N° d'ordre : 23/2011-M/EL

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE
FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

MEMOIRE

Présenté pour l'obtention du diplôme de MAGISTER

En : ELECTRONIQUE

Spécialité : Systèmes Radiofréquences et Micro-ondes

Par : Mr BOUFLIH Yacine

Sujet :

TRANSMISSION SÉCURISÉE PAR LE SYSTÉME PGP

Soutenu publiquement le 07/12/2011, devant le jury composé de :

Mme. R. Touhami	Professeur,	à l'USTHB	Présidente
M. S. Chitroub	Professeur,	à l'USTHB	Directeur de Mémoire
M. S. Mekaoui	Maître de Conférences/A,	à l'USTHB	Examinateur
M. A. Abdelli	Maître de Conférences/A,	à l'USTHB	Examinateur
M. D. Bahloul	Maître de Conférences/A,	à l'USTHB	Examinateur

بسم الله الرحمان الرحيم

Remerciements

Je Remercie Dieu de m'avoir donné la santé, la volonté et le courage qui m' ont permis de réaliser ce travail.

Aussi en terme de ce travail, je tiens à exprimer mes plus sincères remerciements et l'expression de ma profonde gratitude à mon promoteur Mr: Chitroub Salim pour m' avoir accepter dans le laboratoire de recherches et du traitement de signal, pour avoir dirigé ce travail avec compétence et professionnalisme, pour tout son temps précieux qu'il m' a consacré pour la réussite de ce travail, pour sa patience et son soutient moral, pour ses encouragements qui ont été pour moi une source de motivation.

Je remercions aussi les membres du jury qui ont acceptés de juger ce travail.

Bien entendu, Je tiens à remercier mes parents, pour leurs sacrifices et leur patience, tout au long de mes études.

Je remercie toutes les personnes qui m' ont aidés de près ou de loin pour la réalisation de ce travail.

Enfin, je remercie tout particulièrement ceux que je les ai oubliés dans ces remerciements, je les priant de bien vouloir m'excusé pour cet oubli.

Dédicace

Je dédie ce modeste travail

En premier lieu à mes chers Parents pour leurs encouragement leur soutien moral,

A mes frères : SOUFIANE, FARES, RIDHA

A ma unique et chère Sœur : FOUZIA

A ma chère fiancé : HOUDA

A toute ma famille qui se trouve à EL MAIN et au Bordj-Bou Arreridj

A tout mes amis du loin, ou du proche

TABLE DES MATIERES

CHAPITRE 3 : Système de Transmission Sécurisée à base de PGP

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS

PGP	Pretty Good Privacy
IDEA	International Data Encryption Algorithm
RSA	Riverst, Shamir and Adleman
MD5	Message-Digest Algorithm Version 5
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
SHA	Secure Hash Algorithm
DES	Data Encryption Standard
AES	Advanced Encryption System
RFID	Radio Frencency IDentification
GSM	Global System for Mobile communications
GPS	Global Positioning System
UMTS	Universal Mobile Telecommunications Systems
CRT	Chinese Remainder Theorem
S / MIME	Secure / Multipurpose Internet Mail Extensions
TPM	Trusted Platform Module
GPRS	General Packet Radio Service
M2M	Machine to Machine
CD	Compact Disc
DVD	Digital Versalite Disc
NIST	National Institute of Standards and Technology
MIT	Massachussets Institute of Technology

Introduction Générale

Introduction Générale

Depuis l'apparition de l'écriture, la communication entre les hommes a pris une nouvelle forme, simplifiant la persistance des connaissances et les échanges. Plus tard, au fil de l'évolution humaine, l'écriture a pris de plus en plus d'importance et elle est devenue vitale, par exemple en cas de guerre. De tout temps, les chefs de guerre et diplomates avaient besoin de transmettre des données et des informations stratégiques qui sont très privées et très confidentielles à leurs troupes ou à leurs alliés. Cependant, il suffisait à l'ennemi d'intercepter le message pour obtenir un avantage déterminant ce qui met les émetteurs en véritable danger. D'où, la transmission des telles données doivent se faire en toutes sécurité avec d'autres procèdes. Aujourd'hui, et au fur et à mesure que de nouvelles applications apparaissaient, la transmission sécurisée est une nécessité dans plusieurs domaines autres que militaire tels que : télécommunications, réseaux informatiques, télémédecine, …etc.

Le développement du réseau Internet, par exemple, a donné naissance à plusieurs nouveaux champs d'applications à l'occurrence des systèmes de communication professionnelle utilisés dans les grandes entreprises afin interagir efficacement avec leurs clients et leurs partenaires commerciaux. Les réseaux de télécommunication sans fil et la messagerie électronique nécessitent aussi un niveau de confidentialité important, en particulier s'il s'agit de la messagerie électronique professionnelle. Toutes ces exigences de ces nouvelles applications ont incite les chercheurs de développer des nouveaux systèmes de sécurisation de l'information beaucoup plus robustes et meilleurs parmi ceux qu'ils existent déjà. C'est dans ce contexte que s'inscrit le travail développé dans ce mémoire de Magister.

Ainsi, dans ce mémoire nous avons développé un outil très performant de transmission sécurisée en utilisant le système PGP (*Pretty Good Privacy*). Les données transmissent peuvent être des messages des courrières électroniques, des fichiers à archiver, ou tout autre document qu'on souhaite garantir le contenu. Ce système développé fait appel à des mathématiques de la cryptographie moderne. Son efficacité réside dans le fait qu'il utilise à la fois des algorithmes symétriques, tel que l'algorithme IDEA «*International Data Encryption Algorithm*» qui dispose d'une clé secrète de taille plus grande, et les algorithmes asymétriques, tel que l'algorithme RSA «*Riverst, Shamir and Adleman*» qui utilise des factorisations des nombres primaires de plus de 512 bits. En plus, il est basé sur les procédés de vérification du contenu et d'origine des

informations transmises. Ces procédés sont conceptualisés en faisant appel aux concepts de la signature numérique, pour assurer l'identité de l'émetteur, et à la fonction de hachage MD5, pour assurer le contenu des informations transmises. Ainsi, la combinaison de ces quatre algorithmes, IDEA, RSA, signature numérique par RSA et MD5, nous permet de former un système cryptographique complexe et puissant.

Le présent mémoire est constitué de trois chapitres. Dans le premier chapitre, nous allons parler de la transmission sécurisée, de différents canaux de transmission, de supports de transmission et des problèmes de transmission. Un cas particulier très important et d'actualité de la transmission sécurisée sera aussi présenté, à savoir la transmission sécurisée inter-machines M2M (*Machine To Machine*). Dans le deuxième chapitre, nous allons exposer les algorithmes de la cryptographie utilisés en transmission sécurisée tout en ressortant les incontinents et les avantages des chacune. Le chapitre trois sera consacré à présenter en détail notre système développé à base de PGP. Le fonctionnement de ce système sera alors détaillé afin de mettre en évidence ces points forts en termes de sécurité et résistance aux différentes attaques. Les résultats de simulation obtenus seront commentés et évalués aussi dans ce chapitre. Une conclusion, dans laquelle les perspectives envisagées seront exprimées, est donnée à la fin de ce mémoire.

CHAPITRE 1

Transmission Sécurisée : Problèmes et Solutions

1. Introduction

Un réseau suppose plusieurs équipements (ordinateurs, téléviseurs, divers équipements électroniques, téléphones, …) situés à distance les uns des autres. La première chose à mettre en œuvre pour constituer le réseau est la transmission des informations d'un équipement à l'autre, cette tache est réalisée à l'aide d'un system de télécommunication, dont son rôle est de transmettre à distance des informations d'un émetteur à un, ou plusieurs récepteurs à travers d'un canal de manière fiable et à coût réduit. Cependant, le type et la valeur de certaines informations à transmettre sont secrètes, confidentielles, ou des informations de type commerciale comme le cas par exemple des chaines de télévision payantes. Alors une interception de telles informations par un récepteur non légitime peut entraîner des pertes financières, avoir des conséquences d'ordre juridique. Pour répondre à ce problème, une solution consiste à mettre en œuvre un moyen de sécuriser les informations secrètes contre les menaces accidentelles ou intentionnelles. C'est ce qu'on appelle la transmission sécurisée.

2. Sécurité de l'Information dans la Pratique

L'information est un élément constitutif et déterminant dans tous les domaines. Tout au long de l'histoire, l'humanité a essayé d'envoyer des informations d'une façon sécurisée. La dissimulation d'information a été utilisée comme instrument de sécurisation pour les stratégies militaires et échange de données secrètes. Le transfert sécurisé de l'information est devenu une nécessité dans plusieurs domaines autre que le militaire. Il y a une variété importante de domaines d'applications de la transmission sécurisée. A cause de la révolution d'internet et l'utilisation de plus en plus massive d'informations sous forme numérique facilitent les communications et rendent de ce fait plus fragiles les informations que l'on détient. En effet, les réseaux ouverts créent des brèches de sécurité et il est plus aisé à un adversaire d'accéder aux informations. Notamment dans le cas de la messagerie électronique où les courriers transmis sont l'apportée des communicants non légitimes. Donc ces besoins font appel à la cryptographie. Cette dernière désigne l'art de chiffrer une information de façon à ce que seuls les initiés y aient accès. Le chiffrement consiste à transformer un texte en clair en un texte chiffré. Le déchiffrement, réciproquement, c'est traduire un texte chiffré en clair en connaissant la clé de chiffrement. La cryptologie est la science du secret, et regroupe deux branches : la cryptographie, qui permet de chiffrer les messages, et la **c**ryptanalyse, qui permet de les pirater**.**

2. 1. Systèmes développés pour la transmission sécurisée de l'information

L'être humain n'a évidemment pas attendu l'avènement d'internet pour vouloir cacher des messages. Le code de César qui consiste à substituer des lettres issue d'un décalage – le A devient D, le B devient E, etc. a été utilisé dans les correspondances des empereurs romains. Il s'agit de la première méthode de chiffrement. Le principe est de placer dans un tableau, les lettres de l'alphabet dans le message par deux chiffres (les numéros de colonne et de ligne du tableau) [1].

Au cours des siècles, d'innombrables solutions de chiffrement sont apparues, la plupart par substitution mono-alphabétique de lettres. La première et la seconde Guerres mondiales vont consacrer la cryptographie comme corollaire majeur de la communication militaire. Les services du chiffre des armées se livrent une bataille à distance pour, d'une part, mettre au point des systèmes de cryptographie des communications radio inviolables et, de l'autre, casser le code utilisé par les ennemis (la cryptanalyse). L'une des machines les plus célèbres de la seconde guerre mondiale se nomme Enigma; utilisée par les Allemands. Elle fonctionnait par substitution de lettres (avec changement à chaque lettre). En Grande-Bretagne, le mathématicien Alan Türing (inventeur, aussi, d'un précurseur de l'ordinateur personnel) passera la guerre aux côtés de milliers de savants de tous horizons à venir à bout des différentes versions d'Enigma.

L'arrivée de l'informatique et de sa formidable capacité de calcul va permettre la mise au point de systèmes de cryptage beaucoup plus puissants. Deux grands procédés ont vu le jour : à clé privée et à clé publique. Le premier est notamment symbolisé par le DES (*Data Encryption Standard*), adopté par la NSA (l'agence de sécurité nationale américaine) en 1976. Le DES fonctionne avec une clé de 64 bits, dont 56 sont utilisés pour le cryptage. Longtemps inviolable, le DES n'a pas résisté à l'augmentation de la puissance des ordinateurs. Lui succède, en 1998, l'AES (*Advanced Encryption Standard*) et sa clé de 128 bits, qui reste au delà de la portée de la puissance de calcul actuelle. Reste à savoir pour combien de temps [2]. D'ailleurs, ainsi que l'a démontré Claude Shannon en 1949 [3], il n'existe qu'un seul système à clé privée parfaitement sûr. Il a été mis au point par Gilbert Vernam en 1917[4], et consiste simplement en un carré de Vigenère dont la clé est de même taille que le message, choisie parfaitement aléatoirement et à usage unique.

Le principal problème du cryptage à clé privée consiste à trouver un canal parfaitement sûr pour transmettre la clé. En 1977, trois mathématiciens (Rivest, Shamir et Adleman) mettent au point l'algorithme RSA, qui fonctionne sur le principe de la clé publique (ou chiffrement asymétrique), en

application des travaux de Diffie et Hellman [5]. Chaque utilisateur dispose de deux clés, l'une publique et l'autre privée. Si Alice veut envoyer un message à Bob, elle utilise la clé publique de ce dernier. Mais c'est sa clé privée qui lui permettra de décrypter le message. L'astuce, c'est que la seconde dépend de la première, mais sans qu'il soit possible de la retrouver. Simplement parce que s'il est facile de générer des nombres premiers de très grande taille, ainsi que de les multiplier, la puissance de calcul manque pour retrouver, à partir de ce résultat, les nombres premiers de départ. Bien sûr, le système n'est sécurisé qu'avec des clés de grande taille, généralement 1024 bits. L'avantage majeur de la cryptographie à clé publique réside dans l'absence de transfert de clé, donc du risque que celle-ci soit interceptée. En revanche, la longueur des clés rend les algorithmes très lents. La solution se trouve dans un système hybride, tel que le logiciel PGP (*Pretty Good Privacy*) : le message est codé par une clé privée générée aléatoirement, et celle-ci est elle-même chiffrée avec l'algorithme RSA avant d'être transmise.

2.2 – Systèmes de PGP développés pour la transmission sécurisée

La transmission sécurisée est l'ensemble des moyens mis en œuvre pour réduire la vulnérabilité d'un système de communication contre les menaces accidentelles ou intentionnelles. C'est pour cela plusieurs travaux ont été proposés et développés dans ce domaine.

La cryptographie est l'un des composants fondamentaux pour une transmission sécurisée, le crypto-système RSA est la technique de sécurité le plus attractif et populaire pour de nombreuses applications, telles que le commerce électronique et l'accès sécurisé à l'internet. Le système de cryptage RSA prend grand coût de calcul. Dans de nombreuses applications RSA, l'utilisateur utilise une clé publique de petite taille pour accélérer l'opération de chiffrement. Toutefois, l'opération de déchiffrement doit prendre coût de calcul plus important pour effectuer une exponentiation modulaire. Ren-Junn Hwang et Feng-Fu ont proposé dans leurs article [6], une méthode de déchiffrement efficace basé sur le théorème des restes chinois (*Chinese Remainder Theorem*). La méthode de déchiffrement proposé ne prend que 10% des coûts de calcul de la méthode de déchiffrement traditionnelle. Elle réduit également de 66% les coûts informatiques que celle des méthodes de décryptage sur la base de CRT seulement. En un mot, la vitesse de cette méthode proposée est presque 2,9 fois plus rapide que la méthode de déchiffrement basé sur CRT seulement. La méthode proposée améliore les performances de l'opération de déchiffrement RSA.

Les algorithmes de la cryptographie à clé privée sont les plus rapides par rapport aux algorithmes de la cryptographie à clé publique. Verma et Agarwal [7] ont fait une étude qui fournit la comparaison des performances entre quatre algorithmes de chiffrement les plus couramment utilisés: DES, 3DES (*Triple DES*), Blowfish et AES. La comparaison a été effectuée sur des blocs de données de différentes tailles pour évaluer la vitesse des algorithmes de chiffrement et de déchiffrement. L'analyse est basée sur des performances de ces algorithmes sous différents matériels et plateformes logicielles. Il a été conclu que le Blowfish est le meilleur algorithme en vertu de la sécurité contre les attaques non autorisée en tenant compte de la vitesse.

Le PGP est le plus largement utilisé pour sécuriser les courriers électroniques. Il promet la confidentialité, l'intégrité et l'authentification de ses utilisateurs. Ces services de sécurité sont fournis grâce aux divers algorithmes cryptographiques. Étant donné un ensemble de données, les utilisateurs peuvent choisir des algorithmes particuliers selon leurs besoins. Les différents algorithmes fournissent des niveaux de sécurité différents. Kumar et Kashyap [8] ont proposé une méta-heuristique basée sur le évolutif multi-objectif d'optimisation pour la sélection des algorithmes appropriés pour PGP selon les besoins des utilisateurs, de niveaux de coûts, et de la sécurité.

Katagishi et Asami [9] ont développé une passerelle de messagerie pour fournir une sécurité renforcée (SEMAIL : *Secure Email*) qui assure la confidentialité des e-mails sans obliger les utilisateurs à installer un nouveau logiciel ou d'apprendre les procédures d'exploitation complexes. Le système SEMAIL, dans lequel un programme de chiffrement PGP est utilisé.

Kurniawan et Albone [10] ont créé une mini-application du PGP qui a la fonction de cryptage / description et les signatures numériques. Les autres fonctions de PGP avaient été enlevées dans le but de faciliter à tout le monde de préserver la confidentialité de leurs messages.

Les menaces d'intercepter et manipuler des e-mails est un problème réel. Donc beaucoup de logiciels ont été proposées dans le but de sécuriser les courriers électroniques, comme le PGP, OpenPGP et S / MIME (*Secure / Multipurpose Internet Mail Extensions*). Ces logiciels posent encore un autre problème difficile pour la gestion sécurisée et fiable des clés cryptographiques utilisées. Pour résoudre ce problème, Jang et Nepal [11] ont proposé un nouveau protocole pour un système de messagerie sécurisée en utilisant la fonctionnalité de chiffrement basée sur le matériel de *Trusted Platform Module* (TPM) et le concept Ephemerizer. En basant sur les avantages de ces deux technologies, ce protocole assure une sauvegarde de clés cryptographiques de telle sorte que

seules des expéditeurs de courrier électronique et les destinataires peuvent lire les messages e-mails. Par ailleurs, le protocole garantit que personne ne peut lire les messages e-mail qui ont expiré ou qui ont été supprimés.

Will Zfone [12] est un programme de cryptage pour la voix sur IP (Voice Over IP) des appels téléphoniques. VoIP utilise le concept de PGP dans le cryptage et le décryptage des données sonores.

La cryptographie et la sténographie sont des techniques les plus largement utilisées pour surmonter les menaces. La cryptographie consiste à convertir un message texte dans un algorithme de chiffrement illisible. D'autre part, la sténographie intègre message dans une couverture médiatique. Marwaha, P [13] à proposé un système avancé de cryptage des données qui combine les caractéristiques de la cryptographie et de la sténographie avec la dissimulation des données multimédia. Ce système est souvent mis en œuvre dans les fichiers image. Cependant l'incorporation des données dans l'image change de couleur dans ses fréquences d'une manière prévisible. Pour surmonter cette prévisibilité, ils ont proposé le concept de la cryptographie multiple où les données seront cryptées en un chiffre et le chiffre sera caché dans un fichier image multimédia dans un format crypté. Puis ils ont employé des techniques cryptographiques traditionnelles pour atteindre le cryptage des données, et des algorithmes de sténographie visuels pour masquer les données cryptées.

3. Supports de Transmission

Une transmission d'information se fait toujours à distance, un support physique assure le lien entre la source et le destinataire. Les supports de transmission sont nombreux. Parmi ceux-ci, on distingue : les supports métalliques, non métalliques et immatériels. Les supports métalliques, comme les paires torsadées et les câbles coaxiaux, sont les plus anciens et les plus largement utilisés, ils transportent des courants électriques. Les supports de verre ou de plastique, comme les fibres optiques, transmettent la lumière, tandis que les supports immatériels des communications sans fil propagent des ondes électromagnétiques. La fig.1.1 illustre un schéma de base d'une chaine de transmission.

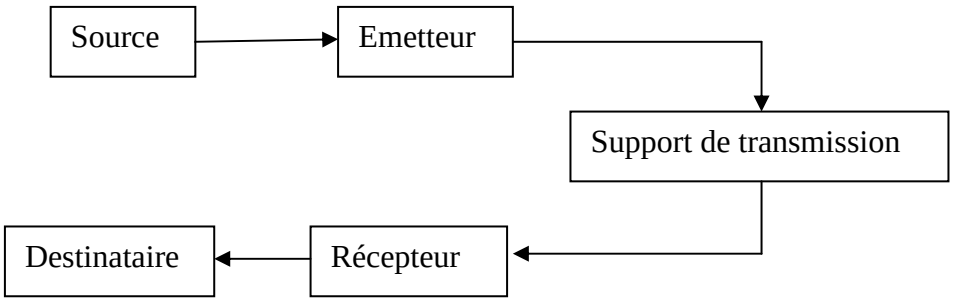

Fig.1. 1. Schéma de base d'une chaine de transmission numérique.

3.1. Supports électriques filaires

L'information est véhiculée par un signal électrique, c'est à dire une onde électromagnétique se propageant à travers un câble métallique. On trouve deux catégories de lignes de transmission utilisées en télécommunications :

- câble bifilaire, de bande passante faible et réservé pour les transmissions à bas débit (inférieur à 2 Mbits/s pour le réseau téléphonique). Il s'agit le plus souvent de paires bifilaires torsadées afin de réduire la surface de couplage aux perturbations extérieures.
- câble coaxial, de bande passante plus importante et qui permet de réaliser des transmissions avec un débit relativement élevé (jusqu'à 565 Mbits/s sur le réseau téléphonique). Son avantage par rapport au câble bifilaire est d'être blindé, réduisant ainsi le couplage des perturbations électromagnétiques, et de présenter un milieu de propagation quasi uniforme le long de la ligne [14].

3.2. Supports optiques filaires

Les fibres optiques sont des guides pour les ondes électromagnétiques dont les fréquences sont de l'ordre du spectre visible. La lumière est guidée le long d'une fibre par réflexions multiples. La figure 1. 2 décrit la structure d'une fibre optique ainsi que le principe de la propagation de la lumière le long de la fibre [14].

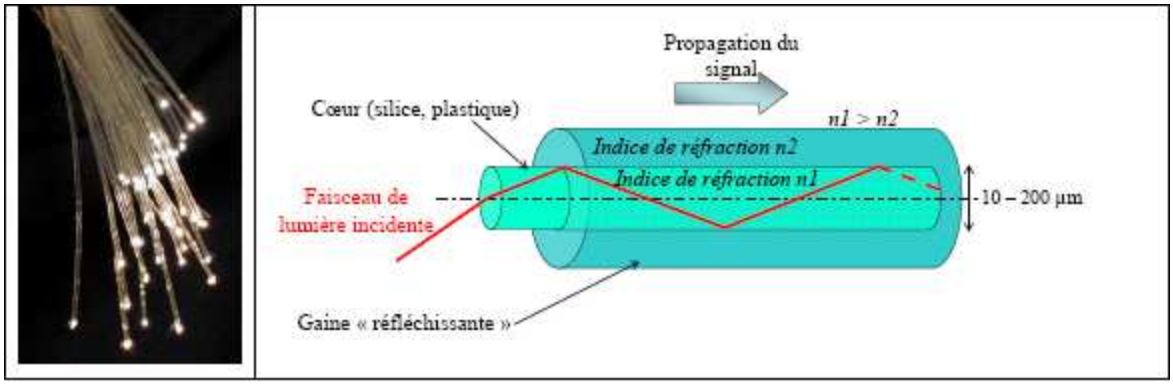

Fig.1. 2. Guidage d'un faisceau lumineux par une fibre optique.

Une fibre optique est constituée d'un fil de verre très fin. Elle comprend un cœur, dans lequel se propage la lumière émise par une diode électroluminescente ou une source laser, et une gaine optique dont l'indice de réfraction garantit que le signal lumineux reste dans la fibre.

Les avantages de la fibre optique sont nombreux : diamètre extérieur de l'ordre de 0,1 mm, poids de quelques grammes au kilomètre. Cette réduction de taille et de poids la rend facile à utiliser. En outre, sa très grande capacité permet la transmission simultanée de nombreux canaux de télévision, de téléphone. Enfin, l'insensibilité des fibres aux parasites électromagnétiques est un avantage très apprécié, puisqu'une fibre supporte sans difficulté la proximité d'émetteurs radioélectriques. Par ailleurs, elle résiste bien aux écarts de température. La fibre optique constitue la plupart des artères des réseaux de télécommunications et des réseaux locaux à très haut débit.

Malgré tous ces avantages, les principaux points négatifs concernent la fragilité de fibres et de leurs connecteurs, ainsi que le coût d'installation et d'entretien des réseaux en fibres optiques.

3.3. Supports Radio

Les ondes électromagnétiques se propagent dans l'atmosphère ou dans le vide, l'absence de support matériel apporte une certaine souplesse et convient aux applications comme la téléphonie ou les télécommunications mobiles, sans nécessiter la pose coûteuse de câbles. Le dispositif de base pour transmettre ou recevoir un signal à travers le canal radioélectrique ou hertzien est une antenne. Les radiocommunications s'étendent sur un spectre très large (de plusieurs KHz à plusieurs GHz). La figure 1. 3 présente l'occupation du spectre radiofréquence.

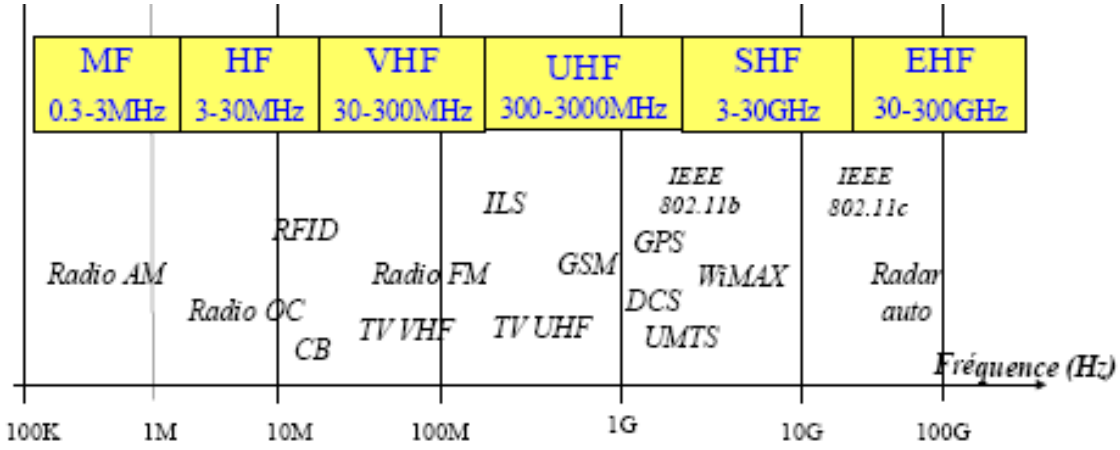

Fig.1. 3. Occupation du spectre radiofréquence.

L'avantage des radiocommunications par rapport aux autres supports de communication (filaire, fibre optique) est le faible coût d'installation d'un réseau à grande échelle, puisqu'il ne nécessite pas d'installer des supports physiques entre chaque nœud et terminaux du réseau, il suffit d'installer une antenne. Néanmoins, il présente de nombreux inconvénients de fait que les ondes radios (notées *RF* pour *Radio Frequency*) se propagent en ligne droite dans plusieurs directions. La vitesse de propagation des ondes dans le vide est de 3.10^8 m/s. Lorsqu'une onde radio rencontre un obstacle, une partie de son énergie est absorbée et transformée en énergie (thermique par exemple), une partie continue à se propager de façon atténuée et une dernière peut éventuellement être réfléchie. L'atténuation augmente avec l'augmentation de la fréquence ou de la distance. De plus lors de la collision avec un obstacle, la valeur de l'atténuation dépend fortement du matériel composant l'obstacle. Généralement les obstacles métalliques provoquent une forte réflexion, tandis que l'eau absorbe le signal.

4. Problèmes rencontrés dans un canal de transmission

4.1. Bruit :

Un canal de transmission idéal doit restituer à la sortie le même signal qu'à l'entrée mais un retard pur τ est inévitable à cause du temps de propagation. Par contre lors de la transmission à travers le canal, le signal subit les atténuations et les déformations propres au canal, ainsi que le bruit provenant de perturbateurs externes. Le canal n'est pas le seul responsable de l'ajout de bruit au signal utile puisque l'ensemble des circuits de réception et de régénération du signal ajoute une part non négligeable de bruit [14].

Le bruit est, par définition, un signal parasite aléatoire, le plus souvent d'origine thermique. Tout signal de fréquence F dont l'amplitude est inférieure ou égale à celle du bruit, ou sous le seuil de bruit, à la fréquence F ne pourra être différencié du bruit par un dispositif électronique de réception.

