Dans ce type d'architecture le client est dit passif ; il ne fait aucune exécution. Le client se présente ici sous forme de terminal (écran-clavier). Le côté exécution et traitement de données sont centralisés dans le serveur. Le client ne sert alors que pour la visualisation et l'insertion des données. Le serveur est un ordinateur puissant, particulièrement stable et qui est spécialement conçu pour les traitements de gestion et qu'on appelle le mainframe.
Le flux réseau qui circule entre le client et le réseau n'est que des informations de présentation. Cette architecture fonctionne en mode connecté.

1.2 Client/Serveur de donnée (2-tiers).

Cette architecture est apparue avec la vague du PC. Une loi énoncée par Grosh démontre qu'a puissance égale, plusieurs petites machines coûtent moins cher qu'une grosse machine. Alors, la tendance des entreprises fut immédiate. Son principe de fonctionnement réside sur le fait que les PC sont capables d'exécuter des applications. Alors la majeure partie des traitements est déplacée vers le client, alors le serveur ne s'occupe que de la gestion des données. Les flux de réseaux entre

le client et le serveur sont des données et des requêtes. Elle fonctionne toujours en mode connecté. Son principal inconvénient est le coût de maintenance.

1.3 Client/Serveur distribuées (3-tiers).

Le principe de cette architecture est d'ajouter un autre serveur dit « tiers » pour l'exécution des applications et appelé serveur d'application. Son fonctionnement se fait de la manière suivante : le client demande au serveur d'application d'exécuter telle ou telle procédure, puis retourne le résultat au client. Le traitement des données se fait entre le serveur d'application et le serveur de données. D'où les données sont devenues plus sécurisées. Le type de flux réseau, qui existe entre le client et le serveur, est des requêtes objets. Cette architecture fonctionne aussi en mode connecté.

2. Les choix dans la réalisation [2] [3] [11] [16] [17] [18]

Durant la phase de réalisation du cryptosystème, des choix ont été pris entre divers autre. Le but de ce paragraphe est de justifier notre choix.

2.1 Choix de l'architecture Client/Serveur.

Nous avons en vue de réaliser une application qui utilise les traitements cotés clients. C'est au niveau client que le chiffrement des données a lieu. Le serveur ne sert qu'à gérer la commutation et les clés publiques des clients. Donc l'architecture la mieux adaptée pour notre réalisation est l'architecture Client/Serveur de données.

2.2 Choix de l'algorithme de chiffrement.

Le choix de l'algorithme de chiffrement est le plus difficile des choix. Mais nous avons réussis par trouver ce qui est mieux pour notre réalisation. Tout d'abord l'algorithme doit être un algorithme très sûr et qui a fait ses preuves durant des années. Ensuite, il doit être facile à comprendre et programmable (pas un logiciel fini). Et enfin, il doit avoir une capacité de chiffrement très avancé. Comme notre système est conçu pour Madagascar, l'utilisation de l'algorithme ne devra pas poser un problème d'acheminement de clé, d'où le besoin d'un algorithme à clé publique.

L'algorithme à clé publique RSA présente tous ces besoins, avec une existence de presque 30 ans ; et depuis, une résistance à la cryptanalyse très remarquable. En augmentant la taille de la clé RSA, elle devient difficile à casser, mais le chiffrement devient un peu long. En choisissant

une taille de clé raisonnable, cet algorithme présente encore une barrière infranchissable pour certaine technologie.

Dans notre réalisation, nous avons choisis une clé de longueur 512 bits (environ 154 chiffres décimaux). Ceci explique le fait que notre cryptosystème est conçu pour Madagascar. Même la taille de la clé préconisée actuellement est supérieure ou égale à 768bits (1024, 1536, 2048) pour que le temps de chiffrement et du déchiffrement soit raisonnable et acceptable.

2.3 Choix du langage de programmation.

Nous avons choisis java comme langage de programmation. En effet, nous avons étudiés au sein du département télécommunication durant trois ans. Ce choix s'explique comme suit :

- Java est un langage orienté objet
- Il est portable sur la plupart des plateformes (Windows, Linux, Solaris, …)
- C'est un langage généraliste ayant un vaste champ d'application (réseau, cryptographie, base de données, calcul scientifique,…). Il permet ainsi de développer des applications professionnelles de grande taille.
- Il intègre une interface graphique de haut niveau
- Il est plus sûr, car de nombreuses vérifications sont faites, aussi bien à la compilation qu'à l'exécution, pour limiter le nombre et la gravité des erreurs.
- Il existe de nombreuses bibliothèques de programmes dans des domaines très variés, de sorte que ce langage devient un langage professionnel de premier plan
- Le code produit (il s'agit d'un pseudo code ou byte-code) est indépendant de la plate-forme utilisée.

Même si java présente tous ces avantages (cette liste est non exhaustive), il présente certains inconvénients. Il faut remarquer une certaine lenteur à l'exécution.

Pour mettre en œuvre la programmation java, nous avons utilisés JBuilder 9, qui est un environnement de développement graphique IDE (Interface Development Environment). JBuilder aide le programmeur au développement du code source, plus précisément minimise les codes écrits.

3. Conception et réalisation [2] [3] [17] [18] [19]

Comme nous avons dits au tout début de ce chapitre, java permet de programmer différents types d'application. Une des grandes forces de java est de pouvoir développer une application réseau sans douleurs. Les concepteurs de la bibliothèque java en ont fait quelque chose

d'équivalent à la lecture et écriture de fichiers, à la différence que les « fichiers » résident sur une machine distante.

Java fournit plusieurs classes et interfaces destinées à faciliter l'utilisation du réseau par programmation. Elles sont regroupées dans java.net.

Les sept couches OSI sont alors respectées par le développement réseau. Celle qui nous intéresse est la couche transport qui est implémentée dans les protocoles UDP ou TCP. Ils permettent alors la communication entre des applications sur des machines distantes. Les deux protocoles se différencient par leur mode : UDP en mode non connecté, et ce n'est pas sûr. Et TCP est en mode connecté, et assure l'intégrité des données transférées.

Pour être identifiée dans le réseau, une machine doit avoir une adresse unique. Cette adresse pour le protocole IP est sous forme de quatre octets séparés chacun par un point. Chacun de ces octets appartient à une classe selon l'étendue du réseau.

Nous allons voir maintenant quelque classe dans la bibliothèque java.net que nous allons utilisés :

3.1 La classe InetAdress

Un objet de la classe InetAdress représente une adresse Internet. Elle contient des méthodes pour lire une adresse. Cette classe ne possède pas de constructeur mais propose trois méthodes statiques :

Méthode	Rôle
InetAdress getByName(String)	Renvoi l'adresse internet associé au nom d'hote fourni en paramètre
InetAdress[] getAllByName(String)	Renvoie un tableau des adresses internet associées au nom d'hôte fourni en paramètre
InetAdress getLocalHost()	Renvoie l'adresse Internet de la machine locale
Byte[] getAdress()	Renvoie un tableau contenant les 4 octets de l'adresse internet
String getHostAdress()	Renvoie l'adresse Internet sous la forme d'une chaîne de caractères
String getHostName()	Renvoie le nom du serveur

Tableau 5.01 : liste des méthodes proposées par la classe InetAdress

3.2 Utilisation du protocole TCP

TCP est un protocole qui permet une connexion de type point à point entre deux applications. C'est un protocole fiable qui garantit la réception dans l'ordre d'envoi des données.

