REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté du génie électrique et d'informatique.
Département d'informatique.

Mémoire

En vue de l'obtention d'un diplôme de Master en Informatique
Option : « Réseaux, Mobilités et Systèmes Embarqués »

Thème

Méthode de génération de clé RSA selon une taille désirée

Proposé et dirigé par :

M^me^ R.HADAOUI.

Réalisé par :

M^r^ SEDDAR Brahim.

M^r^ TALATIZI Arezki.

Promotion : 2014-2015

Remerciement

Nous tenons à remercier vivement :

- ✓ Notre promotrice M^me^, Hadaoui pour tous ses conseils précieux, ses orientations et ses encouragements.
- ✓ Nous remercions vivement les membres du jury d'avoir aimablement accepté de juger notre travail.
- Tous ceux qui nous ont aidés de près ou de loin dans notre travail.

Brahim & Arezki

Dédicaces

Je dédie ce modeste travail à :

- A Mes très chers parents qui ma toujours encouragé dans mes études ;
- A Mes frères et Mes sœurs et ainsi la famille SEDDAR ;
- A tous mes amis(es) des prés et de loin.
- A Mon ami et mon binôme AREZKI avec qui j'ai réalisé ce travail ainsi qu'à toute sa famille.
- A Toute la promotion Master 2 RMSE

2014/2015.

Brahim

Dédicaces

Je dédie ce modeste travail à :

- A Mes très chers parents qui ma toujours encouragé dans mes études ;
- A Mes frères et Mes sœurs et ainsi la famille TALATIZI ;
- A tous mes amis(es) des prés et de loin.
- A Mon ami et mon binôme BRAHIM avec qui j'ai réalisé ce travail ainsi qu'à toute sa famille.
- A Toute la promotion Master 2 RMSE

2014/2015.

Arezki

Tables des matières :

Chapitre III : La cryptographie

Chapitre IV : L'algorithme de RSA

Chapitre V : Conception et réalisation

Liste des Figures et des Algorithmes

Liste des Figures :

Liste des algorithmes :

Introduction Générale :

La cryptographie, ou art de chiffrer, coder les messages, est devenue aujourd'hui une science à part entière. Au croisement des Mathématiques et de l'Informatique**,** elle permet ce dont les civilisations ont besoin depuis qu'elles existent : le maintien du secret. Pour éviter une guerre, protéger un peuple, il est parfois nécessaire de cacher des choses.

La cryptographie étant un sujet très vaste, ce document se focalisera essentiellement sur les méthodes de chiffrement dites modernes, c'est-à-dire celles étant apparues et utilisées après la Seconde Guerre mondiale. En passant par le fameux **RSA**, le protocole le plus utilisé de nos jours. Ayant longtemps été l'apanage des militaires et des sociétés possédant de gros moyens financiers, la cryptographie s'est au fil du temps ouverte au grand public, et est donc un sujet digne d'intérêt.

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 divulguées, et personne ne doit pouvoir les détruire. De même que dans les plus grandes entreprises mondiales, les informations classées sensibles doivent être sécurisées pour assurer la fiabilité de leur échange via un réseau, et même augmenter la sécurité du stockage de ces informations au sein de l'entreprise elle-même.

Notre travail consiste à concevoir et réaliser une application qui permettre à crypter des données en utilisant l'algorithme de cryptage RSA pour la sécurité, et notre mémoire sera organisé de la manière suivante :

- **Chapitre I** : Réseaux Informatique
  Présentation de quelque généralité sur les réseaux, l'architecture client/serveur.
- **Chapitre II** : Sécurité Informatique
  Présentation de quelque généralité sur la sécurité informatique.
- **Chapitre III** : La cryptographie
  On présentera les bases de la cryptographie.
- **Chapitre IV** : L'algorithme de RSA.
  On va étudier l'algorithme RSA.
- **Chapitre V** : Conception et réalisation
  La réalisation de notre application en présentons les différents outils nécessaire pour la mise en œuvre de notre projet ainsi on présentera le fonctionnement de celle-ci.

CHAPITRE I

RESEAUX INFORMATIQUE

Introduction :

Les réseaux informatiques sont nés du besoin de relier des terminaux distants à un site central puis des ordinateurs entre eux et enfin des machines terminales, telles que stations de travail ou serveurs. Dans un premier temps, ces communications étaient destinées au transport des données informatiques.

Dans ce chapitre nous présenterons les différentes classifications de réseau informatique, ainsi leurs modèles d'architectures.