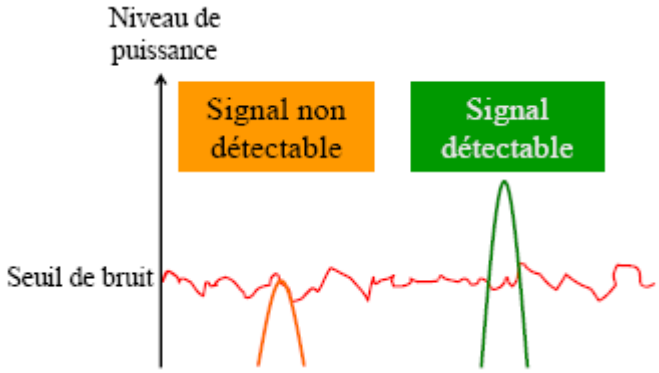

Fig.1. 4. Détection d'un signal au dessus du seuil de bruit.

Le bruit définit donc la limite basse en amplitude permettant la détection d'un signal. Au cours du dimensionnement d'un canal de transmission, il faudra tenir compte du niveau de bruit afin de définir la sensibilité du récepteur. Le bruit peut être caractérisé de plusieurs manières : par sa densité spectrale, c'est-à-dire la répartition énergétique en fonction de la fréquence (puissance par hertz), par sa fonction de répartition ou densité de probabilité en amplitude, et par le rapport signal a brouit SNR (*Signal to Noise Ratio*) qui définit le rapport minimum à respecter entre la puissance du signal sur la puissance du bruit afin de garantir une réception de qualité du signal.

Les bruits existants sont : Bruit Blanc, Bruit de grenaille, Bruit thermique, Bruit d'un circuit actif et facteur de bruit, et Bruit d'une antenne.

La solution apportée par les canaux de transmission pour répondre à la contrainte du bruit et afin de résister aux perturbations induites par le support de transmission. Le récepteur reçoit en général un signal faible, bruité et distordu. Il doit être en mesure de le reconstruire puis de l'interpréter afin de retrouver le signal d'origine. Pour cela un signal à transmettre subit en général des opérations de codage de source, de codage de canal, de modulation, de mise en forme. Il subit les opérations inverses en réception.

4.2. Problèmes de la bande passante

La bande passante est la bande de fréquences dans laquelle les signaux appliqués à l'entrée du support de transmission ont une puissance de sortie supérieure à un seuil donné après traversée du support. Le seuil fixé correspond à un rapport déterminé entre la puissance du signal d'entrée et la

puissance du signal trouvé à la sortie. En général, on caractérise un support par sa bande passante à 3 dB (décibels), c'est-à-dire par la plage de fréquences à l'intérieur de laquelle la puissance de sortie est, au pire, divisée par deux.

Les supports ont une bande passante limitée. Certains signaux s'y propagent correctement, alors que d'autres ne les traversent pas. Intuitivement, plus un support a une bande passante large, plus il transporte d'informations par unité de temps.

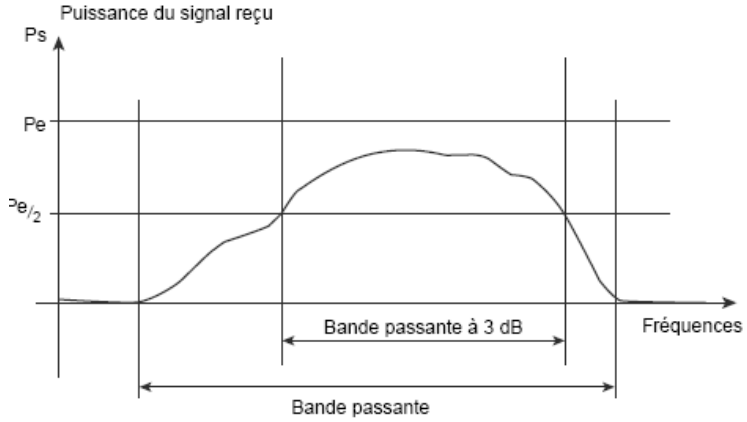

Fig.1. 5. La bande passante d'un canal de transmission.

4.3. Problèmes de sécurité

Les transmissions de données à travers le canal radioélectrique ne peuvent pas être sécurisées et n'importe quelle antenne adaptée à la fréquence de transmission est susceptible de capter le signal. Par conséquent, l'information transmise sera interceptée par des destinataires non légitimes, d'autre part la fibre optique et les supports filaires, et malgré le signal est guidé le long de la ligne et ne peut sortir que par l'autre bout de la ligne, interdisant toute fuite du signal et assurant une sûreté de transmission très élevée, par contre les informations transportées peuvent être piratées sur les réseaux de télécommunications trop vaste tel que l'internet, comme c'est le cas de la messagerie électronique. Donc un système de protection est nécessaire, cette contrainte doit être prise en considérations dans l'architecture d'un bon canal de transmission.

La solution à ce problème à été faite dans l'étage de cryptage est incorporée dans le cas où l'on souhaite sécuriser le transfert des données et leur archivage.

5. Architecture générale d'un canal de transmission sécurisée

La figure 1. 6 fournit une solution générale quasiment de tous les problèmes cités précédemment, ainsi elle fournit une vue générale d'un canal de transmission et un récapitulatif des principales

contraintes qu'il doit respecter pour qu'un message numérique ou analogique soit transmis en toute sécurité et sans erreur [14].

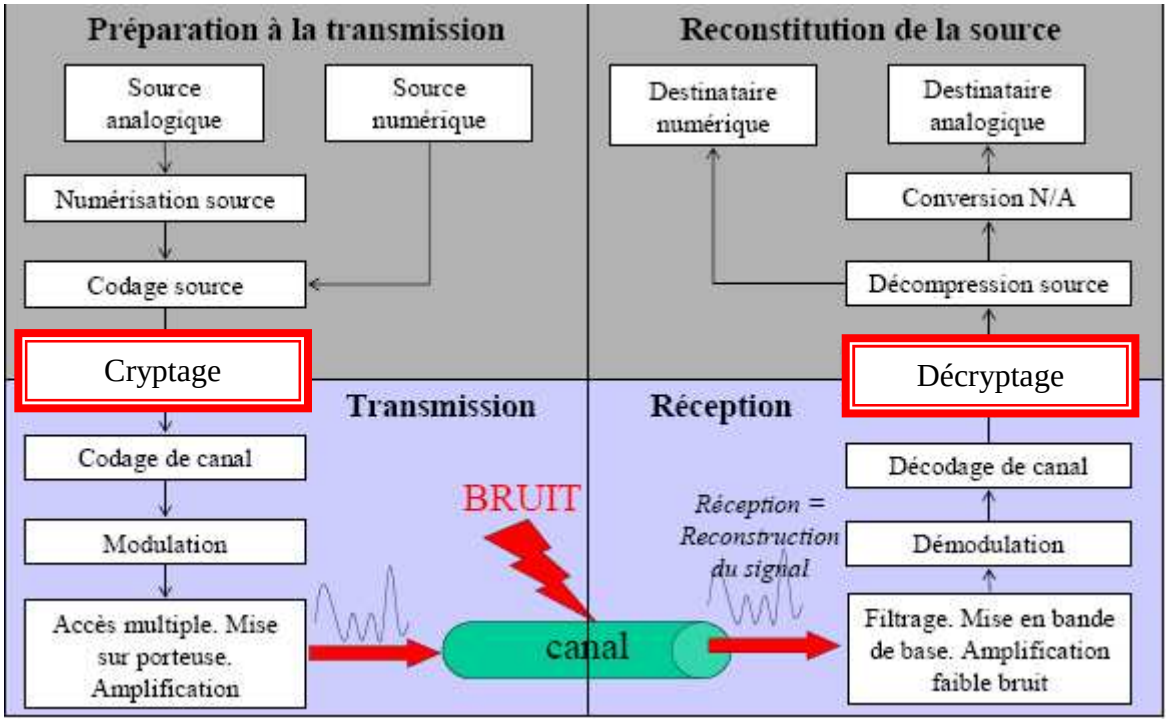

Fig.1.6. Architecture générale d'un canal de transmission sécurisée.

La source primaire d'information peut être soit de type analogique, qu'on numérise ensuite, soit directement de type numérique. L'information analogique est ensuite échantillonnée et numérisée à travers un étage de conversion analogique numérique. La taille du message binaire original ainsi produit est en général très importante et contient en outre un grand nombre de redondance. Il subit alors un codage de source, qui a pour but de le mettre dans un format standard d'échange et de réduire sa taille (compression), ensuite des étages de modulation et de multiplexage sont insérés dans de l'émetteur. Ainsi, des étages de filtrage, démodulation, décodage de canal, décryptage, décompression source, et des convertisseurs numérique-analogique sont incorporés dans le récepteur pour la bonne reconstitution de signal émis par l'émetteur.

A travers ce mémoire, on supposera que le signal émis est vierge de tout parasite puisque toutes les précautions ont été prises afin d'assurer la qualité du signal émis. Ainsi, notre but dans ce mémoire est de réaliser un étage de cryptage au niveau de l'émetteur, et un étage de décryptage au niveau de récepteur pour sécurisation des données.

6. Transmission sécurisée type M2M (*Machine to Machine*)

Le développement des technologies de communication et des équipements intelligents, combinés à l'informatique d'entreprise, a permis l'émergence d'un nouveau type d'usages et d'applications : le

M2M (*Machine To Machine*). En effet, la banalisation des objets communicants fixes ou mobiles, la baisse des coûts de communication, l'amélioration de la performance des réseaux et la disponibilité de plates-formes de services dédiées permettant de gérer une multitude d'objets, ont ouvert aux entreprises de nouveaux domaines d'activités, impactant directement leurs processus et leurs offres. Le concept de M2M utilise les télécommunications et l'informatique pour permettre des communications entre machines, et ceci sans intervention humaine. En français, le M2M désigne de manière commune « la communication de machine à machine ».

La communication entre machines tend à devenir une communication d'une machine ou d'un réseau de machines vers un serveur centralisant les données remontées par ces machines et assurant également le pilotage à distance de ces machines. On parle alors de gestion à distance d'un parc de machines (*remote devices management*). Le déploiement, l'opération et la maintenance de larges réseaux de machines constitue une nouvelle activité d'opérateurs télécoms spécialisés sur les machines, les opérateurs M2M [15].

6.1. Architecture M2M :

Les principaux éléments d'un système de type M2M comprennent (Figure 1. 7):

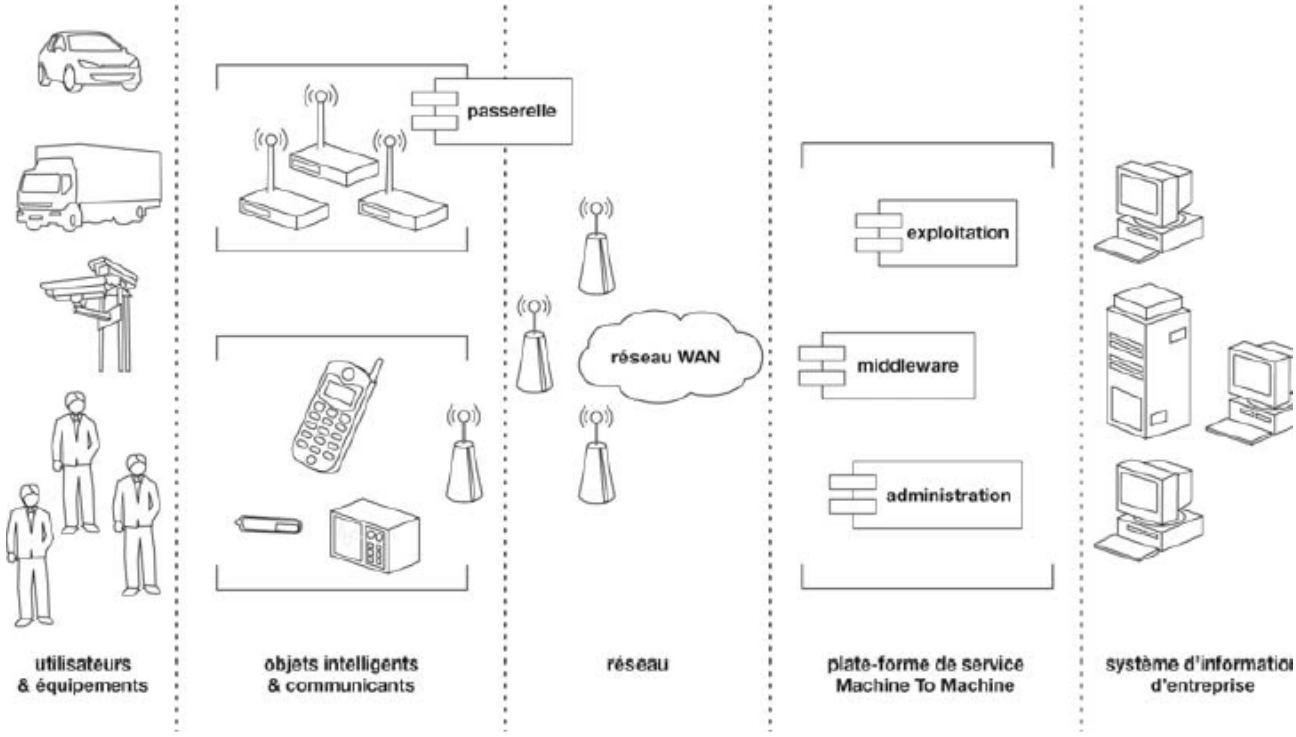

Fig. 1.7- Architecture M2M.

- Un appareil ou un groupe de dispositifs capables de répondre aux demandes de données contenues dans ces appareils ou capables de transmettre des données contenues dans ces dispositifs autonomes.
- Un lien de communication pour connecter l'appareil, voire un groupe de périphériques à un ordinateur serveur ou un autre appareil.
- Un logiciel, processus, ou interface par laquelle les données peuvent être analysées, a déclaré, et / ou mises en œuvre.

Le plus souvent, les systèmes M2M ont une tâche bien spécifique, c'est-à-dire que ce sont des systèmes conçus pour un périphérique spécifique, ou une catégorie très restreinte des dispositifs d'une industrie.

6.2. Domaine d'application de M2M

Le M2M existe aujourd'hui, il peut répondre à des besoins très concrets des entreprises. Cette technologie s'insère peu à peu dans (ou sur) les machines, les appareils, les véhicules, les emballages, les objets quotidiens, les équipements, les espaces publics. Mais aussi les arbres, les zones inondables, les forêts qui sont exposées à des incendies, les animaux domestiques ou sauvages et finalement, nos propres corps.

- Une entreprise suisse a conçu un détecteur pour placer de stationnement sur voieries qui informe en temps réel de la disponibilité de celle-ci. Le M2M peut améliorer la mobilité en informant l'automobiliste en quête de place et apporter les statistiques qui font défaut à nos décideurs.
- Un fabricant d'instruments pour la production de pétrole et de gaz, utilise le M2M pour permettre à ses clients de recueillir à distance des données sur les débits, les pressions, températures, niveaux de réservoir et de l'équipement.
- Un fabricant de pompes à injection pour puits utilise le M2M pour donner à ses clients un moyen d'ajuster le fonctionnement de la pompe à distance (en fonction des conditions météorologiques). On évite ainsi l'ajustement sur le site.
- Dans l'entreprise, la gestion de chaînes de valeurs et de processus complexes, inter-entreprises, en constant ajustement : où est qui et quoi ? Pour qui travaille telle machine et comment organise-t-elle sa propre chaîne d'approvisionnement ? Etc.

- L'automobile de demain communiquera avec les autres véhicules et avec les infrastructures routières pour éviter (ou détecter) des accidents, ou encore pour faire s'afficher en projection sur le pare-brise les panneaux indicateurs, les services à proximité, le prix du carburant dans la prochaine station-service, pour réagir automatiquement en cas d'incident, etc.
- Dans la ville, qui est déjà peuplée de capteurs de toutes sortes, le M2M sera utilisé pour gérer les grandes infrastructures urbaines, les transports en commun, les péages urbains etc.
- Le M2M sera de plus en plus utilisé dans l'habitat, non pas dans le sens d'une automatisation totale, mais pour le rendre plus facile à gérer, à vivre et à adapter à chacun. Ça commence par des choses aussi simples que des interrupteurs qui n'ont pas besoin d'être raccordés à un fil électrique pour commander l'éclairage d'une pièce.
- Dans la vie quotidienne, la santé et l'assistance à domicile auront besoin du M2M : télédiagnostic, médicaments intelligents, robots-nurses ou planchers qui savent détecter un accident ou un malaise, dispositifs d'alerte permettant de ne mobiliser du personnel para-médical qu'à bon escient.
- En matière de surveillance environnementale et de développement durable, le M2M permettra par exemple de surveiller en continu des zones qui sont exposées à des incendies, de réguler automatiquement l'arrosage et l'utilisation de produits chimiques dans l'agriculture, de mieux tracer les chaînes alimentaires, …etc.

7. Discussion

Nous constatons que la transmission est nécessaire dans la communication, pour échanger des données d'une manière générale. Ces données peuvent être des données publiques, donc elles ne nécessitent pas une surveillance, ou un niveau de sécurité. Par contre certaines données portent des informations confidentielles et secrètes. Pour cela, les communicants doivent utiliser la transmission sécurisée pour véhiculer ce genres d'informations. Cette transmission sécurisée est basée sur des outils mathématiques et informatiques tels que la cryptographie, et le système PGP qui permettent d'offrir aux communicants un outil de préserver la confidentialité des informations à échanger. C'est ce que nous allons développer et détailler dans les chapitres suivants.

CHAPITRE 2

La Cryptographie en Transmission Sécurisée

1. Introduction

De tous temps, les services secrets ont utilisé toutes sortes de codages et de moyens cryptographiques pour communiquer entre agents et gouvernements, de telle sorte que les "ennemis" ne puissent pas comprendre les informations échangées. La cryptologie a alors évolué dans ces milieux fermés qu'étaient les gouvernements, les services secrets et les armées. Ainsi, très peu de gens, voire personne n'utilisait la cryptographie à des fins personnelles. C'est pourquoi, pendant tant d'années, la cryptologie est restée une science discrète. De nos jours en revanche, il y a de plus en plus d'informations qui doivent rester secrètes ou confidentielles. En effet, les informations échangées par les banques ou un mot de passe ne doivent pas être divulgués et personne ne doit pouvoir les déduire. C'est pourquoi ce genre d'informations est crypté. Pour ces mêmes raisons d'insécurité sur internet, et par un besoin humain d'intimité que la cryptographie à des fins purement personnelles s'est développée sur le réseau : pour la messagerie électronique. En effet lorsque l'on envoie un message électronique par internet, on peut préférer qu'il reste discret vis à vis de la communauté internet, voire qu'il ne soit compréhensible que par le destinataire du message. En d'autres termes, la cryptographie peut servir si l'on veut envoyer un message confidentiel, ou un message intime à quelqu'un. Dans ce chapitre nous allons voir les différentes techniques de cryptages qui existent, et voir les avantages et les inconvénients de ces techniques.

2. Définition de la cryptographie

La cryptographie est une science mathématique, et de la sécurisation des données, elle consiste à étudier et concevoir des procédés de cryptage [16]. Elle permet ainsi de stocker des informations confidentielles ou de les transmettre sur des réseaux non sécurisés (tels que l'internet), afin qu'aucune personne autre que le destinataire ne puisse les lire.

La cryptographie doit apporter un certain nombre des fonctionnalités telles que la confidentialité, l'authentification, l'intégrité, et la non répudiation.

- Confidentialité : c'est la propriété qu'une information n'est ni disponible ni divulguée aux personnes, entités ou processus non autorisés. Lors d'une communication, il s'agit d'empêcher un tiers de prendre connaissance de l'information contenue dans un message transmis sur un canal non sécurisé.
- Authentification : consiste à vérifier l'identité des différentes parties impliquées dans le dialogue. L'authentification préserve de l'usurpation d'identité et participe de la confidentialité dans le sens où elle assure que celui qui émet est bien l'entité attendue.

- Intégrité des données : On doit éviter que les données transmises soient modifiées ou forgées par un adversaire. Plus précisément : l'intégrité est la prévention d'une modification non autorisée de l'information.
- Non répudiation : permet de prouver qu'un message a bien été émis par son initiateur. Le message ne peut donc plus ensuite être dénié par celui qui l'a émis.

Toutes ces fonctionnalités sont assurées par le biais des algorithmes de chiffrement, et de déchiffrement.

A l'inverse, la cryptographie possède des ennemis qui font l'étude des procédés cryptographiques dans le but de trouver des faiblesses, permettant de retrouver de texte en clair sans connaitre la clé de déchiffrement. Ces ennemis portent le nom « Les crypto-analystes », ou « les attaquants ». L'ensemble formé de la cryptographie, et de la cryptanalyse constitue la cryptologie.

3. Le principe de base

L'objective fondamentale de la cryptographie est de permettre à deux personnes, appelées traditionnellement Alice et Bob, de communiquer à travers un canal peu sûr de telle sorte qu'un opposant, Oscar ne puisse pas comprendre ce qui est échangé [16]. Le canal peut être par exemple une ligne téléphonique ou tout autre réseau de communication voir la Fig.2.1

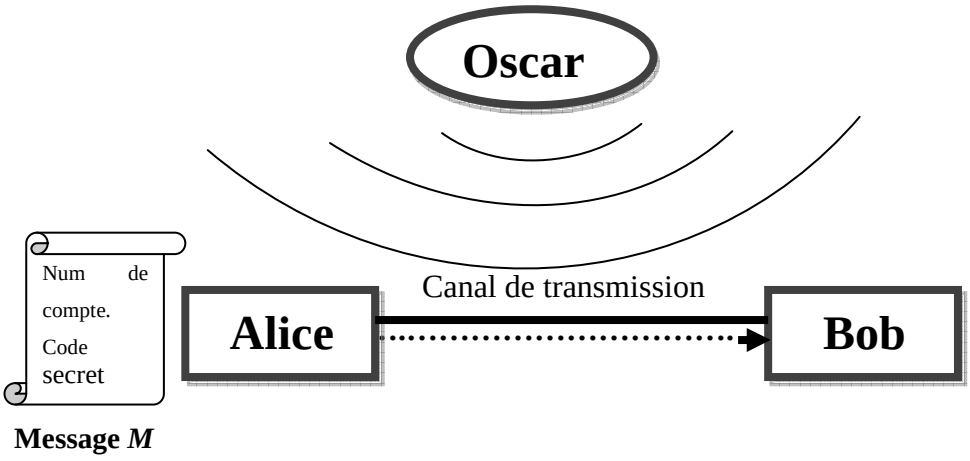

Fig.2.1 - Transmission d'un message sur un canal peu sûr

L'information que Alice souhaite transmettre à Bob est appelé texte clair. Il peut s'agir d'un texte en français, d'une donnée numérique ou n'importe quoi d'autre, de structure arbitraire. Le processus de transformation d'un message ***M*** de telle manière à le rendre incompréhensible est appelé chiffrement. On génère ainsi un message chiffré ***C*** obtenu à partir d'une fonction de chiffrement **E** par $C = E(M)$. Le processus de reconstitution du message clair à partir du message chiffré appelé déchiffrement utilise une fonction de déchiffrement **D**. Il donc requiert donc que :

$$D(C) = D\big(E(M)\big) = M \qquad 2\text{- }1$$

Autrement dit, **D** et **E** sont injectifs.

Un algorithme cryptographique est la définition des fonctions mathématiques utilisées pour le chiffrement et le déchiffrement. En pratique, les fonctions **E** et **D** sont paramétrées par des clés K_e *et* K_d qui peuvent prendre l'une des valeurs d'un ensemble appelé espace des clés. On a donc la relation suivante [16] :

$$\begin{cases} E_{K_e}(M) = C \\ D_{K_d}(C) = M \end{cases} \qquad 2\text{- }2$$

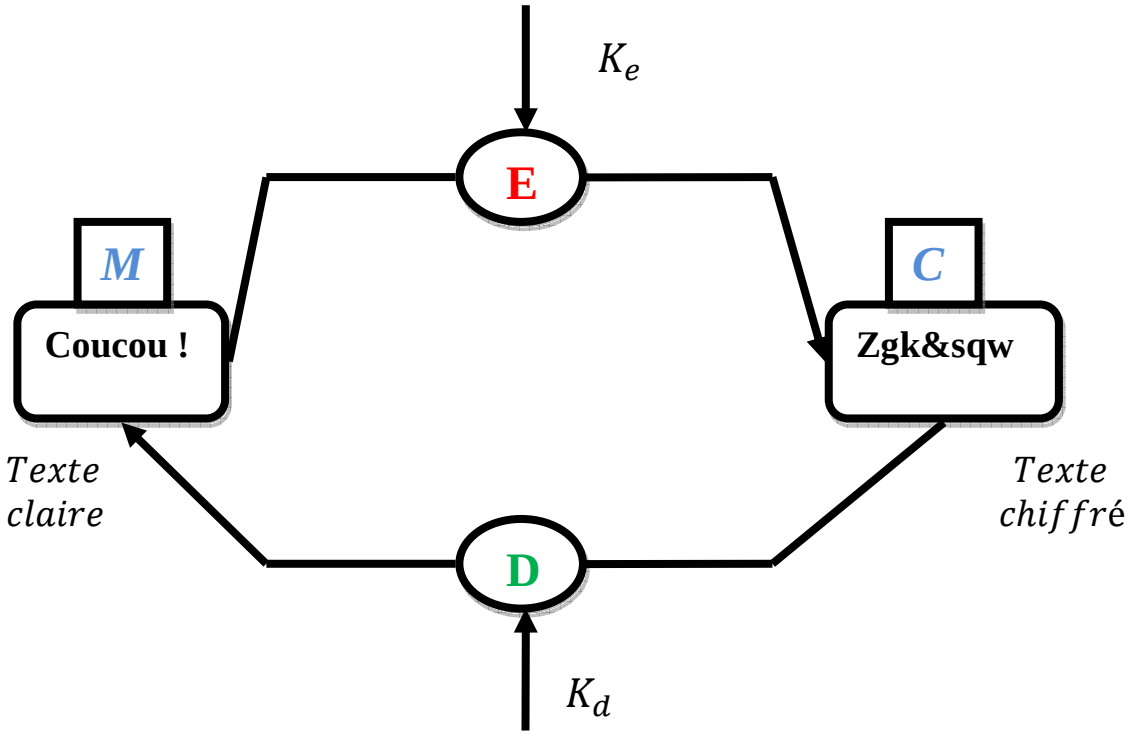

Fig.2.2 - Principe de chiffrement et de déchiffrement d'un message

4. Complexité d'un algorithme

Un algorithme est un ensemble d'instructions qui permettent le calcul d'une valeur de sortie, pour des données d'entrée dans un certain ensemble. C'est une notion très générique, et pour s'en donner

une autre idée, il est possible de considérer qu'un algorithme est une sorte de programme informatique, permettant l'exécution d'une tache [17].

Chaque algorithme est caractérisé par un temps d'exécution, calculé comme le nombre d'´etapes élémentaires dont il a besoin pour calculer la sortie. Ce temps dépend des données d'entrée et d'aléas internes de l'algorithme.

La qualité d'un algorithme de cryptage repose sur sa robustesse (temps pour être cassé) et sa facilité de mise en œuvre.

5. Principe de Kerckhoffs

En 1883, dans la cryptographie militaire, Kerckhoffs énonçait les six lois suivantes [18] :

1. Le système doit être matériellement, sinon mathématiquement, indéchiffrable
2. Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi pour la cryptographie à clé publique.
3. La clé doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants.
4. Il faut qu'il soit applicable à la correspondance télégraphique.
5. Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes.
6. Enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

On distingue deux sortes de la cryptographie : la cryptographie classique, et la cryptographie moderne.

6. La cryptographie classique

Dans la cryptographie classique, la méthode et la clé de chiffrement ainsi que celles de déchiffrement sont connues par l'émetteur et le destinataire. L'émetteur et le destinataire doivent se mettre préalablement d'accord sur la clé. La plupart des méthodes de chiffrement classiques reposent sur deux principes, essentiels : la substitution et la transposition. La substitution consiste à remplacer certaines lettres par d'autres, ou par des symboles. La transposition signifie qu'on permute

les lettres du message afin de le rendre inintelligible. Nous présentons ci-dessous quelques exemples de la cryptographie classiques.

6.1 Le code de César

Le code secret de Jules César est le plus ancien dans l'histoire da la cryptographie. Dans ce cas, l'algorithme constitue à décaler les lettres de l'alphabet et la clé correspond au nombre de caractères de décalage.

Par exemple, si vous codez le mot « SECRET » à l'aide de la valeur 3 de la clé de César, l'alphabet est décalé de manière à commencer à la lettre D.

Ainsi, l'alphabet

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Si vous décalez le début de 3 lettres, vous obtenez

DEFGHIJKLMNOPQRSTUVWXYZABC

Où D = A, E = B, F = C, etc.

Avec ce procédé, le texte en clair « SECRET » est crypté en « VHFUHW ».