TCP utilise la notion de port pour permettre à plusieurs applications d'utiliser TCP. En mode TCP java propose deux sortes de classes :

3.2.1 La classe SocketServer.

Cette classe est utilisée sur le côté serveur : elle attend simplement les appels du ou des clients. C'est un objet de type Socket qui prend en charge la transmission des données. Un objet de cette classe est associé à un port sur lequel il va en attendre les connexions. La classe SocketServer possède plusieurs constructeurs dont les principaux sont :

Constructeur	Rôle
ServerSocket()	Constructeur par défaut
ServerSocket(int)	Créer une socket sur le port fourni en paramètre
ServerSocket(int,int)	Créer sur le port avec la taille maximale de la file fourni en paramètre

Tableau 5.02 : liste des constructeurs de la classe SocketServer

La classe SocketServer possède plusieurs méthodes :

Méthode	Rôle
Socket accept()	Attendre une nouvelle connexion
Void close()	Fermer la socket

Tableau 5.03 : liste des méthodes de la classe SocketServer.

Si un client tente de communiquer avec le serveur, la méthode accept() renvoie une socket qui encapsule la communication avec ce client.

3.2.2 La classe Socket

C'est la classe qui contient les méthodes de création des flux d'entrée et de sortie correspondantes. Les sockets constituent la base des communications par le réseau. Cette classe possède plusieurs constructeurs dont les principaux sont :

Constructeur	Rôle
Server()	Constructeur par défaut
ServerSocket(String, int)	Créer une socket sur la machine dont le nom et le port fournis en paramètre
ServerSocket(InetAdress, Int)	Créer une socket sur la machine dont l'adresse IP et le port sont fournis en paramètre

Tableau 5.04 : Liste des constructeurs de la classe Socket.

La classe Socket possède de nombreuses méthodes :

Méthode	Rôle
InetAdress getInetAddress()	Renvoie l'adresse IP à laquelle la socket est connectée
void close()	Ferme la socket
InputStream getInputStream()	Renvoie un flux en entrée pour recevoir les données de la socket
OutputStream getOutputStream()	Renvoie un flux en sortie pour émettre les données de la socket
int getPort()	Renvoie le port utilisé par le socket

Tableau 5.05 : Liste des méthodes de la classe Socket.

3.3 Réalisation du cryptosystème ou client.

Connaissant l'outil nécessaire, la mise en œuvre de la partie client se déroule comme suit :

- créer une instance de la classe Socket en précisant la machine et le port en paramètre,
- obtenir un flux en entrée et en sortie,
- écrire les traitements.

Le numéro de port que nous avons choisis est : 6789. Nous pouvons choisir un numéro de port supérieur à 1024 (voir ANNEXE 4), car les ports en dessous de cette valeur sont réservés à des applications précises. Nous tenons aussi à préciser que le client, quand il est connecté, écoute aussi le port et attend les messages qui lui sont destinés. Ce procédé est possible grâce à l'utilisation du concept miltithreading de java.

Le multithreading introduit la possibilité d'avoir plusieurs fils d'exécution concurrents dans un même programme. Dans le langage Java, ces fils d'exécution sont appelés threads.

Voici une structure générale d'un client capable d'attendre des réponses :

```
import java.net.* ;
import java.io.* ;
class Client
{
	static final int PORT = 6789; // numéro du port du serveur
	Socket socketClient; // declaration d'un socket
	PrintWriter canalEcriture; //
	Client (String NomHote)
	{
	// Initialisation des variables déclarer au dessus
	try
	{
		socketClient = new Socket(NomHote, PORT) ;// creation de la socket
		canalEcriture = new PrintWriter (socketClient.getOutputStream(), true);
	}
	catch(IOExeption e)
	{
		//traitement de exeption
	}
	New listener (socketClient) ; // lancement du processus d'attente
	}
	class listener extends Thread
	{
	public listener(Socket socket)
	{
		// lancement du Tread
	}
	public void run()
	{
		//traitements à faire au socket
	}
	}
}
```

Ceci n'est que la structure générale d'un client multithread.

Comme nous avons finis avec la partie réseau du client, nous allons maintenant voir la partie cryptosystème.

Pour programmer l'algorithme RSA, java nous offre deux méthodes différentes. Soit en le programmant directement en utilisant les formules qu'on a vu au chapitre 3, en utilisant le paquetage java.math.BigInteger, java.security, soit en utilisant le paquetage JCE (pour Java Cryptography Extensions). Ce dernier regroupe des algorithmes cryptographiques prêts à être employés.

3.3.1 Description de JCE :

Le paquetage JCE fait partie intégrante du JDK standard version 1.4, s'il a été fourni comme extension auparavant.

3.3.2 La classe Provider

Il est possible de choisir n'importe quel fournisseur de service de cryptographie. Mais nous nous sommes contentés du fournisseur par défaut du JDK.

3.3.3 La clé d'encryptage

Le paquetage JCE procure la gestion de clé. Une clé est spécifiée par l'interface Key. On confectionne une clé en utilisant un KeyGenerator ou KeyPairGenerator.

```
kGen = KeyGenerator.getInstance(algorithme) ;
Key sk = kGen.generateKey() ;
Ou
kGen = KeyPairGenerator.getInstance(algorithme) ;
KeyPair sk = kGen.generateKeyPair();
```

3.3.4 La classe Cipher

La classe javax.crypto.Cipher implémente le cryptage d'un texte. Un objet Cipher est obtenu par getInsance doit être initialisé dans un des deux modes (encodage ou décodage), qui sont définis comme des entiers constants finaux dans la classe Cipher. Les noms symboliques pour ces modes sont :

ENCRYPT_MODE, DECRYPT_MODE

Chaque méthode d'initialisation prend un paramètre mode (opmode) et initialise l'objet Cipher pour ce mode. D'autres paramètres incluent la clé (*key*), paramètres de l'algorithme (*params*) et source aléatoire (*random*).

Pour initialiser un objet Cipher, il faut que l'une des méthodes initialement proposées, soit par exemple :

public void init (int opmode, Key key) ;

Il faut noter que lorsqu'un objet *Cipher* est initialisé, il perd tous les états précédents. En d'autres termes, initialiser un objet *Cipher* équivaut à créer une nouvelle instance de *Cipher*.

Pour notre application, nous avons choisis la programmation directe par la formule. Voici un exemple simple de code qui nous a inspirés :

```
import java.math.BigInteger;
import java.security.SecureRandom;
class Rsa
{
        private BigInteger n, d, e;
        public Rsa(int bitlen)
        {
                SecureRandom r = new SecureRandom();
                BigInteger p = new BigInteger(bitlen / 2, 100, r);
                BigInteger q = new BigInteger(bitlen / 2, 100, r);
                n = p.multiply(q);
                BigInteger m = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
                e = new BigInteger("3");
                while(m.gcd(e).intValue() > 1) e = e.add(new BigInteger("2"));
                d = e.modInverse(m);
        }
        public BigInteger encrypt(BigInteger message)
        {
                return message.modPow(e, n);
        }
                public BigInteger decrypt(BigInteger message)
        {
                return message.modPow(d, n);
        }
```

```
}
```

3.4 Realisation du serveur.

Notre serveur est un serveur multithreading, c'est-à-dire capable de servir plusieurs clients en même temps. Voici la structure de base d'un serveur multithreading écrite en java :

```
class server extends Thread
{
        /* attributs */
        ServerSocket server ;
        …
        Server (int port)throws IOException
        {
                /* initialisation */
                …
                /* création du serveur */
                server=new ServerSocket(port);
…
                /* exécution de la méthode run() */
                this.start() ;
        }
        public void run()
        {
                while (true)
                {
                        /* attente d'une connexion */
                        Socket socket=server.accept();
                        …
                        /* création d'un objet de type connection à associer au client
                        connecté … */
                        new connection(socket);
                        …
                }
                server.close();
        }
//
        class connection extends Thread
        {
                /* attributs */
                Socket socket;
                …
                connection(Socket f_socket)throws IOException
                {
                        /* initialisation */
                        socket=f_socket;
                        …
                        /* exécution de la méthode run() de la classe */
                        this.start();
                }
```

```
            public void run()
            {…
            /* récupération des points d'accès aux canaux … */
                    while (true)
                    {
                            /* servir le client */
                            …
                    }
                    socket.close();
            }
    }
}
```

4. Présentation du cryptosystème Client/Serveur à base de RSA.

Dans ce paragraphe, nous allons présenter le fruit de notre étude.