I. Définition :

Un réseau (network) informatique est un ensemble d'ordinateurs et de terminaux interconnectés pour échanger des informations numériques. Il permet de faire circuler des données et ainsi d'échanger du texte, des images, de la vidéo ou du son entre chaque équipement selon des règles et des protocoles bien définis.

I.1 Les objectifs d'un réseau : [1]

Les réseaux permettent :

• Le partage des fichiers

• Le partage d'application : compilation, SGBD

• Partage de ressources matérielles : l'imprimante, disque…

• Télécharger des applications et des fichiers

• L'interaction avec les utilisateurs connectés : messagerie électronique, conférences électroniques, ….

• Le transfert de données en général: réseaux informatiques

• Le transfert de la parole : réseaux téléphoniques

• Le transfert de la parole, de la vidéo et des données : réseaux numérique à intégration de services IP.

I.2 Classification des réseaux : [2]

I.2.1 Classification des réseaux selon leur étendu :

- **LAN : *Local Area Network* = réseau local d'entreprise (RLE en français)**

Un réseau local est un réseau d'ordinateurs situés sur un même site. Les communications sur ce type de réseau y sont généralement rapides (100 Mbits/s ou

1Gbits/s) et gratuites puisqu'elles ne passent pas par les services d'un opérateur de télécommunication. Le fait que le réseau soit sur un site bien délimité n'implique pas nécessairement qu'il soit de taille très réduite. Il est souhaitable de le segmenter en sous-réseaux quand le nombre de nœuds y devient important. L'ensemble reste un réseau local tant qu'il est indépendant des services d'un opérateur extérieur.

- **MAN : *Metropolitan Area Network* = Réseau métropolitain**

Lorsqu'un réseau privé; s'étend sur plusieurs kilomètre, dans une ville par exemple les réseaux locaux sont interconnectés via des liaisons téléphoniques à haut débit ou à l'aide d'équipements spéciaux comme des transmissions hertziennes. Ce type de regroupement de réseaux locaux peut se faire au niveau d'une ville et l'infrastructure du réseau métropolitain peut être privée ou publique.

- **WAN : *Wide Area Network* = Réseau étendu**

Ces réseaux relient plusieurs réseaux locaux en les interconnectant via des lignes louées ou via Internet. Ex. les réseaux bancaires qui établissent des liaisons entre les agences et le siège central.

Dans le cas de l'utilisation d'Internet, on parle de VPN (Virtual Private Network) puisqu'on utilise alors un réseau public pour faire transiter des informations privées.

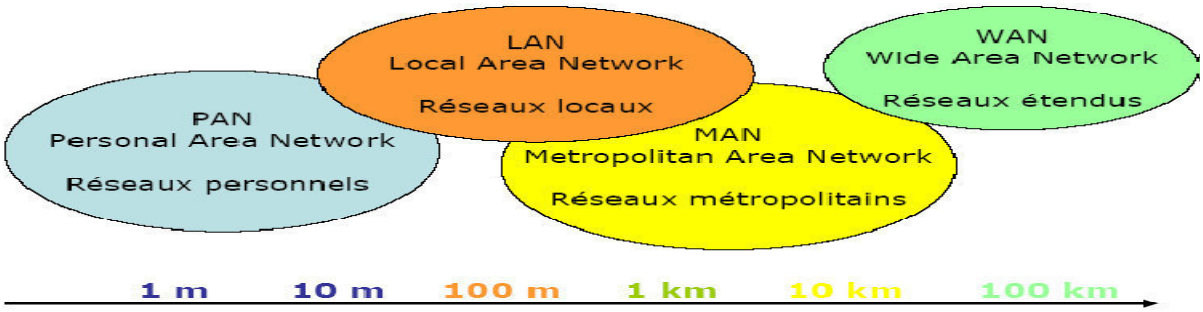

Figure I.1: Classification des réseaux informatiques selon leur étendu. [3]

I.2.2 Classification des réseaux selon la topologie : [4]

La topologie logique, par opposition à la topologie physique, représente la façon dont les données transitent dans les lignes de communication. Les topologies logiques les plus courantes sont Ethernet, Token Ring et FDDI (*Fiber Distributed Data Interface)*.

- **Topologie en bus :**

Une topologie en bus est l'organisation la plus simple d'un réseau. En effet, dans une topologie en bus tous les ordinateurs sont reliés à une même ligne de transmission par l'intermédiaire de câble, généralement coaxial. Le mot « bus » désigne la ligne physique qui relie les machines du réseau.