Il n'y a que 26 façons différentes de chiffrer un message avec le code de César. Donc, il est très facile de le casser, en testant de façon exhaustive toutes les possibilités.

6.2 Le carré de Polybe

Polybe était un écrivain grec. C'est lui qui a inventé le premier chiffrement de substitution [19]. Le carré de Polybe est basé sur une grille comme celle-ci :

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Tab.2.1 – La grille de Polybe

Chaque lettre est codée par 2 chiffres. Exemple : X=53. Les lettres I et J ne sont pas différenciés. "BONJOUR" sera chiffre par " 12343324344542 ".

6.3 Le chiffrement de Vigenère

En 1586, Blaise de Vigenere a développé son Tracte des chiffres ou Secrètes manières d'écrire, qui ressemble au chiffrement de César, à la différence prés qu'il utilise un mot clé au lieu d'un simple caractère. Pour chiffrer un message, on choisit une clé qui sera un mot de longueur arbitraire. On écrit ensuite cette clé sous le message à chiffrer, en la répétant aussi souvent que nécessaire pour que sous chaque lettre du message à chiffrer, on trouve une lettre de la clé. Pour chiffrer, on fait la somme modulo 26 de chaque caractère du message et du caractère de la clé correspondant.
Exemple : On veut chiffrer le texte "CRYPTOGRAPHIE DE VIGENERE" avec la clé "SECRET".
On commence par écrire la clé sous le texte à chiffrer :

Texte	C	R	Y	P	T	O	G	R	A	P	H	I	E	D	E	V	I	G	E	N	E	R	E
Clé	S	E	C	R	E	T	S	E	C	R	E	T	S	E	C	R	E	T	S	E	C	R	E
Nbres	2	17	24	15	19	14	6	17	0	15	7	8	4	3	4	21	8	6	4	13	4	17	4
x_1 x_2	18	4	2	17	4	19	18	4	2	17	4	19	18	4	2	17	4	19	18	4	2	17	4
+Mod26	20	21	0	6	23	7	24	21	2	6	11	1	22	7	6	12	12	25	22	17	6	8	8
Texte	U	V	A	G	X	H	Y	V	C	G	L	B	W	H	G	M	M	Z	W	R	O	I	I

Tab.2.2 – Un déchiffrement de Vigenère

Bien que ce chiffrement soit beaucoup plus sûr que le chiffrement de César, il peut encore être facilement cassé. En effet, lorsque les messages sont beaucoup plus longs que la clé, il est possible de repérer la longueur de la clé et d'utiliser pour chaque séquence de la longueur de la clé la méthode consistant à calculer la fréquence d'apparition des lettres, permettant de déterminer un à un les caractères de la clé [19].
Pour éviter ce problème, une solution consiste à utiliser une clé dont la taille est proche de celle du texte afin de rendre impossible une étude statistique du texte chiffré.

7. La Cryptographie moderne

Si le but de la cryptographie classique est d'élaborer des méthodes permettant de transmettre des données de manière confidentielle, la cryptographie moderne repose sur l'ensemble des moyens permettant de chiffrer des messages et de les déchiffrer, en utilisant des algorithmes plus complexes et plus résistants aux attaques des crypto-analystes. Ces algorithmes se basent sur une combinaison complexe d'opérations de transposition et de substitution portant sur des longues chaînes de bits.

La cryptographie moderne se compose de deux grandes parties : La cryptographie symétrique et la cryptographie asymétrique.

7.1 Cryptographie symétrique

Les algorithmes symétriques sont aussi appelés algorithmes à clé privée. En effet, lorsque l'on crypte une information à l'aide d'un algorithme symétrique avec une clé secrète, le destinataire utilisera la même clé secrète pour décrypter [20]. Il est donc nécessaire que les deux interlocuteurs se soient mis d'accord sur une clé privée auparavant, par courrier, par téléphone ou lors d'un entretien privé. Il existe de nombreux algorithmes à clés secrètes (DES, IDEA, AES, …).
Dans les algorithmes symétriques, on distingue le chiffrement par bloc et le chiffrement continu.

- Le chiffrement par bloc : le message en clair est décomposé en blocs de même taille; chacun d'eux est chiffré individuellement et de la même manière.
- Le chiffrement continu : chaque unité (lettre dans le cas d'un e-mail) du message en clair est chiffrée une par une.

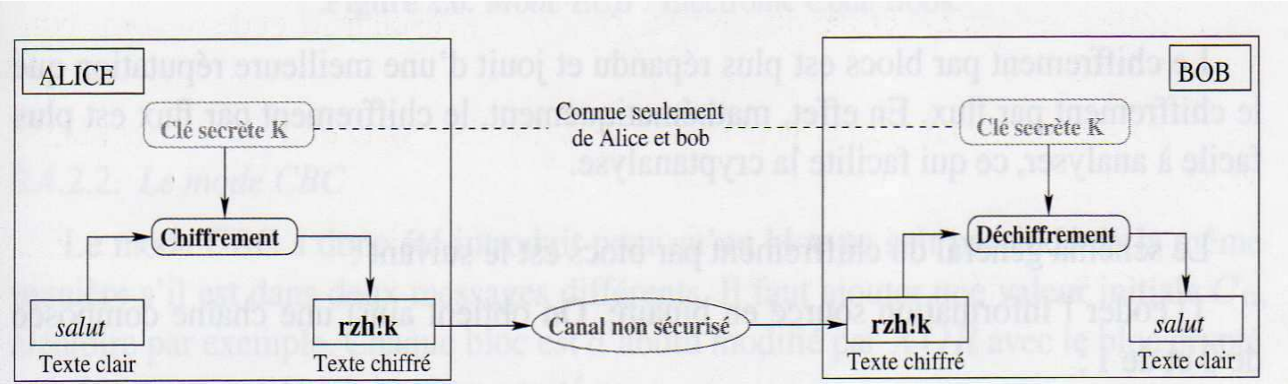

Fig.2.3 - Algorithme de chiffrement symétrique

7.1.1 DES (*Data Encryption Standard*)

DES est l'un des systèmes de chiffrement les plus utilisés. Ce système de chiffrement à clé privée a été conçu dans les années 1970 par IBM pour être un système de chiffrement à clé secrète [20].

7.1.1.1 Algorithme général

Ce système fonctionne par blocs de 64 bits, dont 8 bits (un octet) servent de test de parité (pour vérifier l'intégrité de la clé). Chaque bit de parité de la clé (1 tous les 8 bits) sert à tester un des octets de la clé par parité impaire, c'est-à-dire que chacun des bits de parité est ajusté de façon à avoir un nombre impair de '1' dans l'octet à qui il appartient. La clé a donc une longueur "utile" de 56 bits, ce qui signifie que seuls 56 bits servent dans l'algorithme [20].

L'algorithme se déroule en trois étapes :

1- Soit x un bloc de texte clair de 64 bits. On lui applique une permutation initiale IP fixée pour obtenir une chaine x_0. On a donc :

$x_0 = IP(x) = L_0.\ R_0$, Où L_0 contient les 32 premiers bits de x_0 et R_0 les 32 restants.

2- Seize itérations (ou seize tour) d'une fonction f sont effectuées. On calcule L_i et R_i où $1 \leq i \leq 16$ suivant la règle :

$$\begin{cases} L_i = R_{i-1} \\ R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \end{cases} \qquad 2\text{-}3$$

La Fig.2.4 décrit un tour de chiffrement. La fonction f est une fonction à deux variables, l'une de 32 bits (correspondent à R_{i-1} à la *i*-itération) et l'autre de 48 bits (il s'agit de K_i) dont le calcul sera explicité dans le paragraphe suivant. Les éléments K_i sont obtenus par diversification des bits de la clé initiale K (de 64 bits). Cette opération sera présentée plus loin.

3- La permutation inverse IP^{-1} est appliqué à $R_{16}.L_{16}$ pour obtenir un bloc de texte chiffré $y = IP^{-1}(\ R_{16}.L_{16})$.

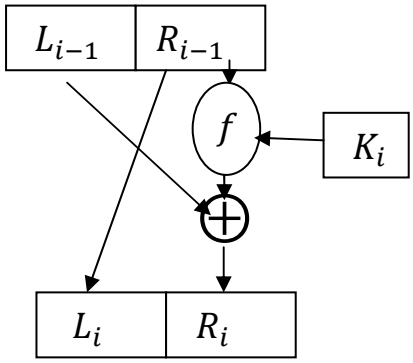

Fig.2.4 – Un tour du DES

Le même algorithme (à quelque légères nuances prés) est utilisé pour le déchiffrement. La sûreté du DES vient de fonction f qui est en fait une substitution suivie d'une permutation. De plus, après seize tours, le résultat est statistiquement « plat », c'est-à-dire que les caractéristiques générales du message source (la fréquence des caractères, le nombre d'espace, etc.) seront indétectables de plus, il dispose d'une caractéristique très importante pour éviter les attaques par analyse différentielle : une légère modification de la clé ou du texte à chiffrer provoque des changements importants dans le texte chiffré [21]. Les détails de la permutation initiale et la permutation finale sont fournit respectivement dans les tableaux Tab.2.3 et Tab.2.4.

Cela signifie que dans le calcul $y = IP(x)$, le 58^{e} bit de x est le premier de y, le 50^{e} bit de x est le second de y, etc. ce formalisme est utilisé dans les deux tableaux.

	58	50	42	34	26	18	10	2
L_i	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
R_i	59	51	43	35	27	19	11	3
	61	53	45	37	29	21	13	5
	63	55	47	39	31	23	15	7

Tab.2.3 – Permutation initiale IP

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

Tab.2.4 – Permutation finale IP^{-1}

7.1.1.2 Détail de la fonction $f(R_{i-1}, K_i)$

On a vu que l'algorithme DES faisait appel à une fonction f prenant deux arguments :

1- Une chaîne R_{i-1} de 32 bits (la partie droite du bloc à chiffrer),

2- Une chaîne K_i de 48 bits (une clé diversifiée).

Le résultat de cette fonction est une chaîne de 32 bits. Le calcul de $f(R_{i-1}, K_i)$ se déroule en plusieurs étapes, illustrées dans la Fig.2.5 :

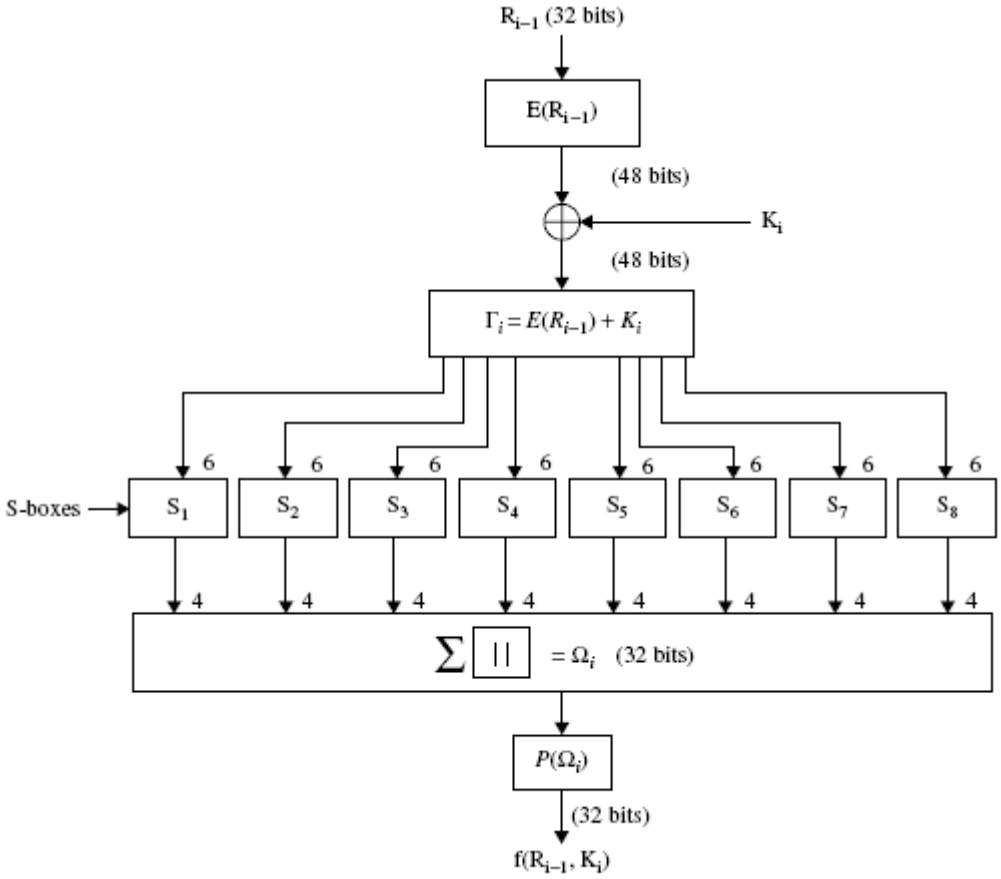

Fig.2.5 : La fonction f

1) - R_{i-1} est augmenté en une chaîne de 48 bits par une fonction d'expansion E.

$E(R_{i-1})$ est composée de tous les bits de R_{i-1} dans certain ordre et seize d'entre eux apparaissent deux fois. La fonction d'expansion est décrite dans le Tab.2.5

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

Tab.2.5 – Fonction d'expansion E

2- $\Gamma = E(R_{i-1}) \oplus K_i$ est calculé. Γ est ensuite découpé en huit sous-chaines consécutives de six bits chacune : $\Gamma = \Gamma_1, \Gamma_2, \ldots \Gamma_8$.

3- l'étape suivante utilise huit « boites-S » S_1, S_2,…. S_8 qui sont dans le Tab.2.6. Chaque boite-S est 4x16 d'entiers compris entre 0 et 15. Soit une sous-chaine de six bits donnée par $\Gamma_i = b_1b_2b_3b_4b_5b_6$. On calcule une chaine de quatre bits $S_i(\Gamma_j)$ de la fonction suivante :

- Les bits b_1b_6 fournissent la représentation binaire de l'indice l de la ligne de S_i à considérer $0 \leq l \leq 3$,
- Les bits $b_2b_3b_4b_5$ fournissent la représentation binaire de l'indice c de la colonne de S_i à considérer $0 \leq c \leq 15$,
- L'intersection de la ligne l et de la colonne c de S_i fournit un entier compris entre 0 et 15, donc une chaine de quatre bits, qui sera le résultat $\{C_i = S_i(\Gamma_j)\}_{0 \leq j \leq 8}$

4- la chaine $C = C_1.C_2 \ldots . C_8$, de longueur 32 bits, est alors réordonnée suivant une permutation fixée P (détaillé dans le Tab.2.7). Le résultat $P(C)$ définit $f(R_{i-1}, K_i)$.

S1	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

S2	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

S3	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

S4	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

S5	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

S6	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

S7	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

S8	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

Tab.2.6 - Les S-Boxes

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

Tab.2.7 - La permutation *p*

7.1.1.3 Détail de la diversification de la clé dans DES

La clé initiale K de 64 bits subit, dans un premier temps, une première permutation CP1 (une réduction de 8 bits les bits de parité de chaque octet). La chaine de caractères résultante de 56 bits est alors divisée en 2 blocs de 28 bits (C_0 et D_0). Chacun d'eux subira alors un décalage à gauche d'une ou deux positions selon valeur de i comme le montre le Tab.2.8. Les deux nouveaux blocs (C_1 et D_1) seront alors concaténés avant une nouvelle permutation CP2 (une autre réduction de 8 bits). On obtient alors une clé partielle K_1 de 48 bits. On réitérera ces opérations 15 fois pour

générer les 15 clés ($K_2, K_3, ..., K_{16}$). Les permutations CP1 et CP2 sont données respectivement par les Tab.2.9 et Tab.2.10.

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Number of left shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Tab.2.8 - La table des décalages à gauche LS

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

Tab.2.9 - La permutation CP1

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

Tab.2.10 - La permutation CP2

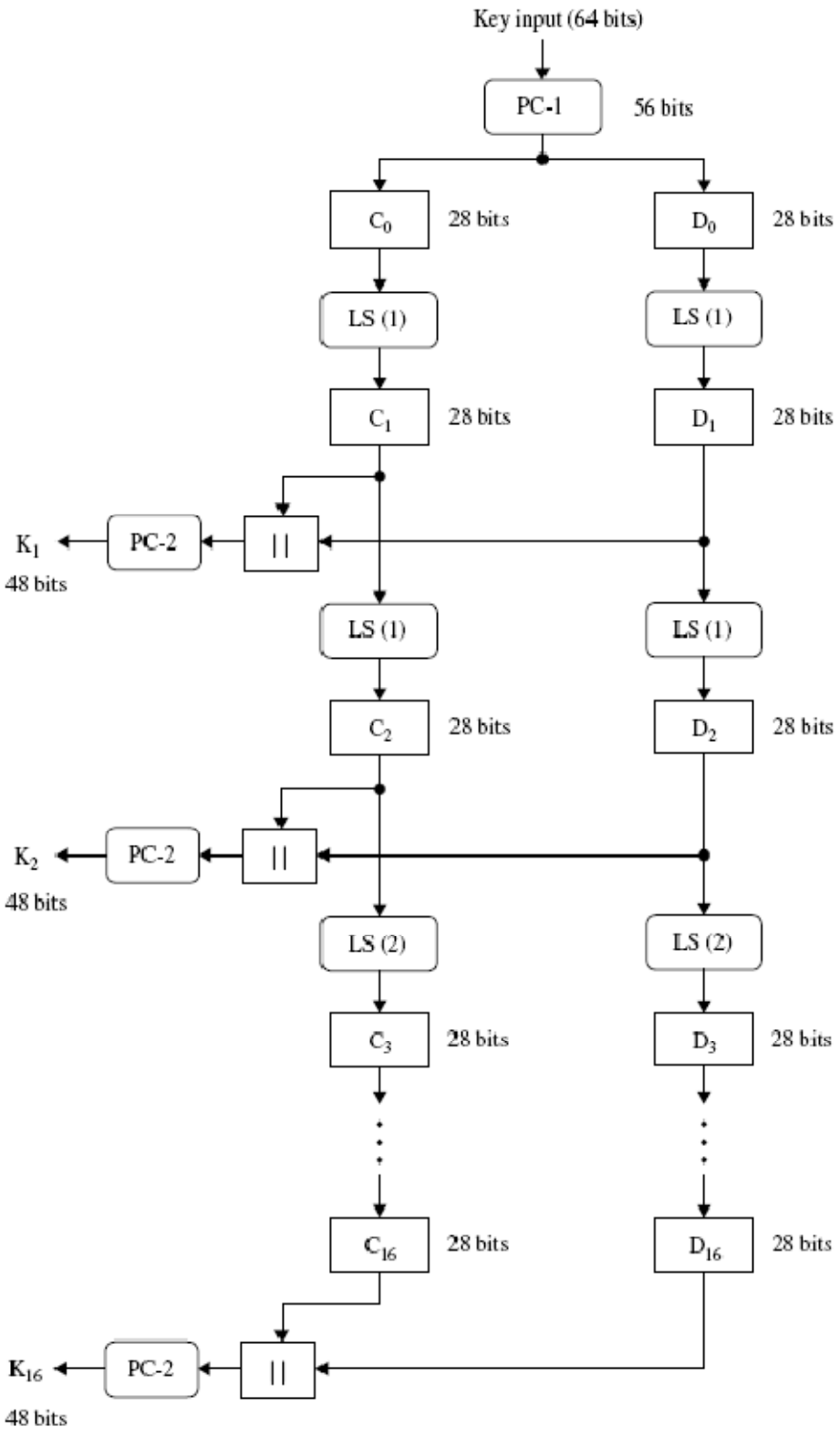

Fig.2.6 - La diversification des clés en DES

7.1.1.4 Le Déchiffrement :

Le déchiffrement est effectué par le même algorithme en inversant l'ordre d'utilisation des clés de tour, c'est-à-dire on commence par la clé K16 à la première itération, et K15 à la seconde, … et K1 à la seizième. Cela est dû au fait que la permutation finale est l'inverse de la permutation initiale [21].

7.1.2 IDEA (*International Data Encryption Algorithm*)

L'algorithme IDEA a été crée par Xuejia Lai et James. Massey [21]. C'est un système de chiffrement par blocs de 64 bits, avec une clé de 128 bits, qui tourne sur 8 rondes. Le même algorithme est utilisé à la fois pour chiffrer et pour déchiffrer. Il utilise trois composants de base qui sont :

- Le *ou exclusif* notée $\oplus$;
- L'addition modulo 2^{16} notée $\boxplus$;
- La multiplication modulo $2^{16}+1$ notée $\odot$.

Le principe est basé sur la difficulté d'inverser les nombres dans un corps intègre abélien $Z//pZ/$.

7.1.2.1 Description

Le fonctionnement d'IDEA est décrit dans la fig.2.7. Le bloc de données de 64 bits est découpé en 4 mots de 16 bits : X_1 X_2 X_3 X_4 qui constituent les entrées de l'algorithme. Celui est divisé en 8 tours.les 4 mots sont combinés pers des Xor, additionnés, multipliés entre eux et avec les 6 mots de 16 bits de la clé. Entre chaque tour, le deuxième et le troisième mot sont échangés. A chaque tour, la suite d'opérations est la suivante :

(1) Multiplication de X_1 par le premier sous bloc de la clé ;
(2) Addition de X_2 et du deuxième sous bloc de la clé ;
(3) Addition de X_3 et du troisième sous bloc de la clé ;
(4) Multiplication de X_4 par le quatrième sous bloc de la clé ;
(5) Combinaison par xor des résultats de (1) et (3) ;
(6) Combinaison par xor des résultats de (2) et (4) ;
(7) Multiplication du résultat de (5) avec le cinquième sous bloc de la clé ;
(8) Addition des résultats de (6) et (7) ;
(9) Multiplication du résultat (8) et du sixième sous bloc de la clé ;

(10) Addition des résultats (7) et (9) ;

(11) Combinaison par xor des résultats de (1) et (9) ;

(12) Combinaison par xor des résultats de (3) et (9) ;

(13) Combinaison par xor des résultats de (2) et (10) ;

(14) Combinaison par xor des résultats de (4) et (10) ;

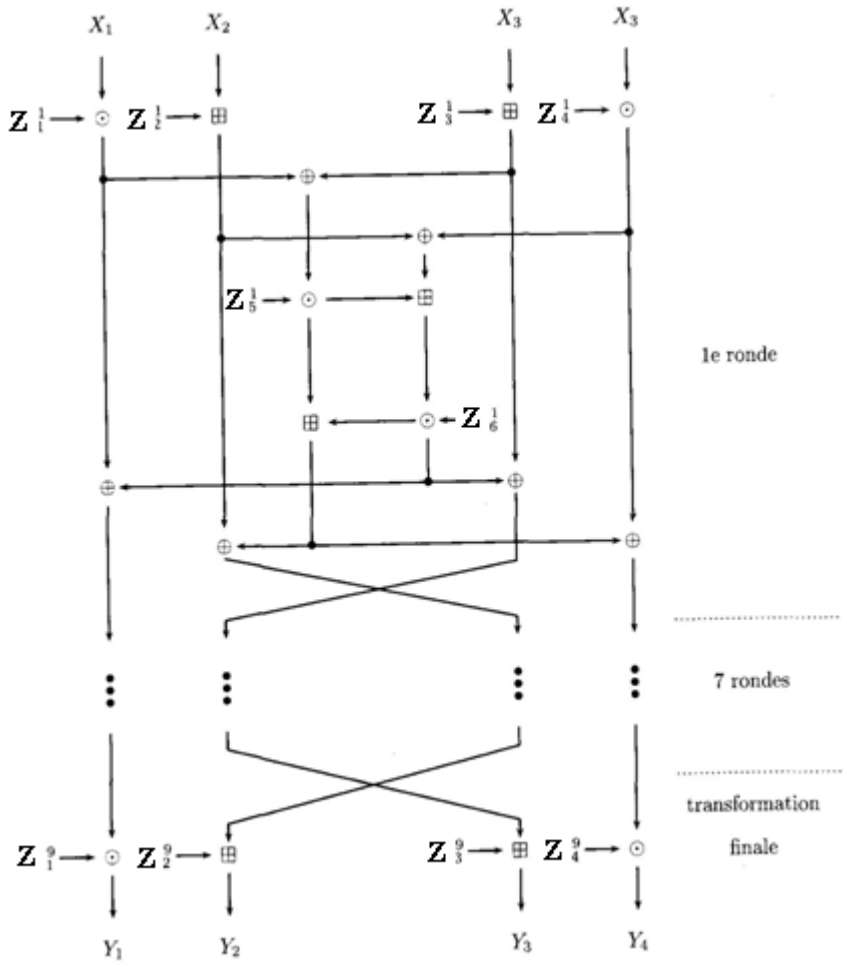

Fig.2.7- Le chiffrement IDEA

La sortie du premier tour est constituée de 4 sous blocs produits par les étapes (11), (12), (13), et (14). On échange les deux mots internes – sauf lors du dernier tour – et on obtient l'entrée du tour suivant. Après le huitième tour, on applique une transformation finale :

(1) Multiplication de X_1 et du premier sous bloc de la clé ;

(2) Addition de X_2 et du deuxième sous bloc de la clé ;

(3) Addition de X_3 et du troisième sous bloc de la clé ;

(4) Multiplication de X_4 par le quatrième sous bloc de la clé ;

Enfin, les 4 mots sont ré-assemblés pour former le cryptogramme $Y_1\ Y_2\ Y_3\ Y_4$.

7.1.2.2 Génération des sous-blocs de la clé

L'algorithme de génération des sous blocs de la clé consiste à générer 52 sous blocs de 16 bits à partir de la clé initiale de 128 bits (6 pour chacune des 8 tours et 4 pour la transformation finale).La clé de 128 bits est découpée en 8 sous-clés de 16 bits. Ce sont les 8 premières sous-clés utilisées par l'algorithme (les 6 du premier tour et les 2 premières du deuxième tour). Ensuite, la clé subit une permutation circulaire de 25 bits vers la gauche et est à nouveau divisée en 8 sous-clés de 16 bits. Les 4 premières sont utilisées par le $2^{ém}$ tour et les 4suivants par le $3^{ém}$ tour. La clé subit à nouveau une permutation circulaire de 25 bits vers la gauche pour les 8 sous-clés suivantes, et ainsi de suite jusqu'à la fin de l'algorithme.

7.1.2.3 Le Déchiffrement

Le déchiffrement suit le même fonctionnement hormis pour la création des sous-blocs de la clé de déchiffrement. Les sous-blocs doivent être l'inverse de la clé de chiffrement pour l'addition et pour la multiplication. Le tableau Tab.2.11 indique les sous-blocs de la clé de chiffrement puis les sous-blocs de clé de déchiffrement correspondants [21].

Tour	Sous-blocs de chiffrement	Sous-blocs de chiffrement
1	$Z_1\ Z_2\ Z_3\ Z_4\ Z_5\ Z_6$	$Z_{49}^{-1} - Z_{50} - Z_{51} Z_{52}^{-1} Z_{47} Z_{48}$
2	$Z_7\ Z_8\ Z_9\ Z_{10}\ Z_{11}\ Z_{12}$	$Z_{43}^{-1} - Z_{45} - Z_{44} Z_{46}^{-1} Z_{41} Z_{42}$
3	$Z_{13}\ Z_{14}\ Z_{15}\ Z_{16}\ Z_{17}\ Z_{18}$	$Z_{37}^{-1} - Z_{39} - Z_{38} Z_{40}^{-1} Z_{35} Z_{36}$
4	$Z_{19}\ Z_{20}\ Z_{21}\ Z_{22}\ Z_{23}\ Z_{24}$	$Z_{31}^{-1} - Z_{33} - Z_{32} Z_{34}^{-1} Z_{29} Z_{30}$
5	$Z_{25}\ Z_{26}\ Z_{27}\ Z_{28}\ Z_{29}\ Z_{30}$	$Z_{25}^{-1} - Z_{27} - Z_{26} Z_{28}^{-1} Z_{23} Z_{24}$
6	$Z_{31}\ Z_{32}\ Z_{33}\ Z_{34}\ Z_{35}\ Z_{36}$	$Z_{19}^{-1} - Z_{21} - Z_{20} Z_{22}^{-1} Z_{17} Z_{18}$
7	$Z_{37}\ Z_{38}\ Z_{39}\ Z_{40}\ Z_{41}\ Z_{42}$	$Z_{13}^{-1} - Z_{15} - Z_{14} Z_{16}^{-1} Z_{11} Z_{12}$
8	$Z_{43}\ Z_{44}\ Z_{45}\ Z_{46}\ Z_{47}\ Z_{48}$	$Z_7^{-1} - Z_9 - Z_8 Z_{10}^{-1} Z_5 Z_6$
9	$Z_{49}\ Z_{50}\ Z_{51}\ Z_{52}$	$Z_1^{-1} - Z_2 - Z_3 Z_4^{-1}$

Tab.2.11 - Les sous clés de déchiffrement

7.1.3 AES (*Advanced Encryption System*)

L'algorithme choisit par NIST (National Institute of Standards and Technology) afin de remplacer le DES pour devenir l'AES est le Rijndael, du nom condensé de ses concepteurs, Rijmen et Daemen. C'est un système de chiffrement symétrique proposé, dit aussi système de chiffrement ''par blocs'' car les messages sont chiffrés par blocs de 128 bits. Selon la version du système utilisé. Ce dernier utilise des clés de 128, 192 ou 256 [22].