4.1 La partie Client.

La partie client fonctionne en deux modes :

- En mode cryptosystème
- En mode client du serveur

4.1.1 Le mode cryptosystème simple.

En mode client, l'application se comporte comme un simple cryptosystème, permettant ainsi de chiffrer et déchiffrer des textes. La figure 5.01 montre l'interface utilisateur du cryptosystème.

Le client est ici représenté comme un petit bloc note de Windows, capable d'écrire, d'ouvrir, et de crypter un fichier texte. Essayons de crypter un texte enregistré dans le disque dur :

- Cliquer sur le bouton ouvrir un fichier : une boite de disloque s'ouvre, on a la figure 5.02
- Choisissez votre fichier

On a par exemple la figure 5.03

- Cliquer après sur le bouton chiffrer

On obtient alors un texte chiffrer. Plus précisément une suite de chiffre comme nous montre la figure 5.04.

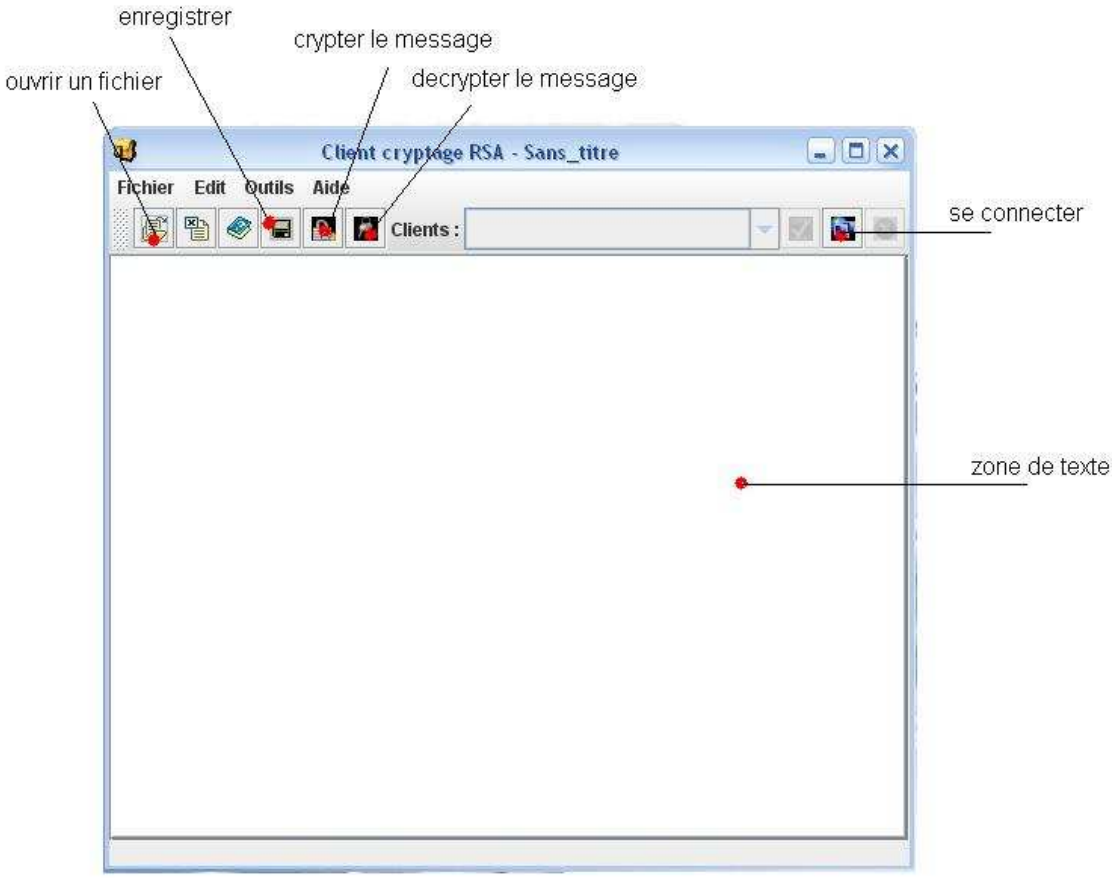

Figure 5.01 : interface utilisateur du cryptosystème

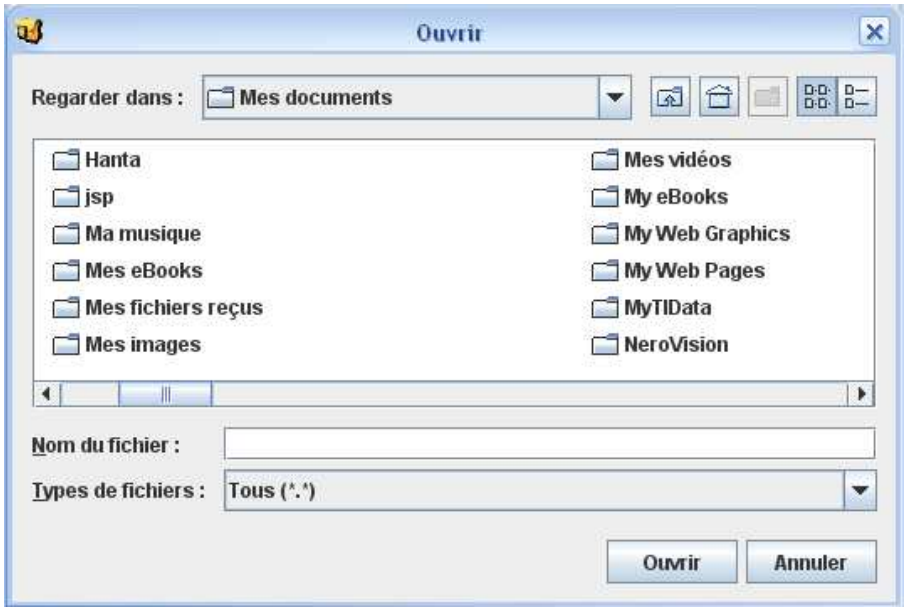