Cette topologie a pour avantage d'être facile à mettre en oeuvre et de posséder un fonctionnement simple. En revanche, elle est extrêmement vulnérable étant donné que si l'une des connexions est défectueuse, l'ensemble du réseau en est affecté.

Figure I.2: Topologie en bus.

- **Topologie en étoile :**

Dans une topologie en étoile, les ordinateurs du réseau sont reliés à un système matériel central appelé **concentrateur** (en anglais hub, littéralement moyen de roue). Il s'agit d'une boîte comprenant un certain nombre de jonctions auxquelles il est possible de raccorder les câbles réseau en provenance des ordinateurs. Celui-ci a pour rôle d'assurer la communication entre les différentes jonctions.

Contrairement aux réseaux construits sur une topologie en bus, les réseaux suivant une topologie en étoile sont beaucoup moins vulnérables car une des connexions peut être débranchée sans paralyser le reste du réseau. Le point névralgique de ce réseau est le concentrateur, car sans lui plus aucune communication entre les ordinateurs du réseau n'est possible.

En revanche, un réseau à topologie en étoile est plus onéreux qu'un réseau à topologie en bus car un matériel supplémentaire est nécessaire (le hub).

Figure I.3: Topologie en étoile.

- **Topologie en anneau :**

Dans un réseau possédant une topologie en anneau, les ordinateurs sont situés sur une boucle et communiquent chacun à leur tour.

En réalité, dans une topologie anneau, les ordinateurs ne sont pas reliés en boucle, mais sont reliés à un **répartiteur** (appelé MAU, Multistation Access Unit) qui va gérer la communication entre les ordinateurs qui lui sont reliés en impartissant à chacun d'entre-eux un temps de parole.

Les deux principales topologies logiques utilisant cette topologie physique sont Token ring (anneau à jeton) et FDDI.

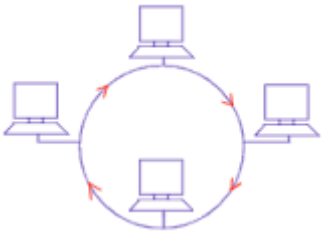

Figure I.4: Topologie en anneau.

- **Topologie en arbre :**

Aussi connu sous le nom de topologie hiérarchique, le réseau est divisé en niveaux. Le sommet, le haut niveau, est connectée à plusieurs nœuds de niveau inférieur, dans la hiérarchie. Ces nœuds peuvent être eux-mêmes connectés à plusieurs nœuds de niveau inférieur. Le tout dessine alors un arbre, ou une arborescence.

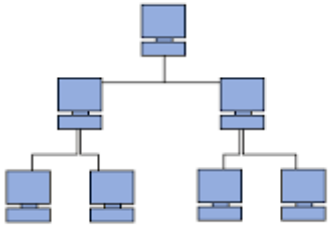

Figure I.5: Topologie en arbre.

Topologie maillée :

Une topologie maillée, est une évolution de la topologie en étoile, elle correspond à plusieurs liaisons point à point. Une unité réseau peut avoir (1,N) connexions point à point vers plusieurs autres unités. Chaque terminal est relié à tous les autres. L'inconvénient est le nombre de liaisons nécessaires qui devient très élevé. Cette topologie se rencontre dans les grands réseaux de distribution (Exemple : Internet). L'information peut parcourir le réseau suivant des itinéraires divers, sous le contrôle de puissants superviseurs de réseau, ou grâce à des méthodes de routage réparties. L'armée utilise également cette topologie, ainsi, en cas de rupture d'un lien, l'information peut quand même être acheminée. Elle existe aussi dans le cas de couverture Wi-Fi. On parle alors bien souvent de topologie mesh mais ne concerne que les routeurs WiFi.

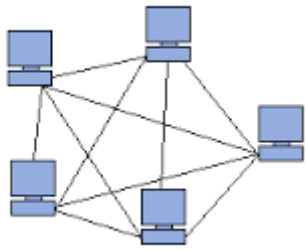

Figure I.6: Topologie en maillée.

I.3 L'architecture des réseaux :

Pour créer un réseau, il faut utiliser un grand nombre de composants matériels et logiciels souvent conçus par des fabricants différents. Pour que le réseau fonctionne, il faut que tous ces appareils soient capables de communiquer entre eux.

Pour faciliter cette interconnexion, il est apparu indispensable d'adopter des normes. Ces normes sont établies par différent organismes de normalisation.