Le principe de fonctionnement de l'AES est décrit dans la fig.2.8. En premier lieu, on ajoute bit à bit le message avec la clé secrète K_0. Puis, comme pour tous les algorithmes de chiffrement par blocs, on itère une fonction F, paramétrée par des sous-clés qui sont obtenues de la clé maître par un algorithme de cadencement de clés.

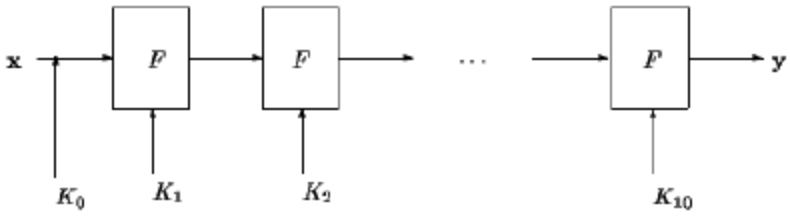

Fig.2.8- Itérations de L'AES

Dans le cas d'AES de Rijndael, on procède par blocs de 128 bits, avec une clé de 128 bits également. Chaque bloc subit une séquence de 5 transformations répétées 10 fois :

- Addition de la clé secrète (par un ou exclusif).

- Transformation non linéaire d'octets : les 128 bits sont repartis en 16 blocs de 8 bits (8 bits=un octet), eux même dispatchés dans un tableau 4×4. Chaque octet est transformé par une fonction non linéaire S.

- Décalage de lignes : les 3 dernières lignes sont décalées cycliquement vers la gauche : la 2eme ligne est décalée d'une colonne, la 3eme ligne de 2 colonnes, et la 4eme ligne de 3 colonnes.

- Brouillage des colonnes : Chaque colonne est transformée par combinaisons linéaires des différents éléments de la colonne (ce qui revient à multiplier la matrice 4×4 par une autre matrice 4×4). Les calculs sur les octets de 8 bits sont réalisés dans le corps à 28 éléments.

- Addition de la clé de tour : A chaque tour, une clé de tour est générée à partir de la clé secrète par un sous-algorithme (dit de cadencement). Cette clé de tour est ajoutée par un ou exclusif au dernier bloc obtenu.

7.1.4 Problèmes de la cryptographie symétrique

L'avantage le plus important de la cryptographie à clés secrètes est de fait qu'elle est très rapide de point de vue temps de calcul. Mais dans un système à clés secrètes, le nombre de clés augmente selon le carré du nombre de correspondants à établir : n*(n-1)/2. Ceci est dû au fait qu'il est

nécessaire d'avoir une clé secrète par paire de correspondants puisque toute communication entre individus ou sites doit rester par principe secrète.

Le Tab2.12 t illustre la croissance du nombre de clés en fonction du nombre de correspondants.

Nombre de correspondants	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**15**	**20**	**50**
Nombre de clés	**1**	**3**	**6**	**10**	**15**	**2**	**28**	**36**	**45**	**105**	**190**	**1225**

Tab.2.12 - Croissance du nombre de clés en fonction du nombre de correspondants

Cette contrainte pose le problème de distributions des clés secrètes sur les correspondants ce qui est le désavantage majeur de ce type de cryptographie. D'autre part ce type de cryptographie ne permet pas de réaliser la signature électronique.

7.3 Cryptographie asymétrique

La cryptographie symétrique repose sur l'existence d'une clé privée, commune à l'émetteur et au récepteur du message, qui doit rester secrète pour garantir la confidentialité de la transaction. Cela pose le problème du transport de cette clé (comment la faire parvenir de façon sûre à quelqu'un se trouvant à l'étranger). Ce problème de distribution des clés est résolu par la cryptographie à clé publique, dont le concept fut inventé par Whitfield Diffie et Martin Helman en 1975. Dans ce type de cryptographie, on utilise deux clés : la clé publique et la clé privée, dont le prince est décrit comme suit :

- Les messages chiffrés avec une clé ne peuvent être déchiffrés qu'avec l'autre clé.
- La clé publique est calculée à partir de la clé privée mais on ne peut pas déduire la clé privée à partir de la clé publique.
- La clé privée doit toujours rester en possession et sous le strict contrôle du détenteur.
- La clé publique est transmise à des tiers pour chiffrer les messages à destination du propriétaire de la clé privée. Par analogie on peut dire que la clé publique permet de fermer un cadenas mais qu'elle ne permet pas de l'ouvrir.
- Toute personne en possession d'une copie de la clé publique peut chiffrer des informations que seul le propriétaire de la clé privée pourra les déchiffrer. Le principe est celui de la boîte aux lettres : tout le monde peut déposer des lettres dans la boîte mais seule une personne, le propriétaire, peut les en retirer.

La fig.2.9 décrit le principe de la cryptographie asymétrique. Alice veut envoyer un message secret à Bob. Donc il le chiffre avec la clé publique de Bob puis il le transmit à travers un canal, à la réception Bob récupère le message chiffré, puis il utilise sa clé privée pour le déchiffrer [20].

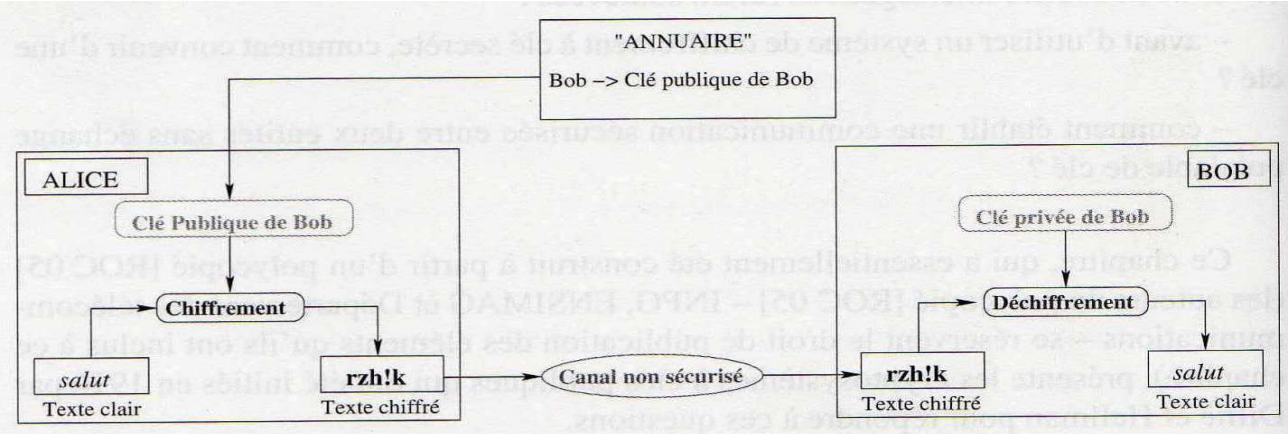

Fig.2.9- Algorithme de chiffrement asymétrique

7.3.1 RSA

L'algorithme RSA est actuellement le crypto-système le plus employé. Inventé en 1977 par R. Rivest, A. Shamir et L. Adleman. Le fonctionnement de ce crypto-système est basé sur la difficulté de factoriser le produit de grands nombres entiers. Le concept de grands nombres entiers varie selon l'époque et selon l'importance du secret à protéger. Aujourd'hui on peut considérer comme sûre les clés de 1024 bits pour des applications commerciales.

7.3.1.1 Fonction indicatrice d'Euler

Soit n un entier naturel. On appelle fonction indicatrice d'Euler, et on note $\varphi(n)$ le nombre d'entiers naturels compris entre 1 et n qui sont positifs premiers avec n [20].

$$\varphi(n) = card\{j \in \{1, \dots, n\} : pgcd(j, n) = 1\}$$

On a $\varphi(1) = 1$ et pour un entier premier p, $\varphi(p) = p - 1$.

Une méthode de calcul de $\varphi(n)$ est de décomposer n en produit de facteurs premiers en utilisant la généralisation du théorème de Fermat par Euler.

Soit a et n deux entiers premiers entre eux alors $a^{\varphi(n)} = 1\, mod(n)$.

Donc $\varphi(n)$ peut être calculé grâce à la formule suivante :

$$\varphi(n) = n\left[\left(1 - {}^{1}/_{p_1}\right)\left(1 - {}^{1}/_{p_2}\right) \dots \left(1 - {}^{1}/_{p_m}\right)\right]$$

Où $p_1, p_2, \dots, p_m$ sont les facteurs premiers de n.

La propriété ci-dessus nous montre le rapport qui existe entre le calcul de $\varphi(n)$ et les facteurs premiers de n. Donc, déterminer $\varphi(n)$ revient à factoriser le nombre n. Or le probleme de la factorisation des très grands nombres est un problème difficile, voire impossible en un temps raisonnable. Dans le cas particulier ou $n = p.q$ avec p et q deux grands nombres premiers distincts, nous pouvons déterminer $\varphi(n)$facilement.

En effet, nous savons que $\varphi(p) =$ p $- 1$ et $\varphi(q) =$ q $- 1$ De plus, comme p et q sont premiers distincts alors p et q sont premiers entre eux alors on à :

$\varphi(n) = \varphi(p.q) = \varphi(p).\varphi(q) =$ (p $- 1$)(q $- 1$) .

Mais si on ne connait pas p et q alors il est difficile de factoriser n afin d'obtenir $\varphi(n)$.

C'est sur cette difficulté mathématique que repose le système RSA.

7.3.1.2 Génération d'une clé pour l'algorithme RSA

La construction d'une clé pour l'algorithme RSA se fait en trois étapes [21] :

1. Générer deux grands nombres premiers p et q et calculer $n = pq$, et $\varphi(n) =$ (p $- 1$)(q $- 1$).
2. Trouver un entier e tel que 1 < e < $\varphi(n)$ et $pgcd(e, \varphi(n)) = 1$.
3. Calculer $d = e^{-1} mod(\varphi(n))$.

La clé publique diffusée est (n, e) et la clef privée est d.

Notons que la connaissance de n et e ne permet pas de reconstituer d. Il faudrait pour cela être capable de factoriser n, ce qui est impossible dès que l'on choisit p et q suffisamment grands.

7.3.1.3 Cryptage par l'algorithme RSA

Le message *m* à transmettre doit appartenir à l'intervalle [0, $n - 1$]. On calcule alors :

$$c \equiv m^e \, mod \, (n)$$

c est le message crypté à transmettre.

7.3.1.4 Décryptage par l'algorithme RSA

Soit c le message crypté. Il suffit de calculer :

$$m \equiv c^d \, mod \, (n)$$

L'algorithme RSA est bien réversible et permet de faire deux choses essentielles :

- chiffrer un message avec une clé publique : seul le détenteur de la clé privée associée pourra alors lire le message.

- chiffrer un message avec une clé secrète : tous les possesseurs de la clé publique associée pourront alors lire le message mais auront en plus la certitude que seul le possesseur de la clé privée a pu chiffrer le message. Donc le RSA permet en outre d'établir une authenticité, c'est-à-dire qu'il gère le principe de la signature.

7.3.2 El Gamal

Le système de chiffrement d'El Gamal repose sur la difficulté calculatoire de calculer un logarithme discret, il à été proposé en 1985 avec un schéma de signature numérique associé. Si le système de chiffrement n'a pas reporté un grand succès, celui de la signature en revanche est à l'origine de plusieurs standards, comme DSS ou DSA.

7.3.2.1 Système de chiffrement El Gamal

(1) On choisit p un premier suffisamment grand pour que le problème du logarithme discret dans $Z_p$soit difficile :

(2) On choisit $\alpha \in Z_p^{\ x}$ un élément primitive ;

(3) On calcule $\beta \equiv \alpha^a mod\ (p)$, pour a un nombre conservé discret (la clé privée) ;

(4) la clé publique est alors la donnée de p, α, β .

L'opération de chiffrement est définie de manière suivante :

$$E: (x, k) \mapsto (y_1 = \alpha^k mod(p), \qquad y_2 = x\beta^k mod(p))$$

Pour k aléatoire discret de Z_{p-1}.

Le clair $x\ est$ « masqué » par la multiplication par β^ket produit y_2. La valeur de valeur de α^k est transmise en tant que partie de cryptogramme comme y_1.

L'opération de déchiffrement est :

$$(\ y_1,\ y_2) \mapsto\ y_2(\ {y_1}^a)^{-1}\ mod(p)$$

Bob, qui connait la clé privée a, calcule β^k à partir de α^k. Comme $\beta = \alpha^a$, il suffit de calculer $(\ \alpha^k)^a = \beta^k$. Il ne reste plus qu'à multiplier y_2 par l'inverse de $\beta^k\ mod\ (p)$ pour retrouver x

Observons que le cryptogramme est deux fois plus long que le clair.

7.3.3 Problèmes de la cryptographie asymétrique

L'avantage le plus important de la cryptographie asymétrique est de fait qu'elle utilise deux clés différentes. Donc le problème de distribution de clé n'apparait pas de ce type de cryptographie. D'autre part elle fournit des garanties d'intégrité et de non répudiation par signature électronique. Par contre la cryptographie asymétrique présente quelques inconvénients tels que : la quantité d'espace mémoire utilisé, et la lenteur des algorithmes tant pour la génération des clés que pour le chiffrement d'un message.

8. La signature numérique

L'un des principaux avantages de la cryptographie à clé publique est qu'elle offre une méthode d'utilisation des signatures numériques. Celle-ci permettent au destinataire de vérifier leur authentification, leur origine, mais également de s'assurer qu'elles sont intactes. Donc le but des signatures est de faire la preuve de l'identité de l'expéditeur et de l'intégrité du message [21]. Une signature dépend à la fois du contenu de message et de l'identité du signataire. Elle doit empêcher différents types de fraudes :

- La falsification, ou l'usurpation de la signature.
- La non-reconnaissance du message par l'expéditeur, dite répudiation.

Le cahier des charges d'une signature est donc le suivant :

- Elle doit être calculable par le signataire pour tout message M.
- Tout individu (et en particulier le destinataire) doit pouvoir vérifier la signature.
- Elle doit être impossible à falsifier.
- L'expéditeur ne doit pas pouvoir affirmer que sa signature à été imitée.

Nous décrivons brièvement le mécanisme utilisant cette méthode de cryptage.

8.1 Mécanisme général de la signature numérique

Un procédé de la signature est composé :

- d'un algorithme privé de signature noté sig qui, à un clair M et pour une clé fixée K, retrouve une signature S :

$$sig_K(M) = S$$

- un prédicat public de vérification noté ver qui, à une clé fixée K et pour tout couple clair/signature (M, S), va vérifier la validité de la signature S pour le clair M :

$$ver_K(M,S) = vrai \Leftrightarrow S = sig_K(M)$$

8.2 Signature par RSA

Le premier exemple de procèdes de signature est le système à clé publique RSA

Bob désire envoyer un message M signé à Alice. Ils disposent pour cela de

	privés	publics
Alice	d_A	$n_A,\ e_A$
Bob	d_B	$n_B,\ e_B$

Tab2.13 - Les couple des clés de signature RSA

Le procédé de signature est alors :

$$sig_K(M) = M^{d_B} \bmod n_B = \mathrm{S}$$

Celui de verification :

$$ver_K(M,S) = vrai \Leftrightarrow S^{\,e_B} \bmod n_B \equiv M$$

9. Fonctions de hachage

Ces fonctions, appelées aussi fonctions de « condensation », condensent en quelque sorte les informations contenues dans un fichier de taille quelconque en un grand nombre qui se révèle alors être l'empreinte du fichier [22].

Ces fonctions sont telles que : si un seul bit du message d'origine est modifié, alors l'empreinte obtenue change radicalement.

Voyons quelles sont les caractéristiques de telles fonctions de hachage :

Considérons h = H(P) avec

h : empreinte de longueur fixe

H : fonction de hachage

P : texte de longueur quelconque

Les propriétés sont les suivantes :

· connaissant P et H, il est facile de calculer h

· connaissant h et H, il est très difficile de calculer P

· connaissant P, il est très difficile de trouver P' tel que H(P) = H(P')

Les fonctions de hachage les plus connues sont MD2, MD4, MD5 (« MD » pour «Message Digest»), SHA (Secure Hash Algorithm) et SHS (Secure Hash Standard)

9.1 MD5 *"Message Digest version 5"*

Md5 Conçu par Ronald Rivest et publié en avril 1992, renvoie une valeur de 128 bits sous forme de 32 chiffres hexadécimaux quelque soit le fichier d'origine [22]. Une autre caractéristique de MD5 réside dans le fait que des données d'entrée identiques produisent toujours une même empreinte.

Md5 Calcul itérativement le résultat de 128 bits sous la forme de quatre mots de 32 bits a, b, c, d initialisés au départ à des constantes: a = 01234567, b = 89ABCDEF, c = FEDCBA98, d = 76543210.

Un message est décomposé en blocs de 512 bits soient 16 sous-blocs de 32 bits notés M[k]: un message est complété à un nombre entier de blocs de 512 bits par bourrage.

Pour chaque bloc de 512 bits on réalise un calcul à partir de la dernière valeur de a, b, c, d on obtient la nouvelle valeur a, b, c, d.

Définition des fonctions F, G, H, I : elles sont non linéaires sur 32 bits à partir de ou, et, non, ouex (ou exclusif).

- $F(X,Y,Z) = (X \bullet Y) + (\bar{X} \bullet Z)$
- $G(X,Y,Z) = (X \bullet Z) + (Y \bullet \bar{Z})$
- $H(X,Y,Z) = X \oplus Y \oplus Z$
- $I(X,Y,Z) = Y \oplus (X + \bar{Z})$

Le calcul comprend 4 rondes de 16 applications successives des fonctions FF, GG, HH, II qui dépendent des fonctions F, G, H, I, des sous-blocs M[k], des variables a,b, c, d, et de constantes i:

FF(a, b, c, d, M[k], s, i) : a = b + ((a + F(b, c, d) +M[k] + T[i] <<< s)

GG(a, b, c, d, M[k], s, i) : a = b + ((a + G(b, c, d) +M[k] + T[i] <<< s)

HH(a, b, c, d, M[k], s, i) : a = b + ((a + H(b, c, d) +M[k] + T[i] <<< s)

II(a, b, c, d, M[k], s, i) : a = b + ((a + I(b, c, d) +M[k] + T[i] <<< s)

<<< s est le décalage à gauche de s positions.

- La première ronde:

FF[a, b, c, d, M[0], 7, 1], FF[d, a, b, c, M[1], 12, 2], FF[c, d, a, b, M[2], 17, 3],

FF[b, c, d, a, M[3], 22, 4], FF[a, b, c, d, M[4], 7, 5], FF[d, a, b, c, M[5], 12, 6],

FF[c, d, a, b, M[6], 17, 7], FF[b, c, d, a, M[7], 22, 8], FF[a, b, c, d, M[8], 7, 9],

FF[d, a, b, c, M[9], 12, 10], FF[c, d, a, b, M[10], 17, 11], FF[b, c, d, a, M[11], 22, 12],

FF[a, b, c, d, M[12], 7, 13], FF[d, a, b, c, M[13], 12, 14], FF[c, d, a, b, M[14], 17, 15],

FF[b, c, d, a, M[15], 22, 16]
Quatre rondes analogues:
FF, GG, HH, II.

10. Discussion

D'après l'étude faite dans ce chapitre, nous avons constaté que la cryptographie est un outil très important dans la transmission sécurisée de faite qu'elle offre de la sûreté de l'information grâce aux algorithmes de cryptages symétriques ou asymétriques. Par contre on a vu que chacun des deux porte des inconvénients qui peuvent poser des problèmes dans la transmission sécurisée, notamment le premier algorithme pose le problème de partage des clés secrets entres les communicants, tandis que l'algorithme à clé publique a un grave défaut : il est lent, beaucoup plus lent que son homologue symétrique. Donc pour des applications où il faut échanger de nombreuses données, ils sont inutilisables en pratique. On a alors recours à des crypto-systèmes hybrides. On échange des clés pour un chiffrement symétrique grâce à la cryptographie à clé publique, ce qui permet de sécuriser la communication de la clé. On utilise ensuite un algorithme de chiffrement symétrique. C'est le but de notre prochain chapitre où nous allons étudier et concevoir un tel système de cryptographie hybride tel que le PGP.

CHAPITRE 3

Système de Transmission Sécurisée à base de PGP

1. Introduction

Lorsqu'on envoie un courrier papier par la poste, la confidentialité du pli est assurée. Ce n'est malheureusement pas le cas si on utilise le courrier électronique. Chaque message peut être intercepté par un tiers qui dispose d'un ordinateur connecté au réseau. Ce tiers peut alors impunément lire le message et même en modifier le contenu. Ce risque peut être limité en faisant usage d'outils cryptographiques tels que les cryptographies symétriques ou asymétriques, cependant celles-ci pose des problèmes des partage des clés secrets pour la cryptographie symétrique, et un temps de calcul important pour la cryptographie asymétrique. Pour répondre à ces contraintes nous allons développer et détailler dans ce chapitre un outil qui utilise une technique de chiffrement hybride basée sur le chiffrement symétrique et le chiffrement asymétrique pour sécuriser des courriers électroniques : PGP, qui est l'acronyme de *pretty good privacy* où Nous allons détailler chaque bloque de son schéma synoptique.

2. Fonctionnement du PGP

Le PGP a été mis au point en 1991 par l'informaticien américain Philip Zimmermann [20], c'est un système hybride que l'on peut classer dans les systèmes "à clé de session", c'est-à-dire un système qui utilise à la fois le principe du chiffrement à clé privée et le principe du chiffrement à clé publique.

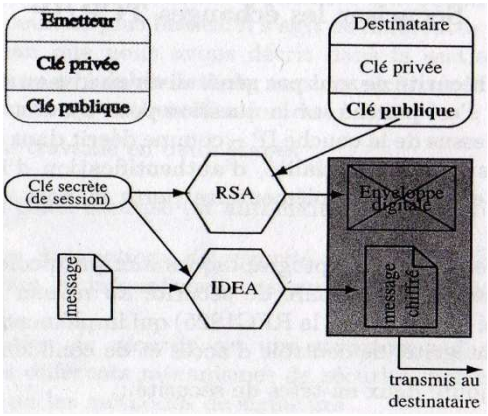

Fig.3.1 - Fonctionnement du PGP

Considérons les différentes étapes du transfert d'un message crypté avec PGP de l'expéditeur X vers le destinataire Y.

(i) X doit envoyer le message crypté à Y.

(ii) Y crée une paire de clé via l'algorithme RSA. Il transmet sa clé publique à X.

(iii) X saisit le texte en clair à envoyer. Ce texte est tout d'abord compressé ce qui offre un double avantage :

- la taille des données à transférer est réduite,
- les risques de décryptage sont minimisés (la plupart des techniques de cryptanalyse se base sur le texte en clair obtenu. Si le texte obtenu est un texte compressé il est, par exemple, plus difficile de calculer la probabilité de retrouver telle ou telle lettre).

Il faut noter que la compression n'est pas systématique. Si le taux de compression d'un fichier n'est pas satisfaisant ou si le fichier est trop petit, cette étape n'est pas réalisée.

(iv) Puis, X crée aléatoirement une clé secrète IDEA. L'expéditeur chiffre le texte avec cette clé IDEA. Le texte ainsi chiffré pourra être déchiffré avec la même clé. Dans PGP, le message est alors crypté selon un système symétrique (à clé secrète).

(v) Le destinataire Y ne connaissant pas cette clé, elle va lui être envoyée avec le message crypté. Toutefois pour éviter qu'elle soit interceptée, la clé sera également cryptée à l'aide de la clé publique de Y. La clé privée IDEA est cryptée avec la clé publique de Y selon un système asymétrique (à clé publique).

Finalement, le résultat obtenu contient :

- le texte chiffré avec la clé IDEA,
- la clé IDEA chiffrée avec la clef publique RSA du destinataire.

A la réception du message, le destinataire utilise sa clé privée RSA pour retrouver la valeur de la clé IDEA. Il utilise la clé obtenue pour déchiffrer le message reçu [21].

2.1. L'algorithme IDEA

Supposons que la clé de session est générée (nous allons voir cette opération dans le cas pratique), nous allons détailler cette algorithme bloc par bloc avec toutes les étapes qui rentrent en jeu, et pour mieux comprendre nous prenons des exemples d'applications.

Comme on a vu dans le chapitre précédent, IDEA est un système de chiffrement par blocs de 64 bits, avec une clé de 128 bits, qui tourne sur 8 tours.

2.1.1. Génération de séquence ment de clé

L'algorithme IDEA utilise 52 sous-clés de 16-bit générées à partir de la clé Z de 128 bits, six sous-clés pour chaque les premiers huit tours, et quatre du neuvième tour pour la sortie de la transformation finale. La clé de cryptage de 128 bits est découpée en huit sous-clés de 16 bits, et composant ainsi les huit premières sous-clés s'appellent $Z_1, Z_2, \dots Z_8$ avec Z_1 prend les seize

premières bits de Z, et Z_2 prend les seize suivant, et ainsi de suite. Les six premières sous-clés sont utilisées pour le 1er tour, et les deux restantes sont utilisées comme étant les deux premières de 2éme tour. Ensuite la clé subit une permutation circulaire de 25 bits vers la gauche et est à nouveau divisée en huit sous-clés de seize bits. Les quatre premiers sont utilisées par le 2éme tour et les quatre suivantes par le 3éme tour. La clé subit à nouveau une permutation circulaire de 25 bits vers la gauche pour les huit sous-clés suivantes, et ainsi de suite jusqu'à la fin de l'algorithme. La Fig.3.2 illustre les différentes sous-clés qui rentrent en jeu dans les différents tours de l'algorithme IDEA qu'elle soit, de chiffrement ou de déchiffrement.