Figure 5.02 : Boite de dialogue ouvrir

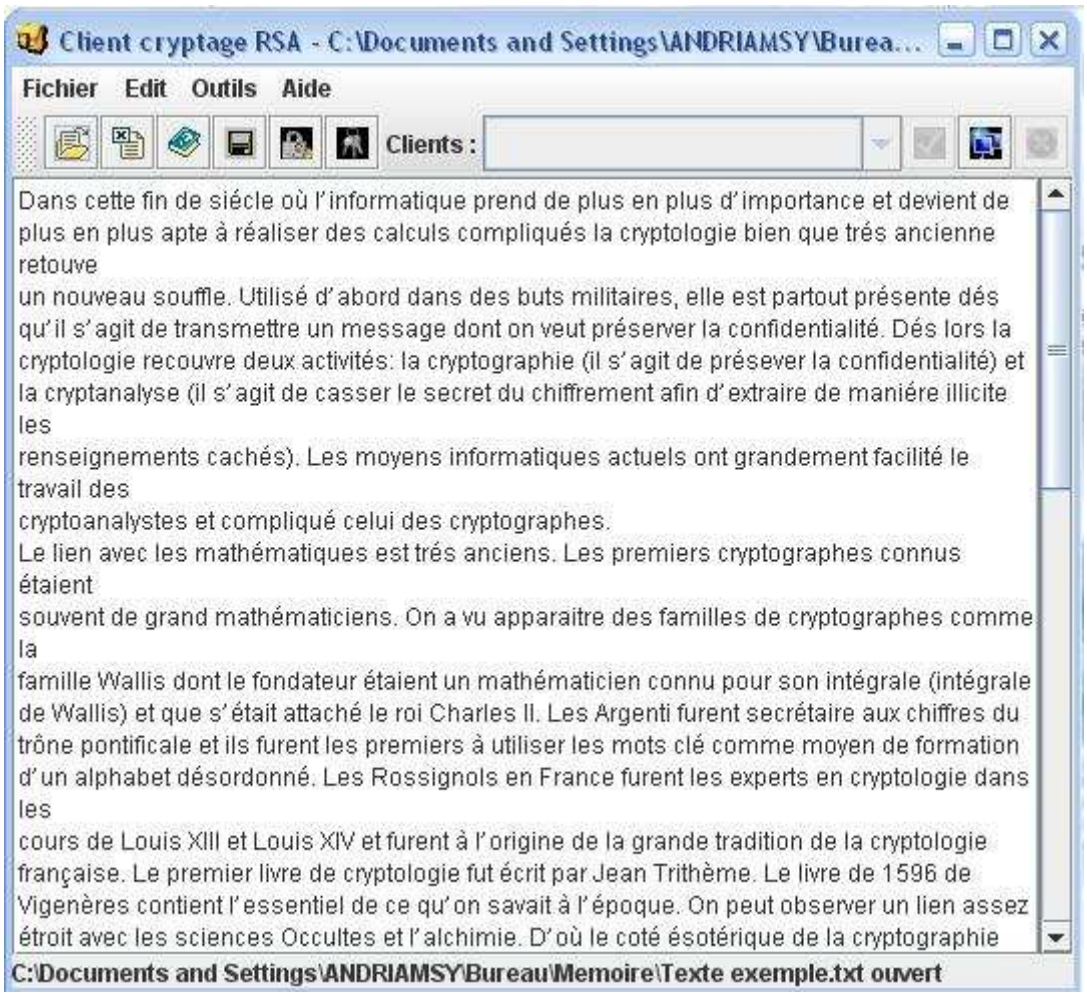

Figure 5.03 : exemple d'un texte clair

Remarque : lors de la première utilisation, un mot de passe de l'utilisateur sera demandé. Ceci a pour but de protéger la clé publique de l'utilisateur dans un fichier appelé « util.bin » qui est un fichier binaire. Après cela, l'utilisateur a besoin de ce mot de passe pour décrypter le texte.

4.1.2 Le mode client du serveur

Dans ce mode, le cryptosystème va se connecter au serveur ; le nom du serveur est alors demandé. Une boîte de dialogue comme la montre la figure 5.05. :
Vous tapez le nom du serveur et vous êtes connectés. Si votre serveur n'est pas disponible ou vous avez tapé un nom de serveur incorrect, la boîte de dialogue représentée sur la figure 5.06 s'affiche et votre connexion est refusée. :

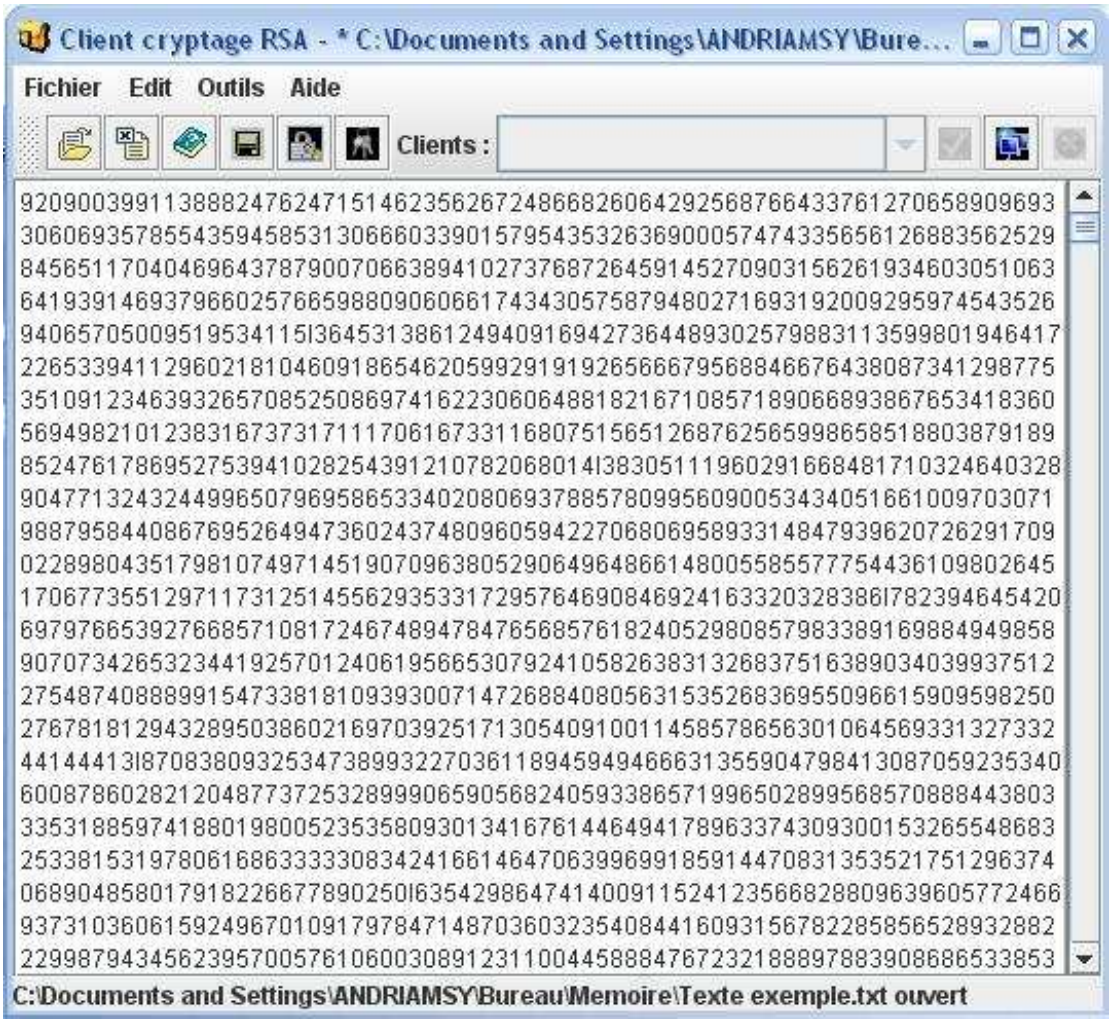

Figure 5.04 : exemple de texte chiffré avec le cryptosystème

Figure 5.05 : Boîte de dialogue demandant le nom de serveur.