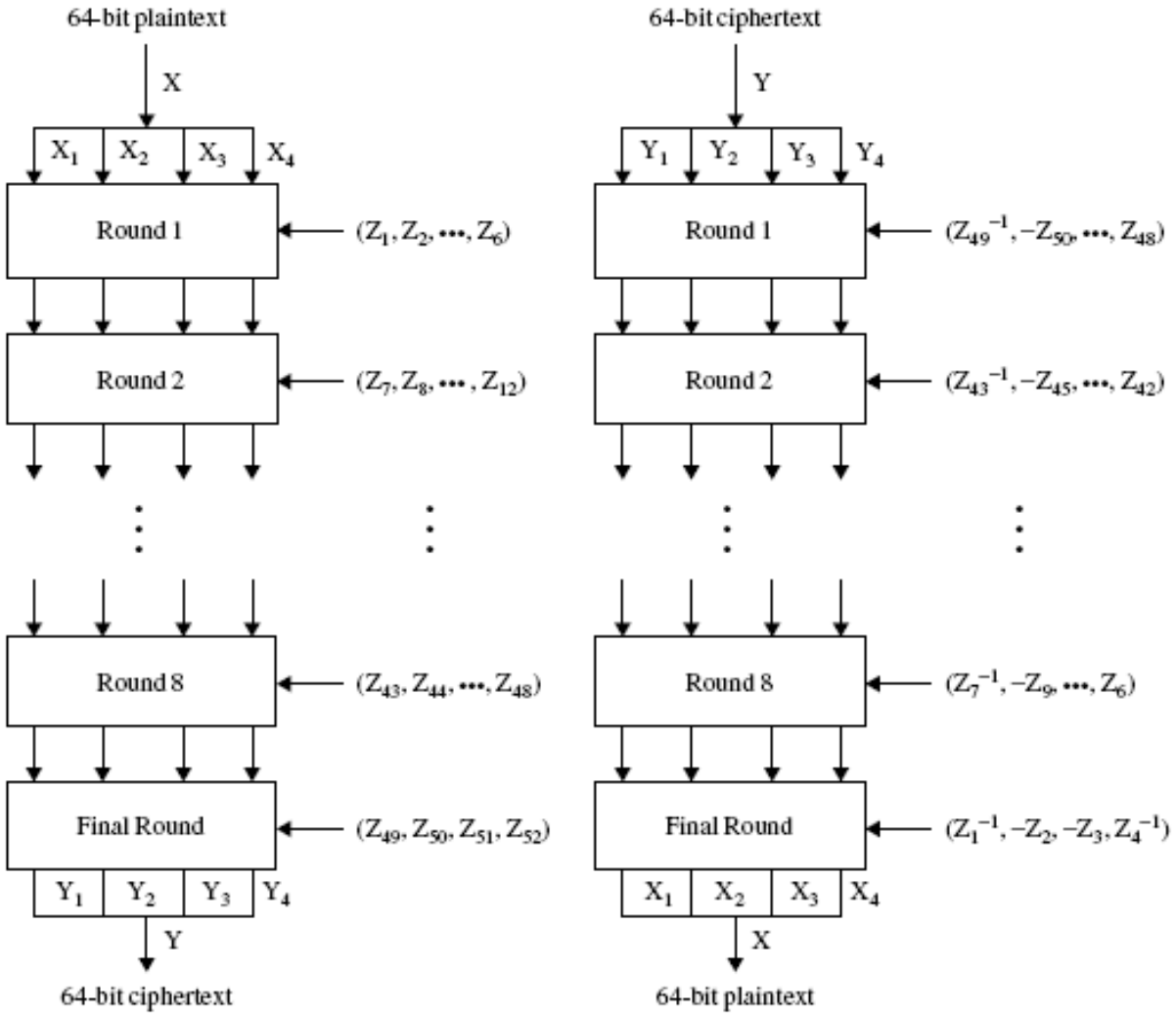

Fig.3.2 – Le diagramme du bloc de cryptage / décryptage IDEA

Si la clé originale notée $Z(1,2,\ldots,128)$, alors les blocs des sous-clés des huit tours qui sont générés à partir Z, de la manière décrite dans le Tab.3.1

Round 1	
$Z_1 = Z(1, 2, \ldots, 16)$	$Z_4 = Z(49, 50, \ldots, 64)$
$Z_2 = Z(17, 18, \ldots, 32)$	$Z_5 = Z(65, 66, \ldots, 80)$
$Z_3 = Z(33, 34, \ldots, 48)$	$Z_6 = Z(81, 82, \ldots, 96)$
Round 2	
$Z_7 = Z(97, 98, \ldots, 112)$	$Z_{10} = Z(42, 43, \ldots, 57)$
$Z_8 = Z(113, 114, \ldots, 128)$	$Z_{11} = Z(58, 59, \ldots, 73)$
$Z_9 = Z(26, 27, \ldots, 41)$	$Z_{12} = Z(74, 75, \ldots, 89)$
Round 3	
$Z_{13} = Z(90, 91, \ldots, 105)$	$Z_{16} = Z(10, 11, \ldots, 25)$
$Z_{14} = Z(106, 107, \ldots, 121)$	$Z_{17} = Z(51, 52, \ldots, 66)$
$Z_{15} = Z(122, 123, \ldots, 128, 1, 2, \ldots, 9)$	$Z_{18} = Z(67, 68, \ldots, 82)$
Round 4	
$Z_{19} = Z(83, 84, \ldots, 98)$	$Z_{22} = Z(3, 4, \ldots, 18)$
$Z_{20} = Z(99, 100, \ldots, 114)$	$Z_{23} = Z(19, 20, \ldots, 34)$
$Z_{21} = Z(115, 116, \ldots, 128, 1, 2)$	$Z_{24} = Z(35, 36, \ldots, 50)$
Round 5	
$Z_{25} = Z(76, 77, \ldots, 91)$	$Z_{28} = Z(124, 125, \ldots, 128, 1, 2, \ldots, 11)$
$Z_{26} = Z(92, 93, \ldots, 107)$	$Z_{29} = Z(12, 13, \ldots, 27)$
$Z_{27} = Z(108, 109, \ldots, 123)$	$Z_{30} = Z(28, 29, \ldots, 43)$
Round 6	
$Z_{31} = Z(44, 45, \ldots, 59)$	$Z_{34} = Z(117, 118, \ldots, 128, 1, 2, 3, 4)$
$Z_{32} = Z(60, 61, \ldots, 75)$	$Z_{35} = Z(5, 6, \ldots, 20)$
$Z_{33} = Z(101, 102, \ldots, 115)$	$Z_{36} = Z(21, 22, \ldots, 36)$
Round 7	
$Z_{37} = Z(37, 38, \ldots, 52)$	$Z_{40} = Z(85, 86, \ldots, 100)$
$Z_{38} = Z(53, 54, \ldots, 68)$	$Z_{41} = Z(126, 127, 128, \ldots, 1, 2, \ldots, 13)$
$Z_{39} = Z(69, 70, \ldots, 84)$	$Z_{42} = Z(14, 15, \ldots, 29)$
Round 8	
$Z_{43} = Z(30, 31, \ldots, 45)$	$Z_{46} = Z(78, 79, \ldots, 93)$
$Z_{44} = Z(46, 47, \ldots, 61)$	$Z_{47} = Z(94, 95, \ldots, 109)$
$Z_{45} = Z(62, 63, \ldots, 77)$	$Z_{48} = Z(110, 111, \ldots, 125)$
Round 9 (final transformation stage)	
$Z_{49} = Z(23, 24, \ldots, 38)$	$Z_{51} = Z(55, 56, \ldots, 70)$
$Z_{50} = Z(39, 40, \ldots, 54)$	$Z_{52} = Z(71, 72, \ldots, 86)$

Tab.3.1 - La génération des sous-clés de cryptage IDEA

Exemple. 2.1.1 : Supposons que la clé originale de 128 bits Z est donnée tel que :

Z = (5a14 fb3e 021c 79e0 6081 46a0 117b ff03).

Les 52 sous-clés de 16-bits générées à partir de la clé Z comme suit : pour le 1er tour, les huit premières sous-clés sont découpées directement à partir de Z. Ensuite la clé Z subit une permutation circulaire vers la gauche de 25 bits, et à nouveau découpée en huit sous-clés de 16-bits. Cette opération est répétée jusqu'à la génération des 52 sous-clés. Le résultat est donné dans le Tab.3.2

$Z_1 = 5a14$	$Z_{27} = dff8$
$Z_2 = fb3e$	$Z_{28} = 1ad0$
$Z_3 = 021c$	$Z_{29} = a7d9$
$Z_4 = 79e0$	$Z_{30} = f010$
$Z_5 = 6081$	$Z_{31} = e3cf$
$Z_6 = 46a0$	$Z_{32} = 0304$
$Z_7 = 117b$	$Z_{33} = 17bf$
$Z_8 = ff03$	$Z_{34} = f035$
$Z_9 = 7c04$	$Z_{35} = a14f$
$Z_{10} = 38f3$	$Z_{36} = b3e0$
$Z_{11} = c0c1$	$Z_{37} = 21c7$
$Z_{12} = 028d$	$Z_{38} = 9e06$
$Z_{13} = 4022$	$Z_{39} = 0814$
$Z_{14} = f7fe$	$Z_{40} = 6a01$
$Z_{15} = 06b4$	$Z_{41} = 6b42$
$Z_{16} = 29f6$	$Z_{42} = 9f67$
$Z_{17} = e781$	$Z_{43} = c043$
$Z_{18} = 8205$	$Z_{44} = 8f3c$
$Z_{19} = 1a80$	$Z_{45} = 0c10$
$Z_{20} = 45ef$	$Z_{46} = 28d4$
$Z_{21} = fc0d$	$Z_{47} = 022f$
$Z_{22} = 6853$	$Z_{48} = 7fe0$
$Z_{23} = ecf8$	$Z_{49} = cf80$
$Z_{24} = 0871$	$Z_{50} = 871e$
$Z_{25} = 0a35$	$Z_{51} = 7818$
$Z_{26} = 008b$	$Z_{52} = 2051$

Tab.3.2 - Les sous-clés de cryptage IDEA

2.1.2. Cryptage IDEA

L'algorithme de cryptage IDEA est donné par la Fig.3.3. Comme tout les blocs de cryptage, il ya deux entrées : le texte clair qui découpé en bloque de 64 bits, et la clé de cryptage de 128 bits. IDEA est basé sur la combinaison des trois différentes opérations : le XOR notée $\oplus$, l'addition modulo 2^{16} notée $\boxplus$ et la multiplication modulo $2^{16}+1$ notée $\odot$.

Le bloc de donnée est découpé en 4 mots de 16 bits : X_1, X_2, X_3, X_4 qui constituent les entrées de l'algorithme IDEA, celui-ci est divisé en 8 tours. A chaque tour, les 4 mots sont combinés par des xor, additionnés, multipliés entre eux et avec les 6 mots de 16 bits de la clé. Entre chaque tour, le deuxième et le troisième mot sont échangés. A chaque tour la suite d'opération est la suivante :

(1)- $X_1 \odot Z_1$

(2)- $X_2 \boxplus Z_2$

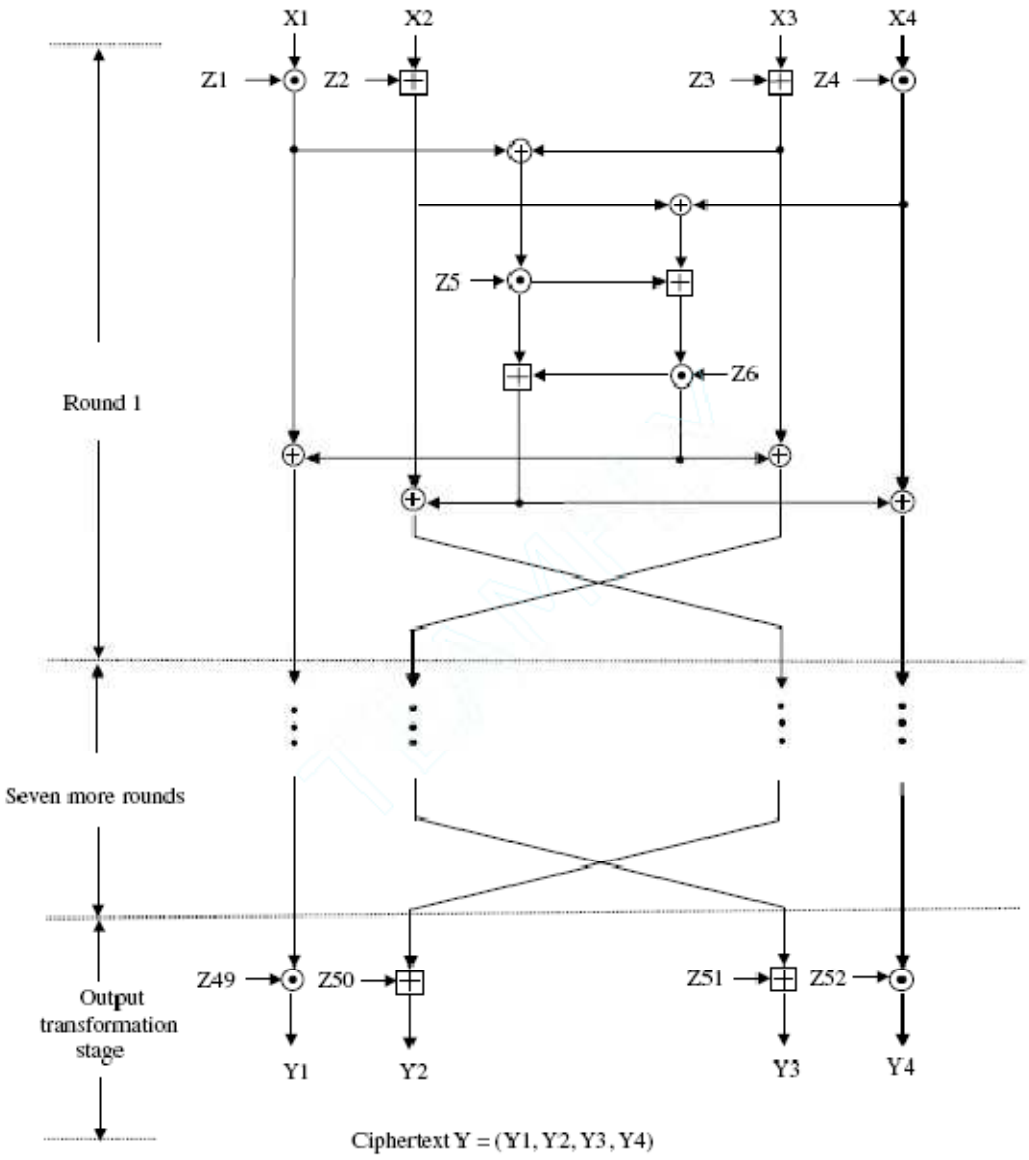

Fig.3.3 - Algorithme de cryptage IDEA

(3)- $X_3 \boxplus Z_3$

(4)- $X_4 \odot Z_4$

(5)- $(X_1 \odot Z_1) \oplus (X_3 \boxplus Z_3) = (1) \oplus (3)$

(6)- $(X_2 \boxplus Z_2) \oplus (X_4 \odot Z_4) = (2) \oplus (4)$

(7)- $(X_1 \odot Z_1) \oplus (X_3 \boxplus Z_3) \odot Z_5 = \big((1) \oplus (3)\big) \odot Z_5$

(8)- $\Big(\big((X_2 \boxplus Z_2) \oplus (X_4 \odot Z_4)\big) \boxplus \Big(\big((X_1 \odot Z_1) \oplus (X_3 \boxplus Z_3)\big) \odot Z_5\Big)\Big) =$

$\big((2) \oplus (4)\big) \boxplus \Big(\big((1) \oplus (3)\big) \odot Z_5\Big)$

(9)- $(8) \odot Z_6$

(10)- $(7) \boxplus (9) = \big(((1) \oplus (3)) \odot Z_5\big) \boxplus ((8) \odot Z_6)$

(11)- $\big((X_1 \odot Z_1) \oplus ((8) \odot Z_6)\big) = (1) \oplus (9)$

(12)- $\big((X_3 \boxplus Z_3) \oplus ((8) \odot Z_6)\big) = (3) \oplus (9)$

(13)- $\big((X_2 \boxplus Z_2) \oplus (10)\big) = (2) \oplus (10)$

(14)- $\big((X_4 \odot Z_4) \oplus (10)\big) = (4) \oplus (10)$

La sortie de chaque tour est constituée de 4 sous blocs produits par les étapes (11), (12), (13), et (14). On échange les deux mots internes « sauf lors de dernier tour » et on obtient l'entrée du tour suivant. Après le huitième tour, on applique une transformation finale :

(1)- $\overline{X_1} \odot Z_{49}$

(2)- $\overline{X_2} \boxplus Z_{50}$

(3) $\overline{X_3} \boxplus Z_{51}$

(4)- $\overline{X_4} \odot Z_{52}$

Où les $\overline{X_i}$,$1 \leq i \leq 4$ représente les sorties du 8^{ém} tour. Enfin les 4 mots sont réassemblés pour former le cryptogramme Y_1, Y_2, Y_3, Y_4.

Exemple. 2.1.2 : Supposons que le texte clair à chiffrer X de 64 bits est donné comme suit :

$$X = (X_1\ X_2\ X_3\ X_4) = (7\text{fa9 } 1\text{c37 ffb3 df05})$$

Dans l'algorithme de cryptage IDEA, le texte clair est de taille de 64 bits et la clé de cryptage est constituée de 52 sous-clés comme c'est fait dans l'exemple 2.1.1.

Comme la montre la Fig.3.3, les quatre sous blocs de 16-bits X_1, X_2, X_3, et X_4 sont XORés, additionnés, et multipliés avec des autres six sous-clés de 16-bits. Suivant l'opération séquentiel commençantes du 1er tour jusqu'à la transformation finale. Après l'exécution le message chiffré $Y = (\ Y_1, Y_2, Y_3, Y_4)$, et les résultats des sorties de chaque tour sont données par le Tab.3.3.

Ainsi le message final chiffré à partir de cet algorithme est le suivant :

$$Y = (\ Y_1, Y_2, Y_3, Y_4) = (106\text{b dbfd f323 0876})$$

Plaintext Input X

7fa9	1c37	ffb3	df05
(X_1)	(X_2)	(X_3)	(X_4)

Round	Round output				
1	C579	F2ff	0fbd	0ffc	
2	D7a2	80cb	9a61	27c5	
3	ab6c	e2f9	f3be	36bd	
4	ef5b	9cd2	6808	3019	
5	7e09	2445	d223	d639	
6	4a6e	d7ac	ac8c	8b09	
7	244d	6f5c	4459	3a9c	
8	0f86	7b0b	54df	759f	
9 (final transformation)	106b (Y_1)	dbfd (Y_2)	f323 (Y_3)	0876 (Y_4)	← Ciphertext Y

Tab.3.3 - Les différentes étapes de calcule de cryptage IDEA

2.1.3. Décryptage IDEA

Le déchiffrement suit le même fonctionnement hormis pour la création des sous-blocs de la clé de déchiffrement. Les sous-blocs doivent être l'inverse de la clé de chiffrement pour l'addition et pour la multiplication, tel que $-Z_i$ dénote l'inverse additive modulo 2^{16} de Z_i, où la propriété $-Z_i \boxplus Zi=0$ est satisfaite, et d'autre part $Zi-1$ dénote l'inverse multiplicative modulo 2^{16}+1 de Zi, où la propriété $Z_i^{-1} \odot Z_i = 1$ est satisfaite.

Donc les sous-blocs de la clé de déchiffrement sont dérivés à partir des sous-blocs de la clé de chiffrement comme le montre le Tab.3.4.

Round	Encryption subkeys	Decryption subkeys
1	$Z_1\ Z_2\ Z_3\ Z_4\ Z_5\ Z_6$	$Z_{49}^{-1} - Z_{50} - Z_{51}\ Z_{52}^{-1}\ Z_{47}\ Z_{48}$
2	$Z_7\ Z_8\ Z_9\ Z_{10}\ Z_{11}\ Z_{12}$	$Z_{43}^{-1} - Z_{45} - Z_{44}\ Z_{46}^{-1}\ Z_{41}\ Z_{42}$
3	$Z_{13}\ Z_{14}\ Z_{15}\ Z_{16}\ Z_{17}\ Z_{18}$	$Z_{37}^{-1} - Z_{39} - Z_{38}\ Z_{40}^{-1}\ Z_{35}\ Z_{36}$
4	$Z_{19}\ Z_{20}\ Z_{21}\ Z_{22}\ Z_{23}\ Z_{24}$	$Z_{31}^{-1} - Z_{33} - Z_{32}\ Z_{34}^{-1}\ Z_{29}\ Z_{30}$
5	$Z_{25}\ Z_{26}\ Z_{27}\ Z_{28}\ Z_{29}\ Z_{30}$	$Z_{25}^{-1} - Z_{27} - Z_{26}\ Z_{28}^{-1}\ Z_{23}\ Z_{24}$
6	$Z_{31}\ Z_{32}\ Z_{33}\ Z_{34}\ Z_{35}\ Z_{36}$	$Z_{19}^{-1} - Z_{21} - Z_{20}\ Z_{22}^{-1}\ Z_{17}\ Z_{18}$
7	$Z_{37}\ Z_{38}\ Z_{39}\ Z_{40}\ Z_{41}\ Z_{42}$	$Z_{13}^{-1} - Z_{15} - Z_{14}\ Z_{16}^{-1}\ Z_{11}\ Z_{12}$
8	$Z_{43}\ Z_{44}\ Z_{45}\ Z_{46}\ Z_{47}\ Z_{48}$	$Z_7^{-1} - Z_9 - Z_8\ Z_{10}^{-1}\ Z_5\ Z_6$
9	$Z_{49}\ Z_{50}\ Z_{51}\ Z_{52}$	$Z_1^{-1} - Z_2 - Z_3\ Z_4^{-1}$

Tab.3.4 - Les sous-clés de cryptage et décryptage IDEA

Regardons à ce tableau, nous voyons que les premiers quatre sous-clé de déchiffrement au niveau d'un tour *i* sont dérivées à partir des premiers quatre sous-clé de chiffrement au niveau de tour de

cryptage (10-i), où la sortie de la transformation finale est considérée comme étant le tour 9. Par exemple les premiers quatre sous-clé de déchiffrement au niveau du tour N°=2 sont dérivées à partir des premiers quatre sous-clé de chiffrement au niveau de tour de cryptage N°=8.comme le montre le Tab.3.5.

Round i	First four decryption subkeys at i	Round $(10-i)$	First four encryption subkeys at $(10-i)$
1	$Z_{49}^{-1} - Z_{50} - Z_{51} Z_{52}^{-1}$	9	Z_{49} Z_{50} Z_{51} Z_{52}
2	$Z_{43}^{-1} - Z_{45} - Z_{44} Z_{46}^{-1}$	8	Z_{43} Z_{44} Z_{45} Z_{46}
.		.	
.		.	
.		.	
8	$Z_7^{-1} - Z_9 - Z_8 Z_{10}^{-1}$	2	Z_7 Z_8 Z_9 Z_{10}
9	$Z_1^{-1} - Z_2 - Z_3 Z_4^{-1}$	1	Z_1 Z_2 Z_3 Z_4

Tab.3.5 - Les sous-clés de décryptage dérivées à partir des sous-clés de cryptage

Table 3.14 Subkey blocks for decryption

$Z_{49}^{-1} = 9194$	$-Z_{26} = ff75$
$-Z_{50} = 78e2$	$Z_{28}^{-1} = 24f6$
$-Z_{51} = 87e8$	$Z_{23} = ecf8$
$Z_{52}^{-1} = 712a$	$Z_{24} = 0871$
$Z_{47} = 022f$	$Z_{19}^{-1} = 4396$
$Z_{48} = 7fe0$	$-Z_{21} = 03f3$
$Z_{43}^{-1} = a24c$	$-Z_{20} = ba11$
$-Z_{45} = f3f0$	$Z_{22}^{-1} = dfa7$
$-Z_{44} = 70c4$	$Z_{17} = e781$
$Z_{46}^{-1} = 3305$	$Z_{18} = 8205$
$Z_{41} = 6b42$	$Z_{13}^{-1} = 18a7$
$Z_{42} = 9f67$	$-Z_{15} = f94c$
$Z_{37}^{-1} = c579$	$-Z_{14} = 0802$
$-Z_{39} = f7ec$	$Z_{16}^{-1} = 9a13$
$-Z_{38} = 61fa$	$Z_{11} = c0c1$
$Z_{40}^{-1} = bf28$	$Z_{12} = 028d$
$Z_{35} = a14f$	$Z_7^{-1} = 55ed$
$Z_{36} = b3e0$	$-Z_9 = 83fc$
$Z_{31}^{-1} = c53c$	$-Z_8 = 00fd$
$-Z_{33} = e841$	$Z_{10}^{-1} = 2cd9$
$-Z_{32} = fcfc$	$Z_5 = 6081$
$Z_{34}^{-1} = 3703$	$Z_6 = 46a0$
$Z_{29} = a7d9$	$Z_1^{-1} = 0dd8$
$Z_{30} = f010$	$-Z_2 = 04c2$
$Z_{25}^{-1} = cc14$	$-Z_3 = fde4$
$-Z_{27} = 2008$	$Z_4^{-1} = 4fd0$

Tab.3.6 - Les sous-clés de décryptage IDEA

Notons que le 1er et le 4^{ém} de la sous clé de déchiffrement sont égaux à l'inverse multiplicative modulo $2^{16}+1$ correspondants aux 1er et 4^{ém} de la sous clé de chiffrement. De tour N°=2 jusqu'au tour N°=8, le 2^{em} et le 3^{ém} tour de la sous clé de déchiffrement sont égaux à l'inverse additive

modulo 2^{16} correspondants aux 3^{ém} et 2^{ém} de la sous clé de chiffrement. Pour le tour N°=1 et N°=9, le 2^{em} et le 3^{ém} tour de la sous clé de déchiffrement sont égaux à l'inverse additive modulo 2^{16} correspondants aux 2^{ém} et 3^{ém} de la sous clé de chiffrement. Notons aussi que, pour les premiers huit tours, les deux dernières sous-clés de déchiffrement du tour i sont égales aux deux dernières sous-clés de chiffrement du tour $(9-i)$.voir le Tab.3.4

Exemple.2.1.3 : en utilisant le Tab.3.4 trouvant les sous-clés de déchiffrement correspondant aux sous-clés de chiffrement obtenus dans Tab.3.2. Le résultat est donné par le Tab.3.6, Le schéma de décryptage IDEA est donné par la Fig.3.4

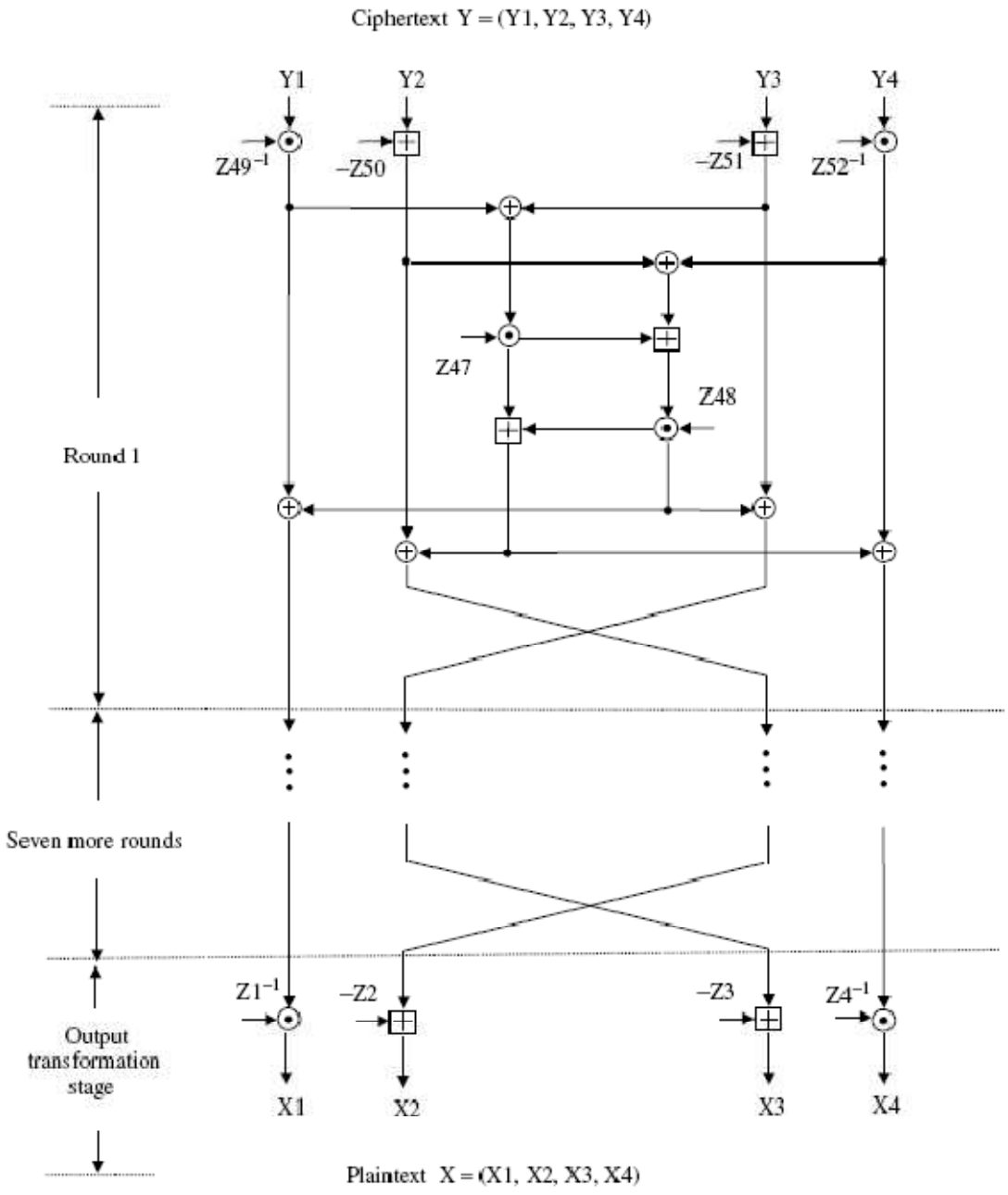

Fig.3.4 - Algorithme de décryptage IDEA

Exemple 2.1.4 : Retrouvant le texte clair $X = (X_1\ X_2\ X_3\ X_4) = (7fa9\ 1c37\ ffb3\ df05)$ en utilisant le message crypté $Y = (\ Y_1, Y_2, Y_3, Y_4) = (106b\ dbfd\ f323\ 0876)$ qui à été obtenu dans l'exemple 2.1.2.