Figure 5.06 : Boîte de dialogue indiquant que le serveur n'est pas disponible

Lorsque le client est connecté correctement, la liste des clients connectés (lui-même s'affiche sur cette liste) sont disponibles sur clients. On a alors la figure suivant :

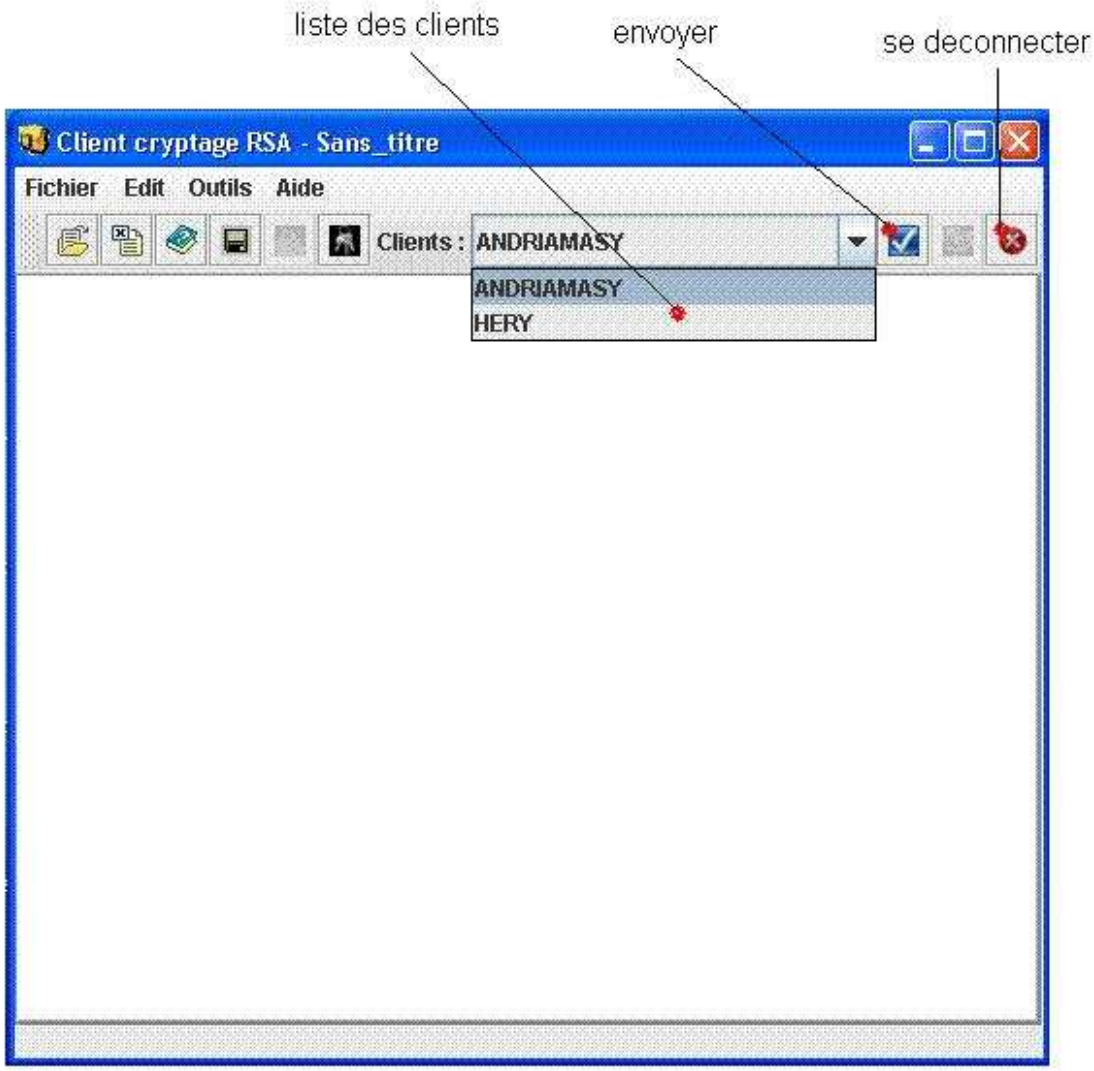

Figure 5.07 : interface d'un client connecté.

Quand le client est connecté au serveur, la fonction de chiffrement est enlevée. Ceci pour assurer une sécurité lors de l'utilisation sur réseau.
Quand le client à reçu un nouveau message, une boîte de dialogue lui averti.

Figure 5.08 : Alerte d'un nouveau message

Après avoir appuyé sur OK, le message crypté s'affiche et il peut le décrypter grâce à son mot de passe et sa clé privée. Le client peut envoyer un message à quelqu'un qui il veut en le choisissant sur la liste et d'appuyer sur le bouton envoyer

4.2 La partie serveur.

L'interface graphique du serveur est représentée sur la figure 5.08. :

Le serveur a pour rôle de gérer une petite base de données de type fichier. Le choix du fichier est ici justifié par la rapidité d'écriture et de lecture, et que la base de données à gérer n'est que de petite taille. La bande de données comporte trois champs :

- Le nom de la machine cliente
- Son adresse IP
- Et sa clé publique

Ces champs sont séparés par un séparateur « # ».

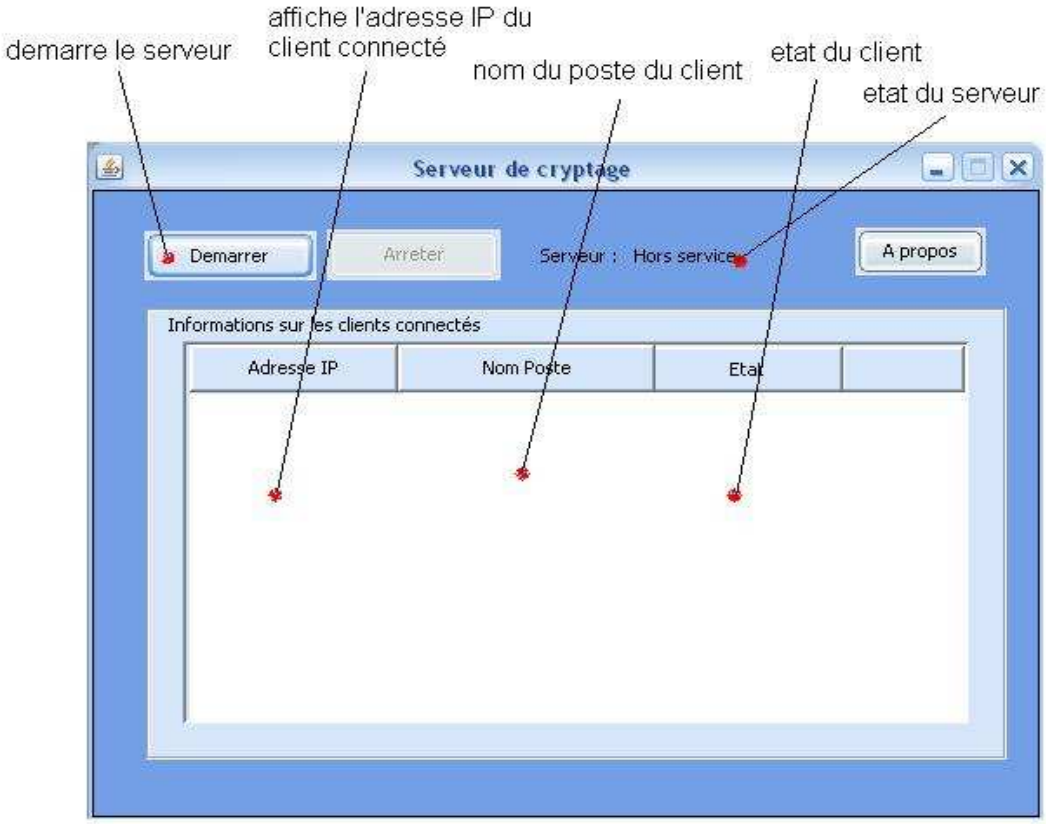

Figure 5.08 : Serveur de cryptage.

La figure 5.09 montre un exemple de ce fichier

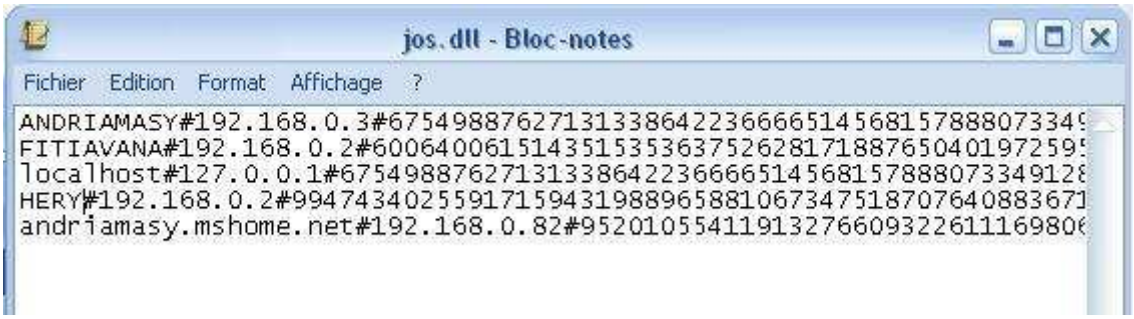

Figure 5.09 : Fichier base de données

Le serveur gère automatiquement cette base de données. Il crée à sa première utilisation, et teste si un client est déjà inscrit ou non.

Le vrai rôle du serveur est de répondre à la requête envoyée par un client (requête d'une clé publique) et de commuter le message vers la destination finale.

Remarque : Avant de clore cette présentation, nous aimerions laisser ces quelques remarques ;

- Comme l'algorithme de chiffrement que nous avons choisis est un algorithme à clé publique, le chiffrement ou le déchiffrement sera un peu long pour un texte assez long.
- L'utilisateur de ce système doit choisir un texte convenable pour être chiffré. Ceci afin d'éviter d'attendre longtemps.
- Comme nous avons utilisés java, ce système peut s'exécuter sous plusieurs plateformes comme Windows, Linux, Mac-Os ou Solaris.

CONCLUSION GENERALE

La cryptographie est l'un des moyens le plus sûr pour assurer la sécurité d'un réseau. Mais les algorithmes cryptographiques sont aussi limités par le temps. Un système utilisant le DES paraît incassable à son age, mais ce n'est plus qu'un jeu d'enfant de nous jours. De même, pour d'autres systèmes, l'avancée technologique devient un très grand souci. D'où la nécessité d'évoluer chaque jour.

Du chiffrement à clé privée, qui a le plus grand problème sur l'acheminement de la clé, mais un chiffrement très rapide, au chiffrement à clé publique, caractérisé par la résolution de la distribution de la clé, mais qui présente l'inconvénient sur sa lenteur. La cryptographie a su assurer la sécurité de divers système dans plusieurs domaines d'activités. Il est à noter que le chiffrement à clé privée est basé sur l'utilisation d'une seule clé pour le chiffrement et le déchiffrement. Et le chiffrement à clé publique repose sur l'utilisation d'un couple de clé l'une dit clé publique et sert pour le chiffrement, l'autre dit clé privée sert pour le décryptage.

Mais le temps est révolu, la tendance est actuellement sur l'utilisation d'un système hybride qui utilise en même temps les deux types de chiffrement. Souvent, le chiffrement à clé privé sert à chiffrer le message, dans le but de gagner un peu de temps. Et le chiffrement à clé publique sert à chiffrer la clé privée utilisée pour le chiffrement. C'est le cas de PGP qui est un algorithme hybride.

Néanmoins, RSA a survécu pendant des années de cryptanalyse. Ceci est dû à sa flexibilité d'évolution. Un algorithme dure plus longtemps s'il s'adapte à l'évolution. C'est le cas de RSA. Conçu au départ pour une utilisation avec une longueur de clé à 128 bits, il s'élève actuellement jusqu'à 2048 bits pour une sécurité parfaite. L'algorithme DES n'a pas duré aussi longtemps que RSA. Ceci est bien évidemment dû à la longueur de la clé à 64 bits.

L'implémentation de la cryptographie dans les anciens réseaux téléphoniques comme le RTC est encore très difficile, mais les nouvelles générations des réseaux téléphoniques intègrent bien dans leur système un système de chiffrement très puissant utilisant le chiffrement par flot. Ceci pour mieux sécuriser les informations sensibles, comme la voix, les données, dans ces réseaux.

La cryptographie devient un outil indispensable pour une sécurisation assurée. Même dans la vie quotidienne certain logiciel propose le chiffrement des emails avant son envoi à quelqu'un.

Pour concrétiser notre étude, nous avons mis au point un système de chiffrement Client/Serveur. Afin de faciliter la sécurité des textes de grandes importances. Bien sûr, ce système n'est pas parfait. Il présente ainsi tous les inconvénients du chiffrement à clé publique et en même temps la lenteur de java. Néanmoins, il présente une sécurité pour un réseau local à Madagascar.

Une évolution de ce système est envisageable comme l'utilisation d'un système de chiffrement hybride, mais nous laissons cela à ceux qui veulent bien poursuivre nos travaux de recherche.

Ce mémoire nous a aidés à comprendre beaucoup de choses, notamment sur le développement d'un système Client/Serveur en java, que sur les différentes possibilités offertes par la cryptographie. Ce n'est qu'un début à nos recherches, et nous espérons que nous aurons encore la possibilité de franchir d'autres étapes plus importantes.

La plus grande difficulté que nous avons eue pendant la réalisation de ce système se trouvait au niveau de la programmation, car nous avions dû recourir à l'étude approfondie de certaine bibliothèque du langage java.

ANNEXES

– ANNEXE 1 : FONCTIONS DE HACHAGE A SENS UNIQUE

- Une fonction de hachage est une fonction qui convertit une chaîne de longueur quelconque en une chaîne de taille inférieur et généralement fixe. La chaîne résultante est appelée *empreint* ou condensé de la chaîne initiale
- Une fonction à sens unique est une fonction facile à calculer mais difficile à inverser. La cryptographie à clé publique repose sur l'utilisation de fonctions à sens unique à brèche secrète : pour qui connaît le secret, la fonction devient facile à inverser.
- Une fonction de hachage à sens unique est une fonction de hachage qui est en plus une fonction à sens unique : il est aisé de calculer l'empreinte d'une chaîne donnée, mais il est difficile d'engendrer des chaînes qui ont une empreinte donnée, et donc de déduire la chaîne initiale à partir de l'empreinte.

On demande généralement en plus à une telle fonction d'être sans collision, c'est-à-dire qu'il soit impossible de trouver deux messages ayant la même empreinte. On utilise souvent le terme fonction de hachage pour désigner, en fait, une fonction de hachage à sens unique sans collision.

La plus part des fonction de hachage à sens unique sans collision sont construites par itération de compression : le message *M* est décomposé de *n* blocs $m_1,\ldots,m_n$, puis une fonction de compression f est appliquée à chaque bloc et au résultat de la compression du bloc précédent ; l'empreinte *h(M)* est le résultat de la dernière compression.