En appliquant les sous clés de déchiffrement obtenus précédemment, et l'algorithme de décryptage tour par tour on obtient les résultats représentés dans le Tab.3.7.

Ciphertext Input Y

106b	dbfd	f323	0876
(Y_1)	(Y_2)	(Y_3)	(Y_4)

Round	Round output				
1	24e4	5069	fe98	dfd8	
2	ffb1	b4a0	75b2	0b77	
3	7420	e9e2	2749	00cc	
4	124c	4800	9d5d	9947	
5	9c42	efcb	28e8	70f9	
6	ed80	a415	78c9	bdca	
7	dca8	8bc1	f202	48a6	
8	3649	01cf	1775	1734	
9 (final transformation)	7fa9 (X_1)	1c37 (X_2)	ffb3 (X_3)	df05 (X_4)	→ Recovered Plaintext X

Tab.3.7 - Les différentes étapes du calcul le décryptage IDEA

Le message restitué est bien le texte clair $X = (X_1\ X_2\ X_3\ X_4) = (7fa9\ 1c37\ ffb3\ df05)$

2.2. L'algorithme RSA

Après avoir crypté le texte avec l'algorithme IDEA avec la clé de session, cette dernière doit être connu par le destinataire, pour cela nous devons la joindre dans le courrier, pas telle qu'elle est, mais cryptée par l'algorithme RSA. Ce système proposé par Rivest, Shamir, et Adleman, est aujourd'hui le plus connu et le plus utilisé par sa simplicité.

Voyant maintenant et détaillant cet algorithme par un schéma synoptique, et pour mieux comprendre nous prenons des exemples d'application.

2.2.1. Cryptage RSA

Dans le système RSA, un utilisateur crée son couple (clé publique, clé privée) en utilisant la procédure suivante :

1) Choisir au hasard deux grands nombres premiers p et q. Il faut que p et q contiennent au moins 100 chiffres décimaux chacun ;

2) Calculer $n = p.q$;

3) Choisir un petit entier e qui est premier avec $\varphi(n) = (p-1)(q-1)$;

4) Calculer d, l'inverse par la multiplication de e modulo $\varphi(n)$, $d = e^{-1}\left(mod\ \varphi(n)\right)$; or

$$e.d \equiv 1\left(mod\ \varphi(n)\right)$$

5) Publier la paire $K_e = (e,n)$ comme sa clé publique RSA ;

6) Garder secrète la paire $K_d = (d,n)$ qui est sa clé privée RSA ;

Toutes ces étapes sont schématisées par la Fig.3.5

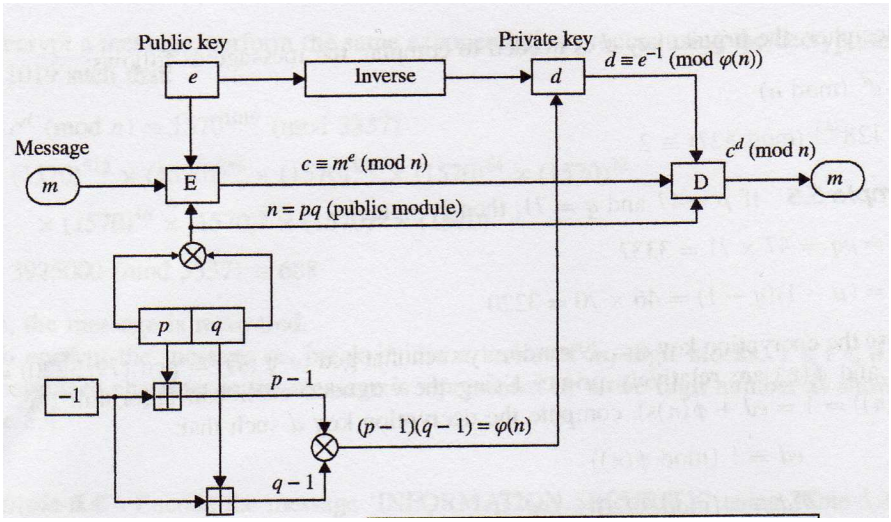

Fig.3.5 – L'algorithme de chiffrement / déchiffrement RSA

On à alors :

- chiffrement RSA $C = M^e\ (mod\ n)$
- déchiffrement RSA $M = C^d\ (mod\ n)$

Exemple.2.2.1 : Si $p = 17\ et\ q = 31$ sont choisis, alors $n = p.q\ = 17 \times 31 = 527$
$\varphi(n) = (p-1)(q-1) = 16 \times 30 = 480$. Si on choisi $e = 7$, alors on calcule :
$d \equiv e^{-1}\left(mod\ \varphi(n)\right)\ \equiv 7^{-1}\ (mod\ 480)\ \equiv 343$. Cette clé de décryptage est calculée en utilisant l'algorithme de la division euclidien.

$e.d \equiv 7 \times 343(mod\ 480)\ \equiv 2401(mod\ 480) \equiv 1$. Donc la clé (e,n) est ulisée pour crypter m.

Si $m = 2$, alors le message m est chiffré comme suit :

$C\ \equiv\ m^e\ (mod\ n) \equiv\ 2^7\ (mod\ 527) \equiv\ 128.$

Pour décrypter le message chiffré C, on utilise la clé privée d comme suit

$m\ \equiv\ C^{\,d}(mod\ n) \equiv\ 128^{343}\ (mod\ 527)\ \equiv\ 2.$

2.2.2. Exponentiation modulaire

Pour implémenter RSA, on a besoin d'un algorithme d'exponentiation modulaire pour calculer $a^b\ mod\ n$. On représente ci-dessous un algorithme qui travaille en temps proportionnel au logarithme de b : pour ce faire, on élève successivement au carré a, a^2, ...

Soit $\langle b_k,\ b_{k-1}, \dots, b_0\rangle$ la représentation binaire de b telle que $b = \sum_{i=0}^{k}\ b_i\ 2^i$. La procédure suivante calcule $a^b\ mod\ n$. [17]

Algorithme 3.1

Exponentiation modulaire (a, b, n)

$c \leftarrow 0\ ;\ d \leftarrow 1\ ;$

Soit $\langle b_k,\ b_{k-1}, \dots, b_0\rangle$ la représentation binaire de b

Pour $i \leftarrow k$ **jusqu'à** 0 **pas** -1 **faire**

$c \leftarrow 2.c\ ;\ d \leftarrow (d.d)\ mod\ n\ ;$

Si $b_i = 1$ **alors**

$c \leftarrow c + 1\ ;\ d \leftarrow (d.a)\ mod\ n\ ;$

fsi

fpour

renvoie d

Calculons par exemple les valeurs de $c\ et\ d$ à la fin de chaque itération pour $7^{560}\ mod\ 561$:

$b = \langle 1000110000\rangle$ et le résultat du calcul est la valeur finale de d qui est encadrée.

i	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
b_i	1	0	0	0	1	1	0	0	0	0
c	1	2	4	8	17	35	70	140	280	560
d	7	49	157	526	160	241	298	166	67	**1**

Tab.3.8 - La compilation de l'algorithme d'exponentiation modulaire

Chaque exposent c calculé est soit : le double de l'exposant précédent, soit le double de la valeur de l'exposant précédent plus 1. La représentation binaire de b est lue de droite vers l gauche afin de contrôler quelles opérations doit être effectuées. Chaque itération utilise une des deux identités suivantes :

$$a^{2c} \; mod \; n = (a^c)^2 \; mod \; n$$

$$a^{2c+1} \; mod \; n = \text{a}.\,(a^c)^2 \; mod \; n$$

Selon la valeur de b_i qui respectivement 0 ou 1. L'élévation au carré est la seule opération effective de l'itération. La variable c permet d'expliquer le fonctionnement de l'algorithme. L'invariant de l'algorithme est $d = a^c \; mod \; n$ puisque c ne fait que doubler ou doubler et croitre de un jusqu'à ce que $c = b$. On observe en outre que la valeur de c juste après le traitement du bit b_i est la même que le préfixe $\langle b_k, \; b_{k-1}, \ldots, b_0 \rangle$ de la représentation binaire de b.

Exemple.2.2.2 : illustrons par exemple le fonctionnement à la main de cet algorithme pour $17^{73} \; mod \; 133$; $b = 73 = \langle 1001001 \rangle$.

i	b_i	17^{2^i}	$17^{2^i} \; mod \; 133$	valeur
0	1	17	17 mod 133	17
1	0	17^2	289 mod 133	23
2	0	23^2	529 mod 133	130
3	1	130^2	16900 mod 133	9
4	0	9^2	81 mod 133	81
5	0	81^2	6561 mod 133	44
6	1	44^2	1936 mod 133	74

Et $17^{73} \; mod \; 133 = 17.17^{2^3}.17^{2^6} = 17.9.74 \; mod \; 133 = 17$

Exemple.2.2.3 : Si $p = 47 \; et \; q = 71$ sont choisis, alors $n = p.q \; = 47 \times 71 = 3337$

$\varphi(n) = (p-1)(q-1) = 46 \times 70 = 3220$. Si on choisi $e = 77$ tel que pgcd $(e, \varphi(n)) \; =$ pgcd $(79, 3220) \; = \; 1$, alors en utilisant l'algorithme de la division euclidien.

On calcule :

$d \equiv e^{-1} \left(mod \; \varphi(n)\right) \equiv 77^{-1} \; (mod \; 3220) \equiv 343$. Cette clé de décryptage est calculée en utilisant l'algorithme de la division euclidien.

$ed \; \equiv \; 1 \; (mod \; \varphi(n))$

$79d \; \equiv \; 1 \; (mod \; 3220)$

$3220 = 79 \times 40 + 60$

$79 = 60 + 19$

$60 = 19 \times 3 + 3$

$19 = 3 \times 6 + 1 \rightarrow gcd(79, 3220) = 1$

$1 = 19 - 3 \times 6 = 19 - (60 - 19 \times 3) \times 6$

$= 19 \times 19 - 60 \times 6$

$1 = (79 - 60) \times 19 - 60 \times 6$

$= 79 \times 19 - 60 \times 25$

$1 = 79 \times 19 - (3220 - 79 \times 40) \times 25$

$= 79 \times 1019 - 3220 \times 25$

$(79)(1019) \equiv 1 \ (mod\ 3220)$

$d = 1019$ (c'est la clé privée)

Maintenant en cryptant le texte 'INFORMATION SECURITY' avec la clé publique$(e, n) = (79, 3337)$. Pour transformer cette chaine de caractères en utilise la table de code ACII représenté dans le Tab.3. 9. Chaque lettre est représentée sur deux chiffres.

Blank	00	E	05	J	10	O	15	T	20	Y	25
A	01	F	06	K	11	P	16	U	21	Z	26
B	02	G	07	L	12	Q	17	V	22		
C	03	H	08	M	13	R	18	W	23		
D	04	I	09	N	14	S	19	X	24		

Tab.3.9 – La représentation des lettres en code Ascii

Donc notre message $m =$ (09140615181301200915140019050321180920 25)

On divise le message m en blocs de quatre chiffres chacun

0914 0615 1813 0120 0915

1400 1905 0321 1809 2025

On utilisant l'exponentiation modulaire, le premier bloc $m_1 = 914$ est crypté avec la clé publique $(79, 3337)$ et le résultat donne $c_1 = 3223$ comme le premier bloc de texte chiffré.

$c_1 \equiv {m_1}^e \ (mod\ n) \equiv 91479 \ (mod\ 3337) \equiv 3223$

On applique la même chose pour les autres blocs, on obtient ainsi un ensemble de blocs de textes chiffrés $c_i, 1 \leq i \leq 10$ donnés comme suit :

3223 3155 1012 1712 1595

2653 0802 2360 0832 1369

Pour décrypter le premier bloc de texte crypté on utilise la paire de la clé $(d = 1019, n = 3337)$

$m_1 \equiv {m_1}^d \ (mod\ n) \equiv 3223^{1019} \ (mod\ 3337) \equiv 914$

$m_2 \equiv {m_2}^d \ (mod\ n) \equiv 3155^{1019} \ (mod\ 3337) \equiv 615$

.
.
.

Ainsi le message recrée dans cette exemple est donné comme suit :

0914 0615 1813 0120 0915

1400 1905 0321 1809 2025

On refait le protocole inverse pour reconstituer les lettres du message et ainsi on obtient le message original.

3. La signature des données par PGP

Pour garantir le contenu de courrier et l'identité de l'expéditeur, le destinataire peut s'apercevoir que le courrier est arrivé de la part de la personne qu'il à bien pensé. Alors le PGP utilise un mécanisme de signature des messages. Il utilise la clé privé de l'expéditeur pour chiffrer une empreinte de message et le signe de cette manière. L'empreinte de message est un mot de 128 bits obtenu au moyen d'une fonction de hachage cryptographique à sens unique. Le mécanisme de signature des messages est décrit dans la Fig.3.6

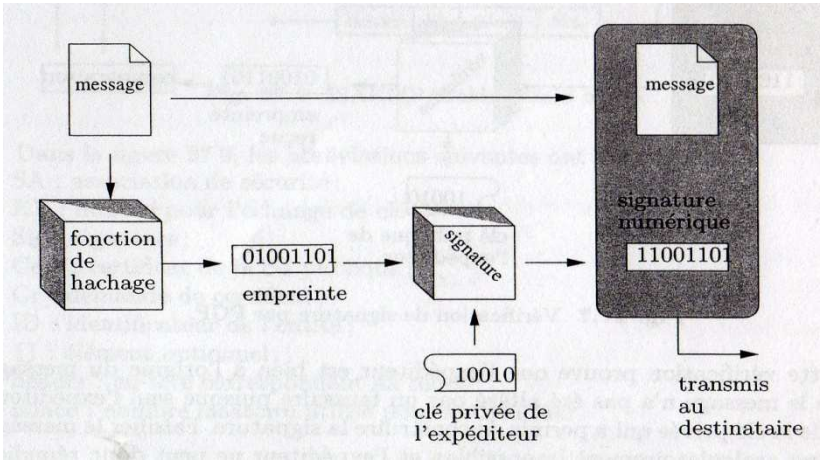

Fig.3.6 – Génération de signature par PGP

L'empreinte de message est un « condensé » de 128 bits de message. Il est caractéristique de message au sens unique, si le message a subi une quelque altération, l'empreinte sera complètement différente. De cette manière, il est possible de détecter toute altération de message par un faussaire. L'empreinte est calculée au moyen d'une fonction de hachage cryptographique à sens unique MD5, acronyme de "Message Digest" version 5.

A la réception de message, le destinataire peut vérifier la signature en utilisant la clef publique de l'expéditeur pour retrouver l'empreinte, puis il applique la fonction de hachage au texte en clair, il compare les deux empreintes comme décrit dans la Fig.3.7

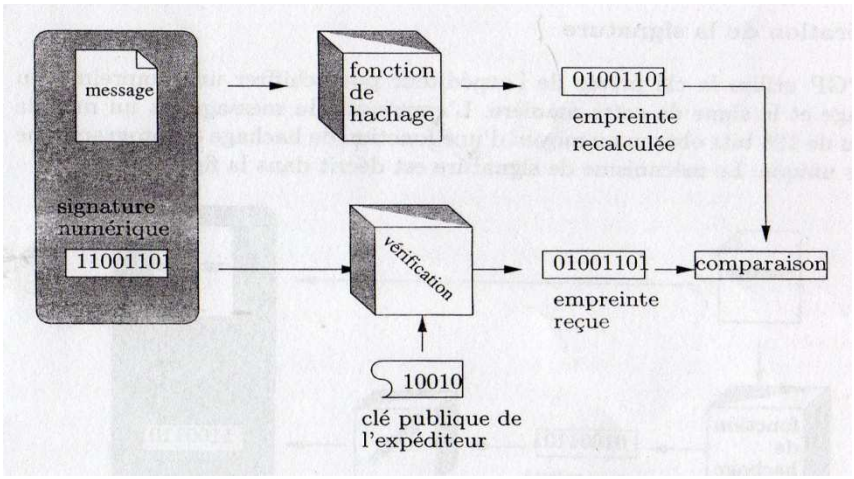

Fig.3.7 – Vérification de signature par PGP

Cette vérification prouve que l'expéditeur est bien à l'origine du message et que le message n'a pas été alterné par un faussaire puisque seul l'expéditeur possède la clé privée qui a permet de construire la signature. Falsifier le message est donc « calculatoire-ment impossible » et l'expéditeur ne peut donc répudier la signature de ce message.

3.1 La fonction de hachage MD5

MD5 est fonction de hachage cryptographique qui prend en entrée un message de longueur arbitraire pour produire des empreintes de 128 bits. Il s'agit de la fonction de hachage utilisée pour constituer l'empreinte de la signature de PGP.

MD5 fonctionne itérativement sur des mots de 32 bits. La fonction prend en entrée une variable de chaînage de quatre mots (la valeur initiale) ainsi que le message composé d'un bloque de seize mots et engendre une sortie de quatre mots (i.e. 128 bits) définissant ainsi la nouvelle variable de chaînage. On chaîne ensuite le bloc de message suivant avec le résultat du hachage du premier bloc comme décrit dans la Fig.3.8

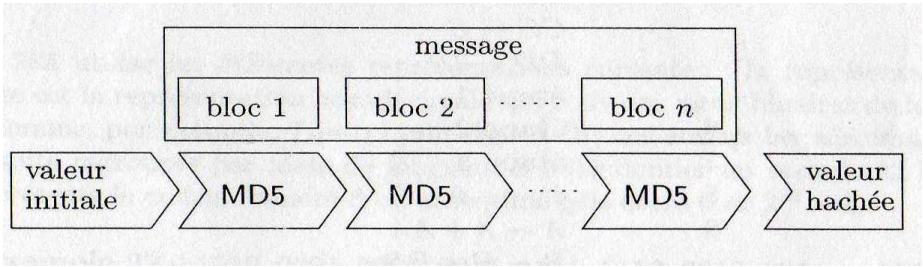

Fig.3.8 – Chaînage des blocs par MD5

Toutes les opérations utilisées par MD5 sont définies sur des mots de 32 bits. La transformation consiste en quatre étapes successives et chaque étape est constituée de seize sous-étapes. Dans

chacune des sous-étapes un mot des variables de chaînage est modifié comme suit : on ajoute un mot de message et le résultat de calcul d'une fonction non linéaire dépendant des trois autres mots de la variable de chaînage ; Ce résultat subit ensuite une permutation circulaire d'un certain nombre de bits.

3.1.1 Complémentation du message

MD5 commence par effectuer un complément dont le rôle est de transformer le message à hacher x pour que celui-ci soit multiple de 512 bits. L'entrée binaire est complétée par un seul 1 rajouté à la fin de message suivi d'autant de 0 que nécessaire, puis découpée en mots de 32bits $M[0]..M[n-1]$
En utilisant l'algorithme suivant :

Algorithme 3.2

1. $d \leftarrow 447 - |x| \ mod\ 512$
2. $l \leftarrow$ représentation binaire de $x\ mod\ 2^{64}\ tq\ |l| = 64$
3. $M \leftarrow x.1.0^{d}.l$

De cette manière on assure que *n* est un multiple de 16.

3.1.2 Description de l'algorithme

Quatre variables de 32 bits sont initialisées A, B, C, et D, ces registres sont initialisées sur les valeurs suivantes en hexadécimal.

A = 01 23 45 67

B = 89 ab cd ef

C = fe dc ba 98

D = 76 54 32 10

Ces quatre variables sont alors copiés dans d'autres variables : A dans AA, B dans BB, C dans CC, D dans DD.

L'algorithme ci-dessous décrit la procédure globale de calcule l'empreinte de 128 bits

Algorithme 3.3

1. $A \leftarrow$ 01 23 45 67
   $B \leftarrow$ 89 ab cd ef
   $C \leftarrow$ fe dc ba 98
   $D \leftarrow$76 54 32 10
2. **pour** $i \leftarrow 0$ **à** $\frac{n}{16} - 1$ **faire**

3. **pour** $j \leftarrow 0$ **à** 15 **faire**
   $X[j] \leftarrow M[16i + j]$
   fpour
4. $AA \leftarrow A$
   $BB \leftarrow B$
   $CC \leftarrow C$
   $DD \leftarrow D$
5. **étape 1**
6. **étape 2**
7. **étape 3**
8. **étape 4**
9. $AA \leftarrow A + AA$
   $BB \leftarrow B + BB$
   $CC \leftarrow C + CC$
   $DD \leftarrow D + DD$

fpour

L'empreinte est le résultat de la concaténation des mots ABCD rangés dans des registres mémoire. Ceux-ci sont illustrés à l'étape 1 de l'algorithme. On traite ensuite itérativement le tableau M par blocs de 16 mots. A chaque itération de boucle 2, on utilise un nouveau bloc de 16 mots de M qu'on range dans le tableau X à la ligne 3. Le contenu des registres est sauvegardé à la ligne 4. On effectue ensuite quatre étapes de hachage décrites ci-dessous qui traitent chacune 16 mots de X. Chaque étape de hachage produit de nouvelles valeurs pour les registres A, B, C, et D. pour terminer, on ajoute aux nouvelles valeurs des registres les valeurs initiales qui ont été sauvegardées par des additions modulo 2^{32}.

Les quatre étapes de MD5 sont toutes différentes et utilisent des opérations logiques suivantes choisies pour rapidité des ordinateurs modernes.

- $X \bullet Y$ le « et » logique bit à bit de X et Y ;
- $X + Y$ le «ou » logique bit à bit de X et Y ;
- $\oplus\ Y$ le «ou exclusif » bit à bit de X et Y ;
- $\bar{X}$ la complémentation bit à bit de X
- $X \ll s$ permutation circulaire de s bits vers la gauche $(0 \leq s \leq 31)$

Les étapes MD5 utilisent les quatre fonctions non-linéaires suivantes :

- $F(X, Y, Z) = (X \bullet Y) + (\bar{X} \bullet Z)$

- $G(X,Y,Z) = (X \bullet Z) + (Y \bullet \bar{Z})$
- $H(X,Y,Z) = X \oplus Y \oplus Z$
- $I(X,Y,Z) = Y \oplus (X + \bar{Z})$

Le résultat d'exécution des quatre fonctions est donné par le Tab.3.10

XYZ	FGHI
000	0001
001	1010
010	0110
011	1001
100	0011
101	0101
110	1100
111	1110

Tab.3.10 – Les fonction non linéaires FGHI

Ainsi qu'une table T [1..64] de 64 éléments construite au moyen de fonction sinus. T[i] représente la partie entière de $4294967296.\, abs(sin(i))$, i en radians et abs la valeur absolue [9].

$T[i] =$ la partie entière de $[2^{32} * |sin(i)|]$.

Quand $0 \leq |sin(i)| \leq 1$ alors $0 \leq 2^{32} * |sin(i)| \leq 2^{32}$ le résultat d'exécution de cette équation est donné par le Tab3.11. Les quatre étapes sont :

- **étape 1**

On note $FF[a,b,c,d,X[k],s,i]$

$$a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s)$$

On effectue les 16 opérations suivantes :

$FF[a,b,c,d,X[0],7,1], FF[d,a,b,c,X[1],12,2], FF[c,d,a,b,X[2],17,3],$

$FF[b,c,d,a,X[3],22,4], FF[a,b,c,d,X[4],7,5], FF[d,a,b,c,X[5],12,6],$

$FF[c,d,a,b,X[6],17,7], FF[b,c,d,a,X[7],22,8], FF[a,b,c,d,X[8],7,9],$

$FF[d,a,b,c,X[9],12,10], FF[c,d,a,b,X[10],17,11], FF[b,c,d,a,X[11],22,12],$

T[1] = d76aa478	T[17] = f61e2562	T[33] = fffa3942	T[49] = f4292244
T[2] = e8c7b756	T[18] = c040b340	T[34] = 8771f681	T[50] = 432aff97
T[3] = 242070db	T[19] = 265e5a51	T[35] = 69d96122	T[51] = ab9423a7
T[4] = c1bdceee	T[20] = e9b6c7aa	T[36] = fde5380c	T[52] = fc93a039
T[5] = f57c0faf	T[21] = d62f105d	T[37] = a4beea44	T[53] = 655b59c3
T[6] = 4787c62a	T[22] = 02441453	T[38] = 4bdecfa9	T[54] = 8f0ccc92
T[7] = a8304613	T[23] = d8a1e681	T[39] = f6bb4b60	T[55] = ffeff47d
T[8] = fd469501	T[24] = e7d3fbc8	T[40] = bebfbc70	T[56] = 85845dd1
T[9] = 698098d8	T[25] = 21e1cde6	T[41] = 289b7ec6	T[57] = 6fa87e4f
T[10] = 8b44f7af	T[26] = c33707d6	T[42] = eaa127fa	T[58] = fe2ce6e0
T[11] = ffff5bb1	T[27] = f4d50d87	T[43] = d4ef3085	T[59] = a3014314
T[12] = 895cd7be	T[28] = 455a14ed	T[44] = 04881d05	T[60] = 4e0811a1
T[13] = 6b901122	T[29] = a9e3e905	T[45] = d9d4d039	T[61] = f7537e82
T[14] = fd987193	T[30] = fcefa3f8	T[46] = e6db99e5	T[62] = bd3af235
T[15] = a679438e	T[31] = 676f02d9	T[47] = 1fa27cf8	T[63] = 2ad7d2bb
T[16] = 49b40821	T[32] = 8d2a4c8a	T[48] = c4ac5665	T[64] = eb86d391

Tab.3.11 – Les résultats T[i]

$FF[a,b,c,d,M[12],7,13], FF[d,a,b,c,M[13],12,14], FF[c,d,a,b,M[14],17,15],$

$FF[b,c,d,a,M[15],22,16]$.La fonction FF est schématisé dans la Fig.3.9

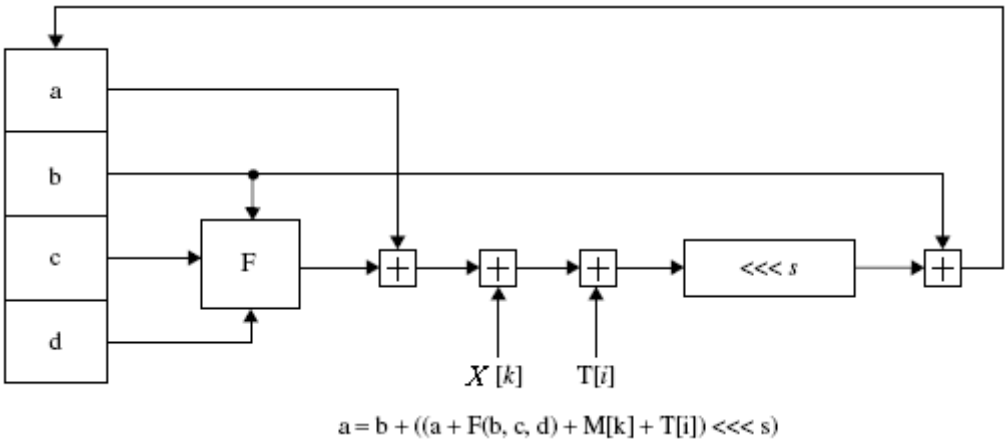

Fig.3.9 – Les opérations de base du MD5

- **étape 2**

On note $GG[a,b,c,d,X[k],s,i]$

$$a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s)$$

On effectue les 16 opérations suivantes :

$GG[a,b,c,d,X[1],5,17], GG[d,a,b,c,X[6],9,18], GG[c,d,a,b,X[11],14,19],$

$GG[b,c,d,a,X[0],20,20], GG[a,b,c,d,X[5],5,21], GG[d,a,b,c,X[10],9,22],$

$GG[c,d,a,b,X[15],14,23], GG[b,c,d,a,X[4],20,24], GG[a,b,c,d,X[9],5,25],$

$GG[d,a,b,c,X[14],9,26], GG[c,d,a,b,X[3],14,27], GG[b,c,d,a,X[8],20,28],$

$GG[a,b,c,d,X[13],5,29], GG[d,a,b,c,X[2],9,30], GG[c,d,a,b,X[7],14,31],$

$GG[b,c,d,a,X[12],20,32],$

- **étape 3**

On note $HH[a,b,c,d,X[k],s,i]$

$$a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s)$$

On effectue les 16 opérations suivantes :

$HH[a,b,c,d,X[5],4,33], HH[d,a,b,c,X[8],11,34], HH[c,d,a,b,X[11],16,35],$
$HH[b,c,d,a,X[14],23,36], HH[a,b,c,d,X[1],4,37], HH[d,a,b,c,X[4],11,38],$
$HH[c,d,a,b,X[7],16,39], HH[b,c,d,a,X[10],23,40], HH[a,b,c,d,X[13],4,41],$
$HH[d,a,b,c,X[0],11,42], HH[c,d,a,b,X[3],16,43], HH[b,c,d,a,X[6],23,44],$
$HH[a,b,c,d,X[9],4,45], HH[d,a,b,c,X[12],11,46], HH[c,d,a,b,X[15],16,47],$
$HH[b,c,d,a,X[2],23,48],$

- **étape 4**

On note $II[a,b,c,d,X[k],s,i]$

$$a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s)$$

On effectue les 16 opérations suivantes :

$II[a,b,c,d,X[0],6,49], II[d,a,b,c,X[7],10,50], II[c,d,a,b,X[14],15,51],$
$II[b,c,d,a,X[5],21,52], II[a,b,c,d,X[12],6,53], II[d,a,b,c,X[3],10,54],$
$II[c,d,a,b,X[10],15,55], II[b,c,d,a,X[1],21,56], II[a,b,c,d,X[8],6,57],$
$II[d,a,b,c,X[15],10,58], II[c,d,a,b,X[6],15,59], II[b,c,d,a,X[13],21,60],$
$II[a,b,c,d,X[4],6,61], II[d,a,b,c,X[11],10,62], II[c,d,a,b,X[2],15,63],$
$II[b,c,d,a,X[9],21,64].$

Un exemple d'application de toutes ces étapes est donné dans l'annexe A.