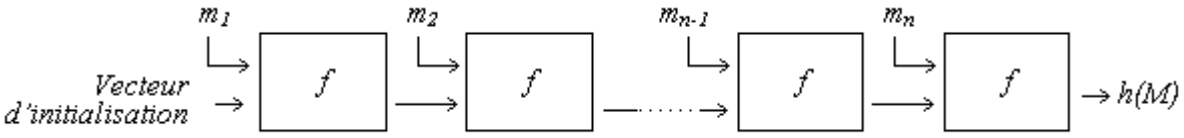

Figure 6.01: Fonction de hachage itérative

Voici des exemples de fonctions ainsi conçues

- MD5 (Message Digest 5)

Développé par Rivest en 1991, MD5 produit une empreint de 128 bits à partir d'un texte de taille arbitraire. MD5 manipule le texte d'entrée par blocs de 512 bits

- SHA (Secure Hash Algorithm)

SHA est la fonction de hachage utilise par SHS (Secure Hash Standard), la norme du gouvernement Américain pour le hachage. SHA-1 est une amélioration de SHA publiée en 1994. SHA-1 produit une empreinte de 160 bits à partir d'un message de longueur maximale 2^{64} bits. Tout comme MD5, SHA-1 travaille sur des blocs de 512 bits.

- RIPE-MD

Développée dans le cadre du projet RIPE (pour RACE Integity Primitives Evaluation) de la communauté Européenne, RIPE-MD fournit une empreinte de 128 bits. RIPE-MD-160 est une version renforcée de RIPE-MD qui fournit une empreinte de 160 bits.

– ANNEXE 2 : SIGNATURE NUMERIQUE.

La norme ISO 7498-2 définit la signature numérique comme des "données ajoutées à une unité de données, ou transformation cryptographique d'une unité de données, permettant à un destinataire de prouver la source et l'intégrité de l'unité de données et protégeant contre la contrefaçon (par le destinataire, par exemple)". La mention "protégeant contre la contrefaçon" implique que seul l'expéditeur doit être capable de générer la signature.

Une signature numérique fournit donc les services d'authentification de l'origine des données, d'intégrité des données et de non-répudiation. Ce dernier point la différencie des *codes d'authentification de message* (voir paragraphe suivant), et a pour conséquence que la plupart des algorithmes de signature utilisent la cryptographie à clef publique.

Sur le plan conceptuel, la façon la plus simple de signer un message consiste à chiffrer celui-ci à l'aide d'une clef privée : seul le possesseur de cette clef est capable de générer la signature, mais toute personne ayant accès à la clef publique correspondante peut la vérifier. Dans la pratique, cette méthode est peu utilisée du fait de sa lenteur.

Une autre méthode utilisée pour signer consiste à calculer une empreinte du message à signer et à ne chiffrer que cette empreinte. Le calcul d'une empreinte par application d'une fonction de hachage étant rapide et la quantité de données à chiffrer étant fortement réduit, cette solution est bien plus rapide que la précédente.

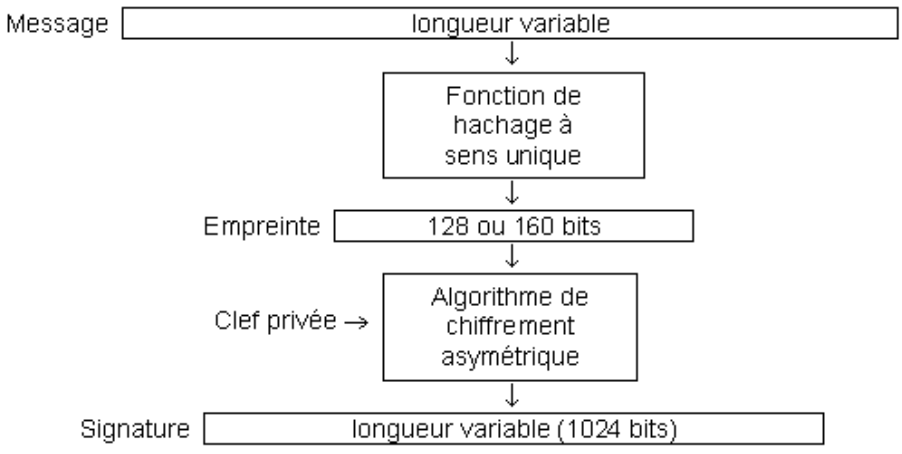

Figure 6.02: Signature

Voici quelques exemples de signature numérique :

- DSS

Proposé par le NIST en 1991, puis finalement adopté en 1994, DSS (*Digital Signature Standard*) est la norme de signature numérique du gouvernement Américain. Elle se base sur l'algorithme DSA (*Digital Signature Algorithm*), qui utilise SHA comme fonction de hachage à sens unique et El Gamal pour la génération et la vérification de la signature.

- RSA

Si DSS est la norme officielle aux U.S.A., RSA est en revanche une norme de fait, qui est en pratique beaucoup plus utilisé pour la signature que DSA.

– **ANNEXE 3 : Liste des numéros de port TCP et UDP.**

Le tableau ci-dessous affiche la liste des numéros de ports et les applications qui l'utilise en utilisant TCP ou UDP.

Numéro de port	Service qui l'utilise
0	Réservé
1	TCP port Service Multiplexage
2	Utilitaire de gestion (*)
3	Processus de compression
5	Travail distant
7	Echo
9	Discard (annuler)
11	Utilisateurs actifs
13	Daytime
17	QUOTD (Quote of the day)
20	Port de donnée FTP (*)
21	Port de contrôle FTP (*)
23	TELNET (*)
25	Simple Mail Transfer Protocol (SMTP) (*)
35	Serveur d'impression privé
37	Time
39	Ressource Location Protocol
42	Name server
43	Nicname
49	Login Host protocol
52	XNS Time Protocol
53	Domain Name Server
54	XNS clearing House
66	Oracle SQL*NET
67	Bootstrap Protocol Server
68	Bootstrap Protocol Client
69	Trivial File Transfert Protocol (TFTP°) (**)

70	Gopher protocol (**)
79	Finger Protocol
80	http
88	Kerberos ou port web de rechange
94	Trivoli Obkect Dispatcher
95	SUPDUP
102	ISO-TSAP
107	Remote telnet Service
108	SNA Gateway Access Server
109	Post Office Protocol version 2 (POP2)
110	POP3
111	Sun remote Procedure Call
119	Network News Transfer Protocol
123	Network Time Protocol
134	INGRES-NET Service
137	NETBIOS Naming Service
138	NETBIOS Datagram Serice
139	NETBIOS Session Service
142	Britton-Lee IDM
161	SNMP (**)
162	SNMP traps (**)
191	Prospero
194	Internet Relay Chat
201	Apple Talk Routing Maintenance
202	Apple Talk Name Binding
213	IPX
215	Insignia (Soft PC)
256	SNMP Checkpoint
513	Renseigne une base de donnée sur la charge des ordinateurs(**)
525	Time Server
533	Netvall
556	RFS Server
565	Who Am I

749	Kerberos Administration
767	Phone
1025	Network Blackjack
3306	mysql
6000	xwindows
7000-7009	Utilisé par Andrew File System
8080	Web Apache
17007	ISODE Directory User Agent

(*) : Service réservé au protocole TCP

(**) : Service réservé au protocole UDP

– ANNEXE 4 : ETUDE COMPARATIF DES DIFFERENTES ALGORITHMES DE CHIFFREMENT

Lorsqu'on veut implémenter un système de chiffrement dans un réseau le choix de l'algorithme de chiffrement est très difficile. Des systèmes cryptographiques de toute sorte sont utilisés de façons courantes. Tous ne nécessitent pas le même niveau de sécurité et les systèmes cryptographiques utilisés varient beaucoup en complexité. Dans certains cas le chiffrement est très simple, dans d'autres on cherche à assurer la meilleure sécurité disponible. On comprend donc que dans certains cas les impératifs de facilité et de rapidité de chiffrement l'emporte sur l'assurance d'une sécurité absolue. Voici un petit aperçu qui va aider à choisir un algorithme de chiffrement

Cryptographie à clé privée ou cryptographie à clé publique ?

Cette question qui se montre comme un grand sujet de débat quand on parle de cryptographie n'est qu'une question d'incompréhension. Les deux cryptographies sont deux choses différentes et qui à pour but différents.

La cryptographie à clé privée est meilleure quand on parle de chiffrer des données. Ceci s'explique par le fait qu'elle est infiniment plus rapide et n'est pas prédisposée à des attaques à texte chiffré choisi. Pour en choisir sur ces algorithmes il faut d'en rendre compte de certains caractères :

- longueur du bloc de chiffrement
- longueur de la clé
- les techniques de chiffrement utilisées

La cryptographie à clé publique quand à lui peut faire les autres choses que la cryptographie à clé privée trouve ces limites. En effet elle est surtout utilisée pour la gestion des clés et des protocoles. Donc ici on ne parle plus de rapidité mais plutôt de sécurité parfaite. Pour en choisir aussi sur l'un de ces algorithmes il faut bien connaître :

- la nature des données à chiffrer
- la longueur de la clé adéquate

Remarque : Il est ici très important de connaître que le chiffrement des données par les algorithmes à clé privée s'effectue 1000 fois plus rapide qu'un chiffrement à clé publique.

– ANNEXE 5 : LES CHIFFREMENTS PAR SUBSTITUTIONS

- Substitution mono alphabétique : dans ce type de substitution ou encore substitution simple, chaque lettre est remplacé par un autre lettre ou un autre symbole dans le message caché.

Exemple :

La sténographie, l'alphabet désordonné, alphabet renversé, le chiffre d'Atbash,…

- Substitution poly alphabétique : la substitutions poly-alphabétique ou encore appelé à double clé ou à alphabet multiple, utilise plusieurs alphabet. Ce qui signifie qu'un même lettre peut être remplacé par plusieurs symboles. Elle consiste à utiliser une suite de chiffre mono alphabétique réutilisé périodiquement.

Exemple :

Le chiffre de Bellasso, le chiffre de Porta, le chiffre de Vigenere, le chiffre de beaufort,…

- Substitution polygrammique : Dans ce type de substitution les lettres ne sont pas chiffrés séparément mais par groupe de plusieurs lettres (en général deux ou trois lettres).

Exemple

Le chiffre de Playfair, le chiffre de Slidefais, le chiffrement à deux carrées,…

- Substitution homophonique : La substitution homophonique ou encore substitution à représentation multiple permet de faire à chaque lettre du message en clair un ensemble possible d'autre caractère. Elle consiste à remplacer une lettre non pas par un symbole unique mais par un symbole choisie au hasard parmi plusieurs.

Exemple :

Représentation multiple de « e », le disque de l'armée Mexicaine

BIBLIOGRAPHIE

[1] F. Arnault, *Théorie des nombres et cryptographie*, Cours de DEA, Université de Limoges, mai 2002.

[2] J. Razakarivony, *Cryptographie et sécurité réseau*, Cours 5me année, Dép. Tél.-E.S.P.A., A.U. : 2005-2006

[3] F. Bergeron, A. Goupil, *La cryptographie de l'Antiquité à l'Internet*, Cours DEA, Dép. de Mathématique, Université du Québec à Montréal, 24 février 2006.

[4] M..E. H A. Aziz, *les principaux algorithmes de la cryptographie*, Rapport d'exposé, Faculté des sciences Unice : Centre Universitaire d'informatique, 4 janvier 2005.

[5] M. Videau, *Critères de sécurités des algorithmes de chiffrement à clé secrète*, Thèse doctoral, Université Paris 6, 10 nov. 2005

[6] G. Labouret, *Introduction à la cryptographie*, cours, 5 nov. 1998

[7] C. Bassil, *SVSP (Secure Voice over IP Simple Protocol) : Une solution pour la sécurisation de la voix sur IP*, Thèse doctoral, TELECOM PARIS : E.N.S.T. : Informatique et réseaux, 12 déc. 2005.

[8] P. Zimmermann, *Introduction à la cryptographie*, Network Associate Inc, 1998

[10] L.E. Randriarijaona, *Réseau TCP/IP*, cours 4me année, Dép. Tél.-E.S.P.A., A.U. : 2004-2005

[11] P. Nicolas, *Cours de réseaux*, Maîtrise d'informatique : Université D'Angers, A.U. : 1999-2000.

[12] M.A. Rakotomalala, *réseau des télécommunications*, cours 5me année, Dép. Tél.-E.S.P.A., A.U. : 2005-2006.

[13] M.A. Rakotomalala, *G.S.M*, cours 4me année, Dép. Tél.-E.S.P.A., A.U. : 2004-2005.

[14] P. Wassef, *Arithmétique*, cours, Dép Mathématique, Université Pierre & Marie Curie, A.U. :2005-2006.

[15] F. L. Jacomo, *Cours Arithmétique*, Stage olympique de Saint-Malo, 1r Août 2003.

[16] L. E. Randriarijaona, *Programmation avancée*, cours 5me année, Dép. Tél.-E.S.P.A., A.U. : 2005-2006.

[17] J. Brondeau, *Introduction à la programmation Java*, DUNOD : Paris 1999

[18] J.M. Doudoux, *Développons en java*, version 0.85 : 2006.

[19] http://java.sun.com/j2se/1.5.0/download.jsp

[20] Z ANDRIAMIASY, *téléphonie avancée*, cours 5me année, Dép. Tél-E.S.P.A., A.U : 2005-2006

[21] B. Barraque, *avantages de l'UMTS et l'état actuel de son déploiement dans le monde, cours*, Université Claude Bernard, A.U 2004-2005

- *Nom* : **ANDRIAMASY**
- *Prénoms* : Ramanamihanta Velontsihoarana Joseph
- *Adresse* : Lot B16 AMBONISOA ANDRANOVELONA ILAFY
  ANTANANARIVO 103
  Tél : 0330300469
  E-mail : stjoveve@yahoo.fr

- *Titre du mémoire* : **CONCEPTION D'UN CRYPTOSYSTEME CLIENT SERVEUR BASEE SUR RSA**
- *Nombres de pages* : 87
- *Nombres de Tableaux* : 7
- *Nombres de Figure* : 29

- *Mots clé* : Cryptographie, Chiffrement, Déchiffrement, Cryptologie, Cryptanalyse, Cryptosystème, Réseau, Attaque, Sécurité

- *Directeur de mémoire* ; **RAZAKARIVONY** Jules

RESUME

Le monde évolue, le temps change, beaucoup de choses ne resteront pas comme elles étaient. Les progrès technologiques ont apportés des évolutions, mais aussi des méfaits comme l'insécurité. Et qui est la base de ce présent mémoire.

La cryptographie est l'une des meilleurs moyens d'assurer la sécurité des informations. Nous avons élaborés durant ce rapport l'étude des algorithmes cryptographiques, ainsi que leurs implémentations dans certains réseaux de télécommunications.

Pour notre part nous avons proposés un système (client/ serveur) permettant de sécuriser un simple réseau local. Nous avons ainsi lié les études théoriques à la réalisation pratique.

ABSTRACT

The world evolves time changes. Many things will not remain as they were. Technological progress brought evolutions, but also misdeeds like the insecurity, which is the base of this present report.

Cryptography is one of the best ways of ensuring the safety of informations. We prepared during this report the study of the cryptographic algorithms, like their implementations in certain telecommunications networks.

For our part, we proposed a system (Client / Server) allowing to make safe a simple local area network. Thus, we showed the theorical studies with the practical realization.