3.2 La signature RSA

L'empreinte produite par le MD5, sera signé par l'expéditeur avec l'algorithme de signature RSA, où le procédé de génération des paramètres est identique a celui utiliser pour l'algorithme de cryptage RSA. Chaque utilisateur a trois entiers $e, d, et\ n$ tel que $n = p.q$ avec $p\ et\ q$ deux grands nombre premiers. Pour la paire de la clé (e,d), $ed \equiv 1\ (mod\ \varphi(n))$ doit être satisfaite. Si un expéditeur A veut envoyer un message signé C correspondant à un message m au destinataire B, A signe m avec sa clé privée, effectue :$C = m^{d_A}\ (mod\ n_A)$. D'abord A effectue $\varphi(n_A) = lcm(p_A - 1, q_A - 1)$. Où lcm est le premier multiple commun. l'expéditeur A selecte la paire clé (e_A, d_A), tel que $e_A d_A \equiv 1\ (mod\ \varphi(n_A))$

La paire de clé (n_A, e_A) sera publié, la Fig3.10 illustre l'algorithme de signature RSA

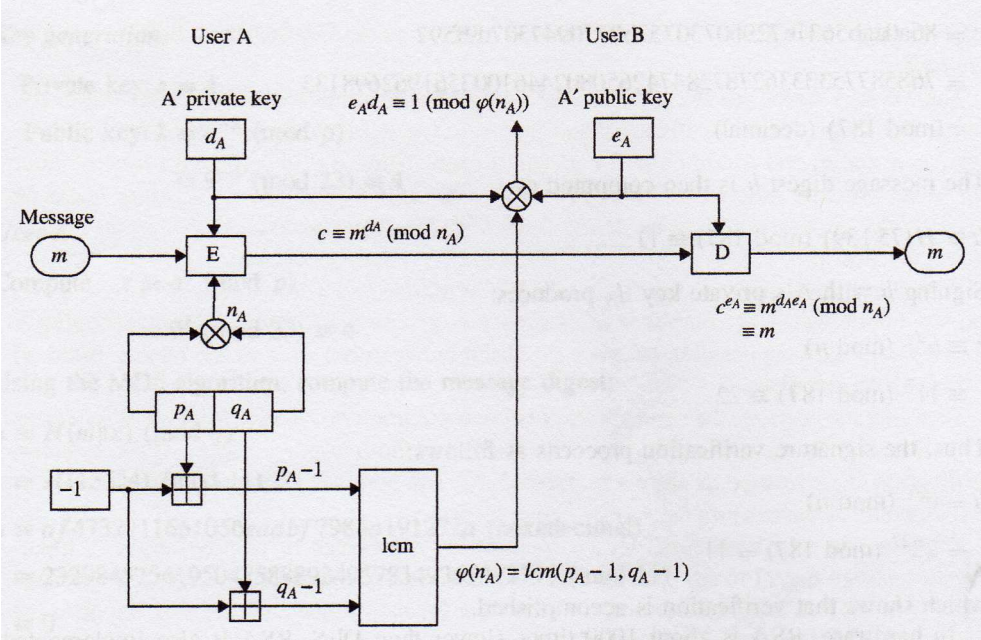

Fig.3.10 – L'algorithme de signature RSA

Exemple.3.2

Choisir $p = 11\ et\ q = 17$. Alors $n = pq = 187$

Effectue $\varphi(n) = lcm(p-1, q-1) = lcm(10,16) = 80$

Sélectionne $e_A = 27$. Alors $e_A d_A \equiv 1\ (mod\ \varphi(n_A)\ )$

$27d_A \equiv 1\ (mod\ 80\ )$

$d_A = 3$

Suppose que $m = 55$. Alors le message signé est:$C \equiv m^{d_A}\ (mod\ 187) \equiv 55^3\ (mod\ 187) \equiv 132$

Pour vérifier la signature $m = c^{e_A}\ (mod\ 187) \equiv 132^{27}\ (mod\ 187) \equiv 55$. Et ainsi la signature est bien vérifiée.

4. Les certificats

Rappelons que l'un des points clés des crypto-systèmes est la vigilance que l'utilisateur apporte à la vérification de l'appartenance de la clé au bon propriétaire.

Ces systèmes sont en effet sensibles aux attaques dites de 'l'homme au milieu' qui consiste pour l'attaquant à créer une clé qu'il fait passer pour la clé d'une autre personne, s'accordant ainsi la possibilité de lire tous les messages sensément destinés à la personne dont il a usurpée l'identité. Pour éviter cela, on crée des certificats numériques dont le but est d'apporter la preuve de la validité d'une clé et de son appartenance à un propriétaire donné.

Ces certificats peuvent être considérés comme les cartes d'identité des clés et se composent de trois parties :

- la clé publique pour laquelle le certificat est généré,
- des informations sur le détenteur de la clé (nom d'utilisateur, mails etc.) et sur le certificat lui-même (durée de validité …),
- d'une ou plusieurs signatures de personnes attestant de la validité de ces informations et de la clé.

En règle générale, une autorité appelée « autorité de certification » est seule habilitée à produire des certificats et les signer à l'aide de sa clé privée. Dans PGP, le système retenu pour les certifications numériques ne se base pas sur une autorité de certification centralisée mais sur un système de confiance.

Il est possible à chaque personne de signer de sa clé privée un certificat. Contrairement à un système centralisé, un certificat PGP pourra contenir une multitude de signatures numériques.

Le format d'un certificat PGP comprend entre autres les informations suivantes :

- Le numéro de la version de PGP utilisée pour créer la clé associée à ce certificat.
- La clé publique sur laquelle porte ce certificat ainsi que l'algorithme employé pour la générer.
- Des informations sur le propriétaire de cette clé (nom, mail, nom d'utilisateur etc.).
- La signature numérique du propriétaire effectuée à l'aide de la clé privée qui correspond à la clef publique du certificat.
- La période de validité du certificat.
- Les éventuelles signatures effectuées par d'autres utilisateurs.

PGP reconnaît également les certificats X. 509 [23]. Il est possible de produire ses propres certificats PGP. Pour les certificats X. 509, il faut effectuer la demande auprès d'une autorité de certification. La structure des certificats est normalisée par le standard X.509v3 de l' HYPERLINK "http://www.itu.int" UIT (Union Internationale des Télécommunications).

5. Les niveaux de confiance

Avec le système des certificats à signatures multiples se pose le problème de déterminer si les signatures apportées aux certificats sont dignes de confiance. Pour pallier cet inconvénient, un système basé sur les niveaux de confiance et de validité est mis en place.

De base, il existe dans PGP trois niveaux de validité :

- Valide.
- semi – valide.
- Invalide.

Ainsi que trois niveaux de confiance :

- confiance totale.
- confiance moyenne.
- aucune confiance.

Lorsque l'utilisateur génère sa paire de clé dans PGP, celle-ci se voit implicitement attribuer les niveaux maximums pour ces deux domaines.

Considérons maintenant que l'utilisateur importe la clé de X dans son système. Il valide la clé de X en signant son certificat et lui accorde une confiance maximum.

La clé de X à désormais valeur d'autorité de certification dans le système ainsi lorsqu'une clé signée par X est importée, elle sera considérée comme valide dans le système.

Si, désormais, une confiance moyenne est accordée à X et Y et que l'utilisateur importe une clé, deux cas de figure se présentent :

- si la clé est seulement signée par X ou Y, elle ne sera pas valide dans le système
- si la clé est signée par X et Y, elle sera valide.

En résumé, pour qu'une clé soit validée dans le système PGP de l'utilisateur, elle devra avoir reçu soit :

- la signature de ce dernier,
- la signature d'une clé à laquelle une confiance totale aura été accordée,
- la signature d'au moins deux clés auxquelles une confiance moyenne aura été apportée.

Les signatures effectuées par des clés qui ont le niveau de confiance 'Aucune Confiance' ne sont tout simplement pas prises en compte.

6. Implémentations pratiques

Dans le but d'achever l'évaluation de notre travail, nous allons présenter dans cette partie les différentes comparaisons de nos systèmes de chiffrement IDEA, RSA et PGP. Ces comparaisons se basent sur les temps de chiffrement et du déchiffrement. Pour cela, le langage de programmation utilisé est Matlab, sous l'environnement Windows. Nous avons développé une interface qui englobe les systèmes de chiffrement et du déchiffrement comme le montre la Fig.3.11

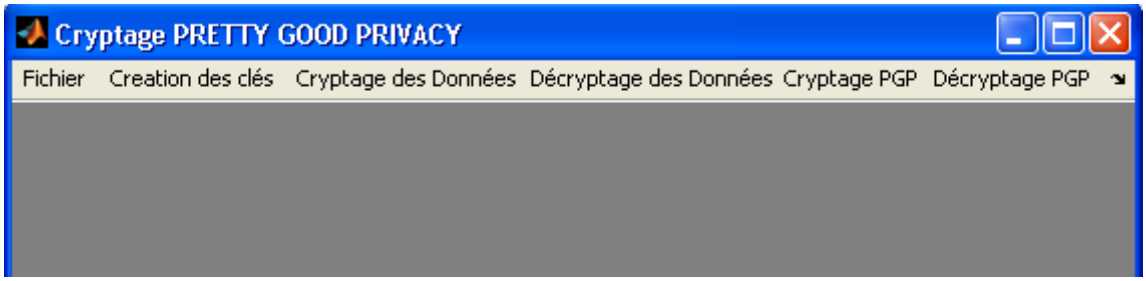

Fig.3.11 – Interface du programme réalisé

6.1 Génération des clés

Les deux systèmes de chiffrement symétrique, ou asymétrique se basent sur des clés qui permettent d'accomplir l'algorithme de cryptage selon leur type.

6.1.1 Génération des clés symétriques

Dans notre cas c'est la génération de la clé de session pour le cryptage IDEA utilisé par le PGP. La génération de la clé se fait aléatoirement pour créer une clé de 128 bits comme le montre la Fig.3.12

Fig.3.12 – Génération des clés symétrique

6.1.2 Génération des clés asymétriques

Le RSA utilise deux paires de clés, une paire de clés publiques et une privées. Où chaque utilisateur possède une clé publique, il la publie à ces correspondants. Chaque utilisateur garde sa clé privée secrètement. Pour un utilisateur qui ne possède pas ce genre de clé, le programme représenté dans la Fig.3.13 lui permet de l'obtenir.

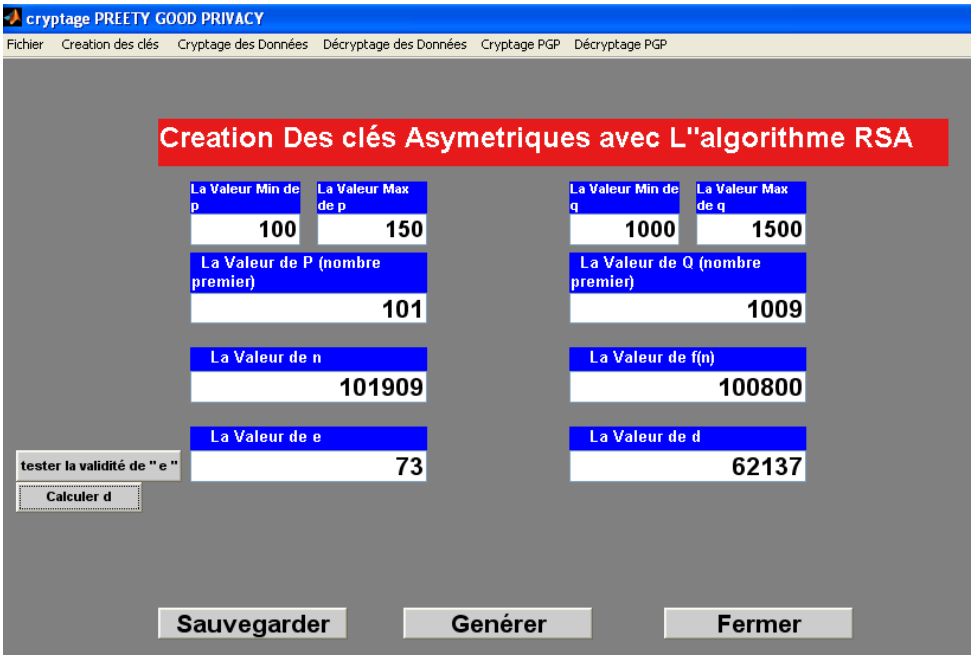

Fig.3.13 – Génération des clés asymétrique

Après l'exécution, le programme permet d'obtenir un fichier contenant toutes les informations sur la clé générée, ce fichier est représenté sur la Fig.3.14

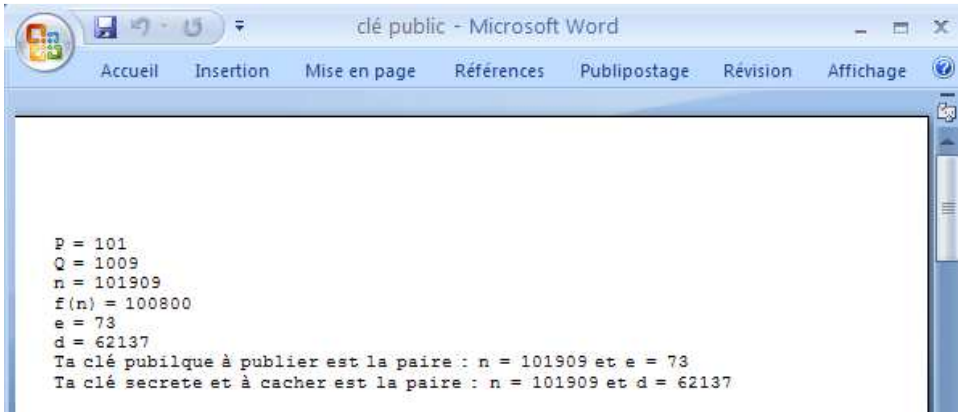

Fig.3.14 – Résultat de clés asymétrique générée

6.2 Implémentation RSA

On peut choisir l'importe quel fichier texte dans le poste de travail, on le charge et on lui applique l'algorithme RSA selon la clé de correspondant désiré. Comme le montre la Fig.3.15

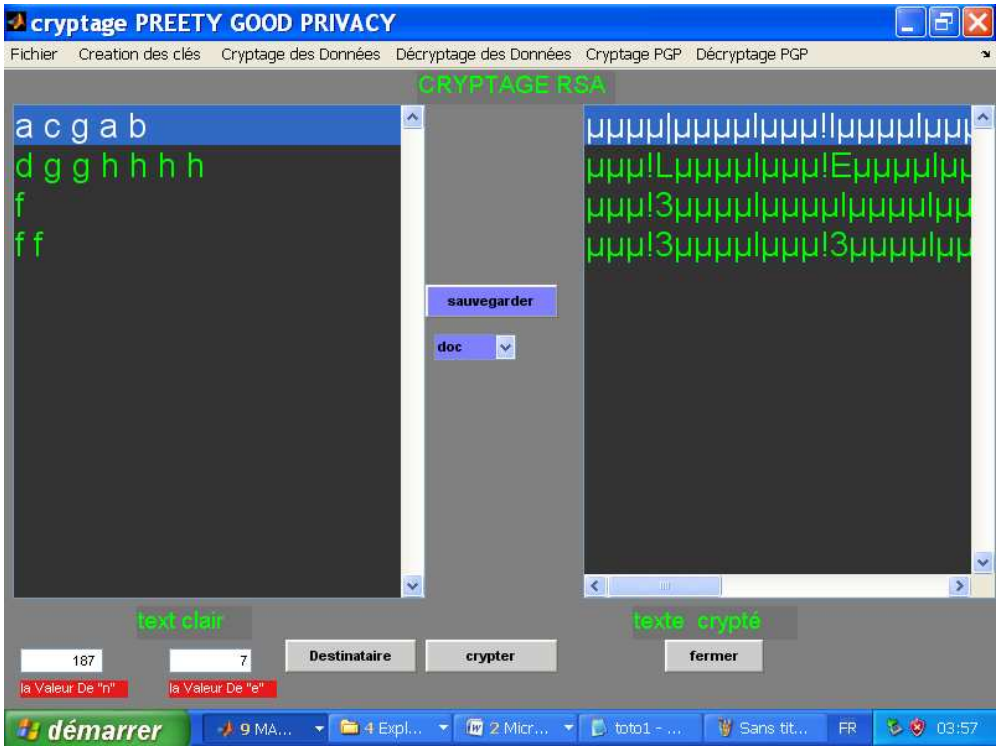

Fig.3.15 - Interface du RSA

Le résultat peut être sauvegardé puis l'envoyer par la suite, le même principe s'applique au décryptage RSA.

Nous constatons que l'algorithme RSA est lent proportionnellement au volume de texte à crypter.

6.3 Implémentation IDEA

L'algorithme IDEA peut être appliqué à l'importe quel fichier texte dans le poste de travaille, on introduisant une clé privée de 128 bits et on crypte le texte comme le montre la Fig.3.16

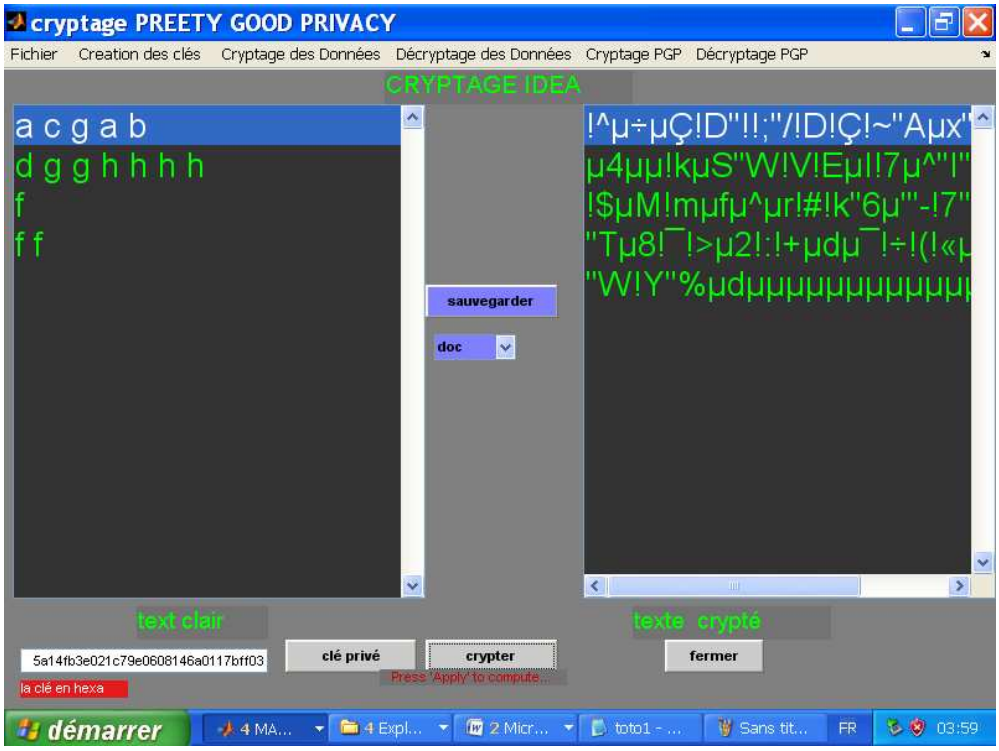

Fig.3.16 - Interface du IDEA

Le résultat peut être également sauvegardé, le même principe s'applique au décryptage IDEA.

Nous constatons que l'algorithme IDEA est plus rapide comparativement au RSA cependant son problème c'est l'échange de la clé secrète.

D'autre part pour des raisons de garantir le contenu et de confirmer que le message crypté, est bien provenu de la part de la personne que l'on pense, nous pouvons utiliser le RSA pour signer des empreintes par la clé privé de signataire, puis le destinataire peut vérifier cette signature par la clé publique de son expéditeur.

Les deux fonctions hachage et signature numérique sont implémentées dans notre interface, la fonction de hachage consiste à produire des empreintes de taille fixe de l'importe quel texte quel que soit sa taille, le cas traité dans notre travail c'est le MD5 qui prend en entrée un texte de taille arbitraire pour produire une empreinte de 128 bits en sortie comme le montre la Fig.3.17

Ensuite cette empreinte sera signée par l'expéditeur avec l'algorithme de la signature RSA comme le montre la Fig.3.18.

Fig.3.17 - Interface du MD5

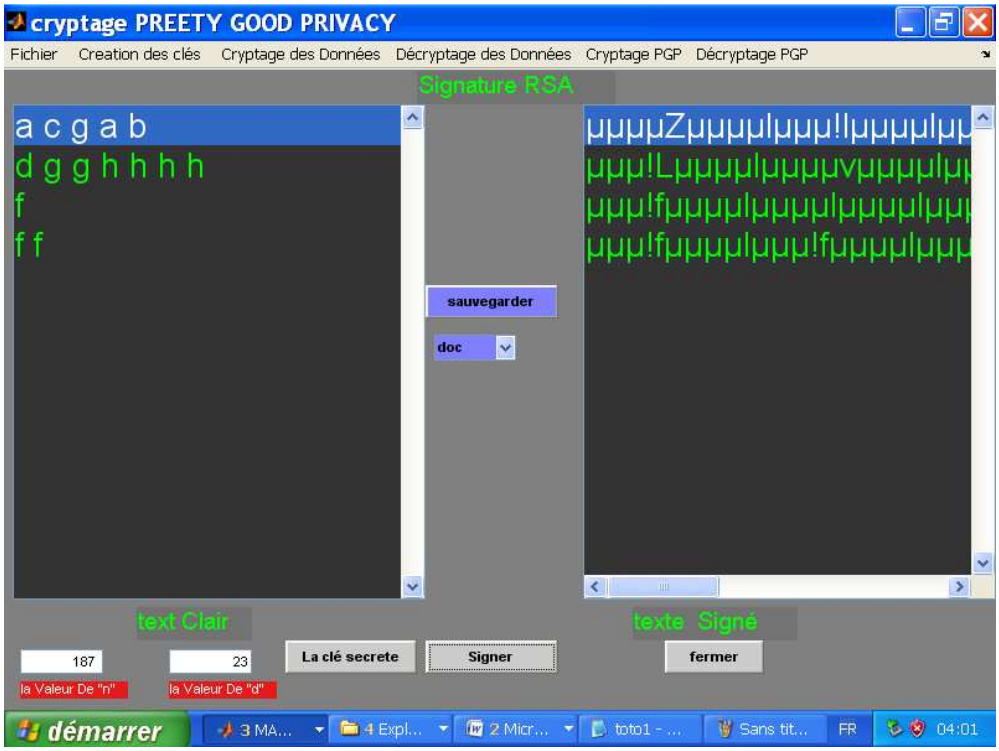

Fig.3.18 - Interface du Signature numérique

6.4 Implémentation PGP

6.4.1 Implémentation de chiffrement PGP

Tous ces algorithmes décrits auparavant sont combinés dans le système que nous avons développé dans ce travail, où le PGP est un système de chiffrement hybride, il combine les avantages de l'IDEA et le RSA. IDEA est beaucoup plus rapide que RSA alors qu'il est pratiquement impossible de percer la clé secrète de RSA. Toutes les procédures utilisées par l'IDEA et le RSA sont exploitées par le PGP.

Le principe de PGP est le suivant : un texte à crypter est choisi dans le poste de travail, ce dernier va être crypté par l'algorithme IDEA avec une clé de 128 bits générée d'une manière aléatoire comme le montre la Fig.3.19

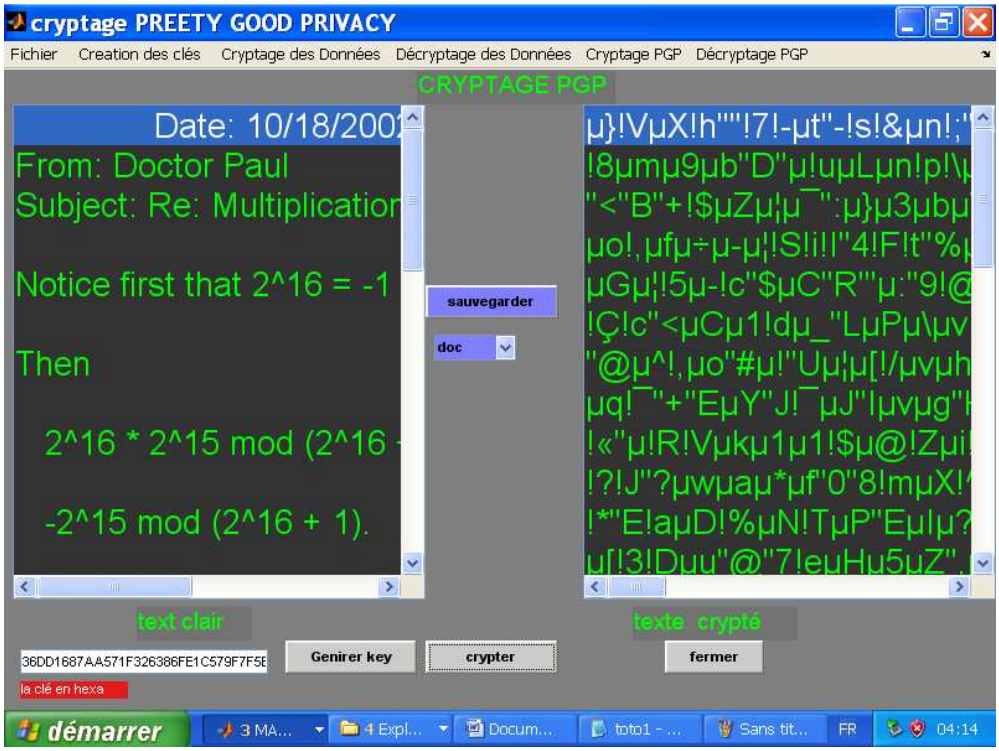

Fig.3.19 - Interface du PGP

Le texte crypté obtenu va être sauvegardé dans un répertoire choisi par l'utilisateur comme le montre la Fig.3.20

Fig.3.20 - Emplacement du texte crypté par PGP

Une fois le texte sauvegardé dans un dossier dans le répertoire choisi, on passe au cryptage de la clé de session avec la quelle le texte précédent a été crée, cette clé va être crypté par l'algorithme RSA avec la clé publique de notre destinataire comme le montre la Fig.3.21

Fig.3.21 - Cryptage de la clé de session

Une fois la clé est cryptée le résultat est enregistré dans le même dossier qui contient le texte crypté avec IDEA. Ainsi on obtient dans ce dossier le texte crypté avec IDEA et la clé de session cryptée. Le procèdes de la signature numérique est aussi utilisé dans notre travail, où l'utilisateur signe une empreinte MD5 de texte originale avec la signature RSA comme le montre les Fig.3.22 et Fig.3.23

Fig.3.22 - Interface du PGP-MD5

Fig.3.23 - Interface du PGP-signature

Le résultat est enregistré dans le même répertoire mais dans un autre dossier qu'on appel vérification de la signature.

Ainsi notre répertoire est prêt, il contient tous les fichiers nécessaires, on le compresse avec le Zip est on le envoie par Email à notre correspondant en toute sécurité.

6.4.2 Implémentation de déchiffrement PGP

A la réception le correspondant ouvre sa boite Email, il charge le fichier compressé, il le décompresse, il trouve ainsi un répertoire qui contient deux dossiers le premier contient le texte crypté avec IDEA, et la clé de session crypté par RSA, cependant le deuxième dossier contient la signature numérique.

Donc la première chose à faire c'est de décrypter la clé de session avec sa clé privée comme le montre la Fig3.24

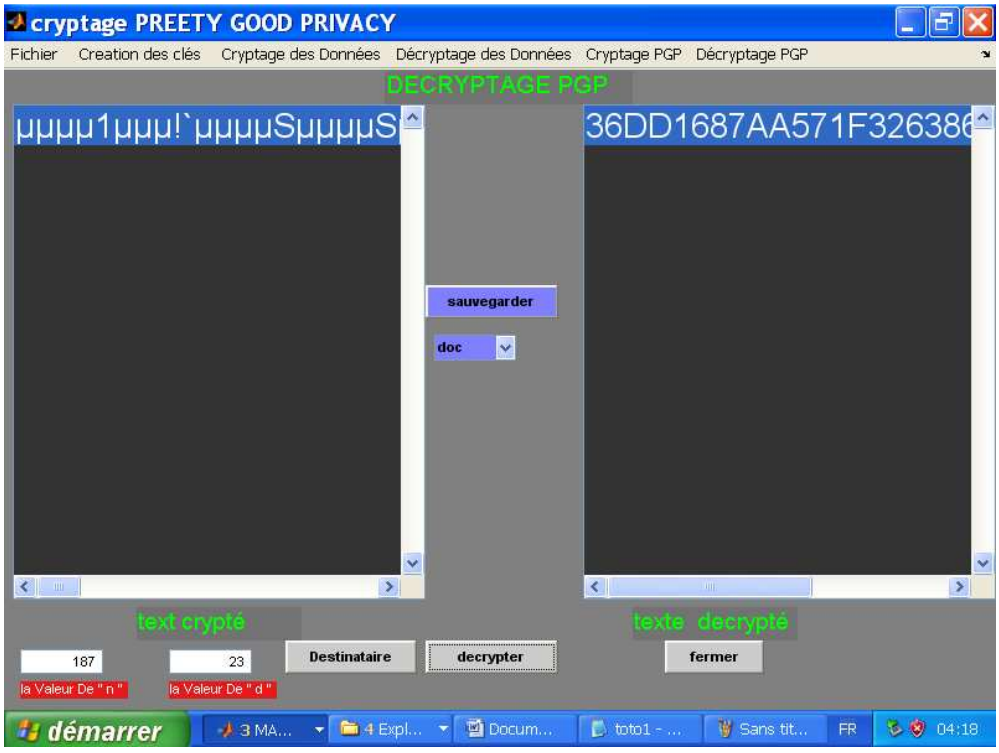

Fig.3.24 - Décryptage de la clé de session

Une fois la clé de session récupérée, il passe au décryptage de texte crypté avec cette clé comme le montre la Fig.3.25.

On constate bien que la clé récupérée et le texte déchiffré sont bien ceux qui ont été envoyés par l'expéditeur, mais le destinataire ne peut pas s'apercevoir que c'est les mêmes car il n'a pas vu le message original, c'est pour cela qu'il peut lui-même s'assurer de l'identité de l'expéditeur ainsi le contenu de message joindre en vérifiant la signature numérique comme le montre la Fig.3.26.

Donc le destinataire ouvre le fichier de la signature dans le deuxième dossier, il introduit la clé publique de l'expéditeur, il récupère ainsi l'empreinte du texte original. Pour confirmer que le contenu du texte décrypté est bien le même que le message original, le destinataire applique a son

tour une fonction de hachage sur le texte décrypté, il obtient ainsi une autre empreinte, il la compare à celle qui a été envoyée. Si c'est les mêmes comme c'est le cas ici, le destinataire est sûr de l'information transmise, sinon on peut dire que le message a perdu sa répudiation et sa confidentialité.

Fig.3.25 - Décryptage du texte crypté

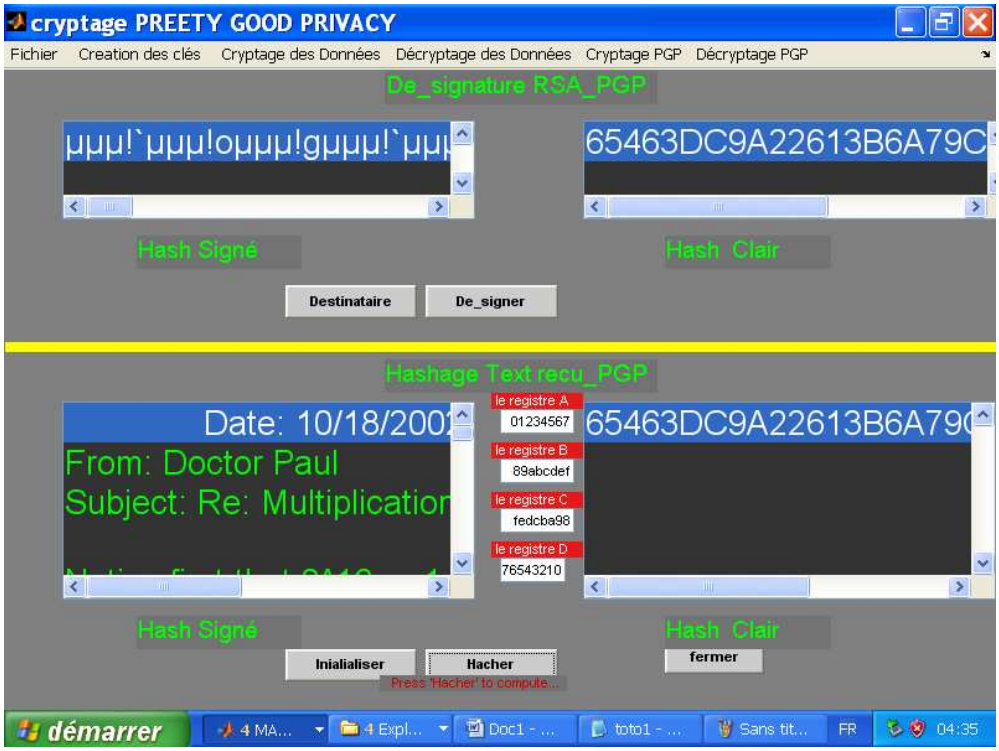

Fig.3.26 - Procèdes de la vérification

7. Discussion

Nous constatons que le système de chiffrement asymétrique représenté par l'algorithme RSA met un temps de calcul important et proportionnel au volume de texte à crypter, cependant il est très résistant aux attaques des crypto-analystes du fait que la clé privée reste secrète, et conservée par la personne concernée. D'autre part nous avons constaté que le système de chiffrement symétrique représenté par l'IDEA est l'algorithme le plus rapide du point de vue temps de calcul, par contre la contrainte de ce type de chiffrement réside dans l'échange de la clé, cela nécessite un canal sécurisé pour la transmettre, ce qui n'est pas rentable.

Le chiffrement Hybride PGP met un temps de calcul proche de celui d'un système symétrique, et en plus il évite l'échange de la clé secrète par un canal sécurisé, et cela grâce à l'algorithme RSA qui chiffre la clé secrète. Donc nous remarquons bien que le PGP possède plusieurs avantages tels que La rapidité en premier lieu, un niveau de sécurité très haut, et un moyen de vérification très important de l'identité de l'expéditeur, ainsi le contenu de l'information transmise. Tous ces facteurs mettent le PGP comme un moyen de sécurisation très fiable dans le domaine de la transmission sécurisée.

Conclusion Générale

Conclusion générale

Par le biais de cette thèse, nous avons pu réaliser un travail qui consiste à définir un système de chiffrement hybride nommé PGP utilisé dans le domaine de la transmission sécurisée. Ce système est basé sur la combinaison de deux types de chiffrement, le chiffrement symétrique, et le chiffrement asymétrique. Le premier type s'agit d'un chiffrement basé sur clé secrète qui ne doit pas être connue que par les deux entités communicantes, cette condition pose un problème de partage des clés. Par contre ce type de chiffrement reste fiable de point de vue temps de calcul. Tandis que le deuxième type s'agit d'un chiffrement basé sur une paire de clé, une clé publique distribuée sur tous les membres qui veulent envoyer des informations cryptées, mais la seule personne qui peut déchiffrer c'est celle qui possède la clé privée, le principe est celui de la boîte aux lettres : tout le monde peut déposer des lettres dans la boîte mais seule une personne, le propriétaire, peut les en retirer.

Le premier obstacle franchi pour la réalisation de ce travail, était la formalisation du problème de chiffrement de façon à éviter le partage des clés secrètes sur des canaux non sécurisés, tout en gardant le principe de chiffrement symétrique qui a un avantage relie à la vitesse de calcul des algorithmes, c'est ce qu' on a constaté avec l'algorithme IDEA utilisé. La clé utilisée par cet algorithme est crypté à son tour par un algorithme asymétrique représenté par le RSA.

Les résultats obtenu dans ce travail montre que le temps de calcul de l'algorithme IDEA est meilleur par rapport au celui de la RSA, et de PGP lui-même. Par contre sa robustesse réside dans la clé secrète. La diffusion de cette clé rend le chiffrement inutile. Pour cela elle ne doit être tombée que dans les bonnes mains. Le deuxième algorithme RSA mis un temps de calcul important. D'ailleurs il est le plus grand parmi les trois algorithmes, et dépend du volume de texte à crypter. Par contre l'avantage de l'algorithme réside du faite que la clé secrète est bien conservée par la personne concernée. Donc nous constatons que le PGP possède plusieurs avantages tels que la rapidité en premier lieu, où le message est chiffré par un cryptage symétrique. La clé IDEA est chiffrée de façon asymétrique. Toutefois, le volume de données que représente cette clef est négligeable par rapport au volume de données que représente le message. Par conséquent, le temps de chiffrage global est proche de celui d'un système symétrique. Deuxièmement le PGP offre une plus haute sécurité qu'un système à clef symétrique. En effet, si l'on peut considérer que le niveau de sécurité du système PGP est celui de son maillon le plus faible c'est-à-dire que le système à clé privée servant à coder le message. Il faut toutefois nuancer

cette conclusion. Car dans un système à clef privée standard, le canal d'échange de la clef est le point faible du système. Si l'on désire changer de clef afin de minimiser les risques, cela nécessite de définir un moyen d'échanger cette nouvelle clef, opération difficile à mettre en pratique. Dans PGP en revanche, la clef utilisée pour coder le message est nouvelle pour chaque message. Ce qui implique que pour effectuer une attaque il est nécessaire de casser au choix : autant de clefs privées que de messages, le système de clefs RSA, ce qui rend finalement PGP plus résistant qu'un système à clef privée classique.

L'étude expérimentale faite dans ce travail est limitée uniquement sur les textes, ou les chaines de caractères. Cependant la recherche dans ce domaine reste toujours en cours pour trouver des systèmes de sécurisation beaucoup plus robustes et meilleurs parmi ceux qui existent déjà. Certes, concevoir et réaliser un nouveau système cryptographique plus sûr et résistant ne serait que bénéfique pour cette science.

Dans le but de délimiter les champs de nos travaux, nous envisageons de nouvelles perspectives, ce type de cryptage peut être utilisé dans d'autres applications telles que les images, ou encore la vidéo où l'intérêt de la confidentialité est plus important. Ce sont nos futurs objectifs dans d'autres projets.

Bibliographie

BIBLIOGRAPHIE

[1] Mark Stamp, "*Information Security Principles and Practice*", San Jose State University. John Wiley 2006.

[2] H. Ollivier Robert, *" Pretty Good Privacy : un système de chiffrement sûr enfin à la portée de tous "*, *Internet Sécurité*, Tribunix N=°57, Septembre 1994, pp 84-86.

[3] Marie Virat, *" Courbes elliptiques sur un anneau et applications cryptographiques "*, Thèse de Doctorat, l'Université de Nice-Sophia Antipolis, avril 2009.

[4] Joshua Mason, Kathryn Watkins, "*A Natural Language Approach to Automated Cryptanalysis of Two-time Pads*", *Data Encryption Security, October 1996, pp 235-236.*

[5] Whitfield Diffie and Martine Hellman, "*New Directions in Cryptography*", *Transactions on Information Theory*, Vol. 22, N= 6, NOVEMBER 1976 pp 644-646.

[6] Cohen, A.E.; Parhi, K.K.; *" Architecture Optimizations for the RSA Public Key Cryptosystem: A Tutorial*", Circuits and Systems Magazine, Vol. 11, Septembre2011, pp 24 – 34.

[7] Verma, O.P.; Agarwal, R.; Dafouti, D.; Tyagi, S. "*Peformance analysis of data encryption algorithms"*, Third International Conference on Electronics Computer Technology (ICECT), 8-10 April 2011,Kanyakumari ,Inde, pp 399 - 400.

[8] Kumar, D.; Kashyap, D.; Mishra, K.K.; Misra, A.K. "*Security Vs cost: An issue of multi-objective optimization for choosing PGP algorithms*", International Conference on Computer and Communication Technology (ICCCT), 17-19 Sept. 2010 , Allahabad, Uttar Pradesh, Inde pp 532 - 534 .

[9] Katagishi, K.; Asami, T.; Ebihara, Y.; Sugiyama, T.; Toraichi, K. "*A public key cryptography-based security enhanced mail gateway with the mailing list function*", Conference on Communications, Computers and Signal Processing, 22-24 août 1999,Victoria, BC, Canada, pp:262-264.

[10] Kurniawan, Y.; Albone, A.; Rahyuwibowo, H. "*The design of mini PGP security"*, International Conferenceon Electrical Engineering and Informatics (ICEEI), 17-19 July 2011, Bandung, Indonesia, pp1-6.

[11] Jang, J.; Nepal, S.; Zic, J. "*Trusted Email protocol: Dealing with privacy concerns from malicious email intermediaries"*, 8th IEEE International Conference on Computer and Information Technology, 8-11 July 2008 ,Sydney, NSW, pp 402-406.

[12] McLaughlin, L. "*Philip Zimmermann on What's Next after PGP"*, Security & Privacy, IEEE ,Feb. 2006. pp 10-13.

[13] Marwaha, P.; Marwaha, P. "*Visual cryptographic steganography in images"*, International Conference on Computing Communication and Networking Technologies (ICCCNT), 29-31 July 2010, Karur, Inde, pp 1-6

[14] Alexandre Boyer, " *canaux de transmissions bruités* ", Institut National des Sciences Appliquées de Toulouse, Octobre 2010.

[15] Barbara Dalibard, Livre blanc, " *Machine To Machine* enjeux et perspectives ", Orange Business Services, Syntec Informatique. Octobre 2007.

[16] Touradj Ebrahimi, Franck Leprévost, Bertrand warsusfel, " *Cryptographie et sécurité des systèmes et réseaux* ", 2ém Edition, Lavoisier, 2006.

[17] Benoît Chevallier-Mames, "*Cryptographie à clé publique : Constructions et preuves de sécurité*", Thèse de Doctorat, l'Université Paris VII, Novembre 2006.

[18] Céline Chevalier, " *Étude de protocoles cryptographiques : à base de mots de passe* ", Thèse de Doctorat l'Université Paris 7, décembre 2009.

[19] Omary Fouzia, " *Applications des algorithmes évolutionnistes à la cryptographie* ", Thèse de Doctorat d'état, université Mohamed V- Agdal, Faculté des sciences, Rabat 20 juillet 2006.

[20] Bruno Martin, " *Codage, cryptologie, et applications* ", presses polytechnique et universitaire romandes, premier édition Lausanne 2004.

[21] Man Young Rhee, "*Internet Security: Cryptographic Principles, Algorithms, and Protocols"*, John Wiley, 2003.

[22] Daniel Mesquita, Lionel Torres Gilles, SassatelliI Michel Robert, "*Adéquation Algorithme Architecture : Des Opérateurs Arithmétiques Pour La Cryptographie Dans Les Architectures Reconfigurables "*, Juin 2003

[23] Bourgeois Morgan, " Initiation à PGP : GnuPG ", http://mbourgeois.developpez.com, 2006.

Annexe

Annexe A Exemple d'application du MD5

Exemple : calcul de l'empreinte d'un message M de taille 152 bits suivant : « 258b137a c317af24 a139b417 a8c5512f 36cb51 » avec les registres initiales : A = 67452301 ; B = efcdab89 ; C = 98badcfe ; D = 10325476.

Le message complété de 512 bits est produit à partir du message M de 152 bits en le complétant avec l'algorithme décrit en chapitre 3 ce qui donne 360 bits supplémentaires.

Donc

Message complémenté (512bits) = message originale (152bits) + complément (360bits):

7a138b25 24af17c3 17b439a1 2f51c5a8
8051cb36 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000098 00000000

I. Tour N°=1 exécution du FF[a, b, c, d, M[*k*], s, *i*] a = b + ((a + F(b, c, d) +M[k] + T[i]) <<< s) = b + U <<< s, $0 \leq k \leq 15$, $1 \leq i \leq 16$ où U <<< s noté les valeurs des 32bits obtenus par un décalage circulaire vers la gauche U de s positions.

(1) le premier mot bloc est: (M[0], T[1], s = 7). F(b, c, d) est donnée comme suit :

b: 1110 1111 1100 1101 1010 1011 1000 1001
c: 1001 1000 1011 1010 1101 1100 1111 1110
d: 0001 0000 0011 0010 0101 0100 0111 0110

F(b, c, d) : 1001 1000 1011 1010 1101 1100 1111 1110
9 8 b a d c f e

Calcule U = (a + F(b, c, d) +M[0] + T[1]) <<< s, s = 7

a: 67452301
F(b, c, d): 98badcfe
M[0]: 7a138b25
T[1]: d76aa478

U: 517e2f9c

U' = 517e2f 9c <<< 7 = (0101 0001 0111 1110 00410 1111 1001 1100) <<< 7

Où U <<< 7 noté le décalage circulaire de U vers de 7 bits, la valeur U après le décalage est :

U' = 1011 1111 0001 0111 1100 1110 0010 1000

b f 1 7 c e 2 8

À partir de a = b + U' nous avons :

b: efcdab89

U': bf17ce28

a: aee579b1

Donc, FF[a, b, c, d, M(0), 7, 1] donne les résultats de a, b, c, et d respectivement sont : aee579b1, efcdab89, 98badcfe, 10325476.

(2) le 2^{ém} mot bloc est (M[1], T[2], s = 12)

Utilisant la sortie de l'opération (1), le 2^{ém} mot bloc est calculé comme suit:

d: 10325476

F[a, b, c]: bedfadcf

M[1]: 24af17c3

T[2]: e8c7b756

U: dc88d15e

U' = U <<< 12 : 8d15edc8

A partir de : d = a + U', nous avons

a: aee57961

U': 8d15edc8

d: 3bfb6779

Donc, le résultat de l'opération (2) pour le 2^{ém} mot bloc devient FF[d, a, b, c, M[1],12, 2] = (aee57961, efcdab89, 98badcfe, 3bfb6779).

Toutes les transformations de premier tour sont calculées similairement, et construisent les résultats suivant à partir des 16 opérations:

[1] aee57961 efcdab89 98badcfe 10325476

[2] aee57961 efcdab89 98badcfe 3bfb6779

[3] aee57961 efcdab89 1e52ee63 3bfb6779

[4] aee57961 2279e391 1e52ee63 3bfb6779

[5] 65976331 2279e391 1e52ee63 3bfb6779

[6] 65976331 2279e391 1e52ee63 b766cf0e

[7] 65976331 2279e391 e776a653 b766cf0e

[8] 65976331 d4a89062 e776a653 b766cf0e

[9] 140e3c3d d4a89062 e776a653 b766cf0e

[10] 140e3c3d d4a89062 e776a653 59a02fdf

[11] 140e3c3d d4a89062 d62326dc 59a02fdf

[12] 140e3c3d 9d8eb345 d62326dc 59a02fdf

[13] 7dccd1ee 9d8eb345 d62326dc 59a02fdf

[14] 7dccd1ee 9d8eb345 d62326dc 0359415c

[15] 7dccd1ee 9d8eb345 bff77632 0359415c

[16] 7dccd1ee 10821d51 bff77632 0359415c

II. Tour N°=2 exécution du GG

(1) le premier mot bloc est l'opération :

a = b + ((a + G(b, c, d) +M[1] + T[17]) <<< s)

Mettant V = a + G*(*b, c, d*)* +M[1] + T[17] où a = 7dccd1ee, b = 10821d51, c = bff77632, d = 0359415c, M[1] = 24af17c3, et T[17] = f61e2562. G(b, c, d) est donné comme suit:

b: 0001 0000 1000 0010 0001 1101 0101 0001

c: 1011 1111 1111 0111 0111 0110 0011 0010

d: 0000 0011 0101 1001 0100 0001 0101 1100

G(b, c, d): 1011 1100 1010 0110 0011 0111 0111 0010

B c a 6 3 7 7 2

Calculant V = a + G*(*b, c, d*)* +M[1] + T[17];

a: 7dccd1ee

G(b, c, d): bca63772

M[1]: 24af17c3

T[17]: f61e2562

V: 55404685

V: 0101 0101 0100 0000 0100 0110 1000 0101

Donc V' = V <<< 5, V' devient V'= 1010 1000 0000 1000 1101 0000 1010 1010

a 8 0 8 d 0 a a

De a = b + V', nous avons

b: 10821d51

V': a808d0aa

---

a: b88aedfb

Ainsi, GG[a, b, c, d, M[1], T[17], 5] de la première opération (1) est obtenu comme suit :

b88aedfb, 10821d51, bff77632, 0359415c

Alors les 16 opérations de la transformation GG du tour N°=2 donnent les résultats suivants :

[1] b88aedfb 10821d51 bff77632 0359415c

[2] b88aedfb 10821d51 bff77632 f14f0cf3

[3] b88aedfb 10821d51 20aeb48b f14f0cf3

[4] b88aedfb 6b6c164c 20aeb48b f14f0cf3

[5] 80426a6a 6b6c164c 20aeb48b f14f0cf3

[6] 80426a6a 6b6c164c 20aeb48b 2ac992e7

[7] 80426a6a 6b6c164c f0263bcd 2ac992e7

[8] 80426a6a 719e1da6 f0263bcd 2ac992e7

[9] cbec5d78 719e1da6 f0263bcd 2ac992e7

[10] cbec5d78 719e1da6 f0263bcd 455ddcd7

[11] cbec5d78 719e1da6 a05494c9 455ddcd7

[12] cbec5d78 167849a5 a05494c9 455ddcd7

[13] 5b8a2ae8 167849a5 a05494c9 455ddcd7

[14] 5b8a2ae8 167849a5 a05494c9 af92e3c8

[15] 5b8a2ae8 167849a5 2e6d799d af92e3c8

[16] 5b8a2ae8 29e29554 2e6d799d af92e3c8

III. Tour N°=3 exécution du HH

(1) le premier mot bloc est l'opération :

a = b + ((a + H(b, c, d) +M[5] + T[33]) <<< 4), où a = 5b8a2ae8, b = 29e29554, c=2e6d799d, d = af92e3c8, M[5] = 00000000, T[33] = fffa3942, et s = 4. H(b, c, d) est donné comme suit:

b: 0010 1001 1110 0010 1001 0101 0101 0100

c: 0010 1110 0110 1101 0111 1001 1001 1101

d: 1010 1111 1001 0010 1110 0011 1100 1000

H(b, c, d): 1010 1000 0001 1101 0000 1111 0000 0001

a 8 1 d 0 f 0 1

Posant:

W = a + H(b, c, d) +M[5] + T[33]

a: 5b8a2ae8

H(b, c, d): a81d0f01

M[5]: 00000000

T[33]: fffa3942

W: 03a1732b

W = 0000 0011 1010 0001 0111 0011 0010 1011

À partir de $W' = W <<< 4$, we have W' = 0011 1010 0001 0111 0011 0010 1011 0000

3 a 1 7 3 2 b 0

On à : $a = b + W'$, a est calculé comme suit:

b: 29e29554

W': 3a1732b0

a: 63f9c804

Ainsi, HH[a, b, c, d, M[5], T[33], 4] de la première opération (1) est obtenu comme suit : 63f9c804, 29e29554, 2e6d799d, af92e3c8.

Alors les 16 opérations de la transformation HH du tour N°=3 donnent les résultats suivants :

[1] 63f9c804 29e29554 2e6d799d af92e3c8

[2] 63f9c804 29e29554 2e6d799d 3bf27cdf

[3] 63f9c804 29e29554 38408ad2 3bf27cdf

[4] 63f9c804 39049458 38408ad2 3bf27cdf

[5] bae75a5e 39049458 38408ad2 3bf27cdf

[6] bae75a5e 39049458 38408ad2 edcbf07c

[7] bae75a5e 39049458 02788da0 edcbf07c

[8] bae75a5e 279f19dc 02788da0 edcbf07c

[9] e292ec26 279f19dc 02788da0 edcbf07c

[10] e292ec26 279f19dc 02788da0 937294f5

[11] e292ec26 279f19dc 784ef22d 937294f5

[12] e292ec26 67e9dd0d 784ef22d 937294f5

[13] fbc16051 67e9dd0d 784ef22d 937294f5

[14] fbc16051 67e9dd0d 784ef22d 9fb3bb46

[15] fbc16051 67e9dd0d 14f356d2 9fb3bb46

[16] fbc16051 814dbccf 14f356d2 9fb3bb46

IV. Tour N°=4 exécution du II

(1) le premier mot bloc est l'opération :

$a = b + ((a + I(b, c, d) + M[0] + T[49]) <<< 6)$. où a = fbc16051, b = 814dbccf, c = 14f356d2, d = 9fb3bb46, M[0] = 7a138b25, T[49] = f4292244, et s = 6. I (b, c, d) est donné comme suit:

b: 1000 0001 0100 1101 1011 1100 1100 1111

c: 0001 0100 1111 0011 0101 0110 1101 0010

d: 1001 1111 1011 0011 1011 1011 0100 0110

I(b, c, d): 1111 0101 1011 1110 1010 1010 0010 1101

F 5 b e a a 2 d

Posant Z = a + I(b, c, d) +M[0] + T[49]

a: fbc16051

I(b, c, d): f5beaa2d

M[0]: 7a138b25

T[49]: f4292244

Z : 5fbcb7e7

Z = 0101 1111 1011 1100 1011 0111 1110 0111

À partir de Z' = Z <<< 6, nous avons Z' = 1110 1111 0010 1101 1111 1001 1101 0111

e f 2 d f 9 d 7

On à: a = b + Z', a est calculé comme suit:

b: 814dbccf

Z': ef2df9d7

a: 707bb6a6

Ainsi, l'opération (1) de II[a, b, c, d, M[0], T[49], 6] est obtenue comme suit : 707bb6a6 814dbccf 14f356d2 9fb3bb46.

Alors les 16 opérations de la transformation II du tour N°=4 donnent les résultats suivants :

[1] 707bb6a6 814dbccf 14f356d2 9fb3bb46

[2] 707bb6a6 814dbccf 14f356d2 b374ac1a

[3] 707bb6a6 814dbccf 1dcb5424 b374ac1a

[4] 707bb6a6 ebc0a7cd 1dcb5424 b374ac1a

[5] e1adb47e ebc0a7cd 1dcb5424 b374ac1a

[6] e1adb47e ebc0a7cd 1dcb5424 2307ce67

[7] e1adb47e ebc0a7cd fc5d488d 2307ce67

[8] e1adb47e 65cbb221 fc5d488d 2307ce67

[9] 25173275 65cbb221 fc5d488d 2307ce67

[10] 25173275 65cbb221 fc5d488d e801a803

[11] 25173275 65cbb221 9da76743 e801a803

[12] 25173275 0f04df84 9da76743 e801a803

[13] d4921a8b 0f04df84 9da76743 e801a803

[14] d4921a8b 0f04df84 9da76743 400fe907

[15] d4921a8b 0f04df84 f3d96b57 400fe907

[16] d4921a8b 24903b0e f3d96b57 400fe907

Les quatre registres de 32-bit A, B, C et D sont utilisés pour former le message digest de 128 bits. Ces registres sont initialisé aux valeurs suivantes : aa = 67452301, bb = efcdab89

cc = 98badcfe, dd = 10325476

La dernière opération de la transformation est :

a = d4921a8b, b = 24903b0e

c = f3d96b57, d = 400fe907

Ensuite, les additions suivantes sont finalement utilisées pour produire le message digest

A = a + aa

B = b + bb

C = c + cc

D = d + dd

Le message digest produit comme étant la sortie A, B, C et D est la concaténation de A, B, C et D.

a: d4921a8b	b: 24903b0e
aa: 67452301	bb: efcdab89
A: 3bd73d8c	B: 145de697

c: f3d96b57	d: 400fe907
cc: 98badcfe	dd: 10325476
C: 8c944855	D: 50423d7d

La concaténation des quatre sorties de A, B, C et D est le message digeste de 128-bit

Ainsi : A|| B|| C|| D = 3bd73d8c 145de697 8c944855 50423d7d