I.3.1Le modèle OSI (Open System Interconnection) :

I.3.1.1 Définition :

Pour faciliter l'interconnexion des systèmes, un modèle appelé OSI (Open Systems Interconnexion) a été défini par l'ISO (International Standards Organisation). Ce modèle appelé modèle de référence OSI par ce qu'il traite de la connexion entre système ouvert à la communication avec d'autre système et aussi il définit ainsi un langage commun pour le monde des télécommunications et de l'informatique.

Le modèle OSI répartit les protocoles utilisés selon sept couches logicielles.

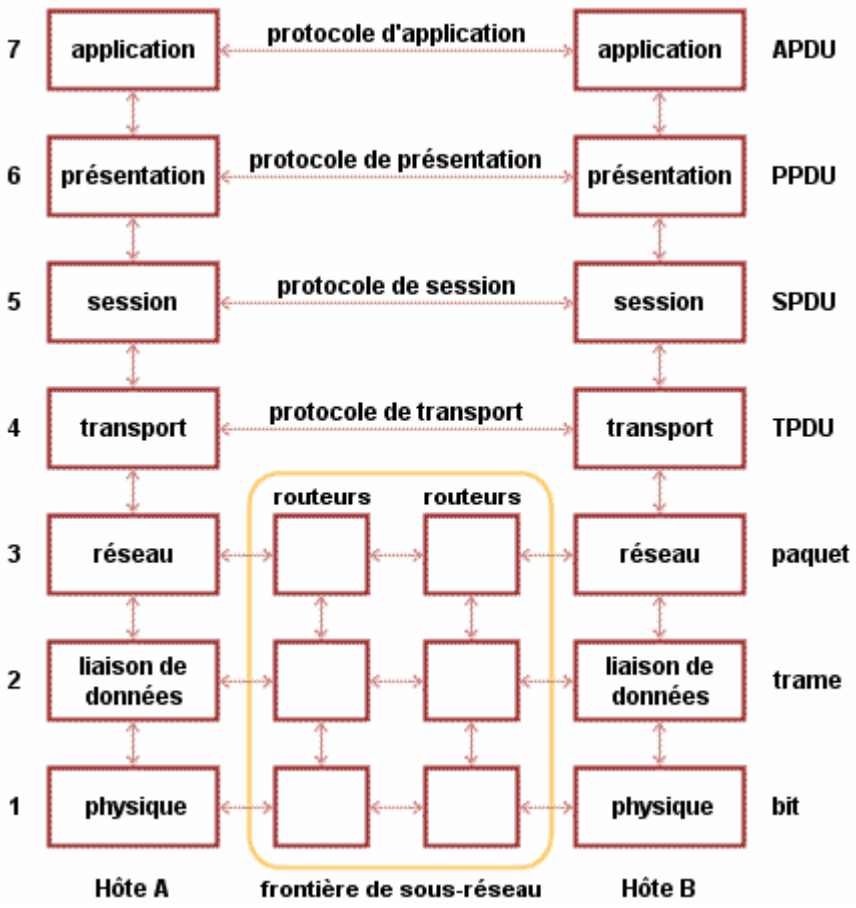

Figure I.7 : Architecture du modèle OSI. [5]

I.3.1.2 Les rôles des différentes couches : [6]

- **Couche physique :** Assure le transfert de bits, on trouve dans cette couche:
  - L'étude des interfaces de connexion.
  - L'étude des modems, des multiplexeurs et concentrateurs.
- **Couche liaison de données :** Responsable de l'acheminement d'unités de données appelées **trames** en assurant la meilleure qualité de transmission possible. Le protocole standard est HDLC
- **Couche réseaux :** Transporte des unités de données de taille fixe appelée **paquets.** Exemples de protocoles standards : X25 et IP.
- **Couche transport :** Transport des unités de données appelées **messages.** Le protocole TCP et UDP et TCP/IP
- **Couche session :** Assure l'établissement et le contrôle de séances de communication.
- **Couche présentation :** Présentation globale et unifiée de l'information, interprétation, cryptage, compression de données.
- **Couche Application :** Application spécifiques, comme Telnet, FTP, rlogin, SSH….

I.3.1.3 Avantages du modèle OSI : [7]

- Les interfaces sont uniformisées
- Il réduit la complexité
- Il assure une parfaite compatibilité des différentes technologies.
- Il permet l'accélération des progrès technologiques en matière de réseau.
- Il permet de diviser les communications sur le réseau en éléments plus petits et plus simples.
- Il uniformise les éléments du réseau afin de permettre le développement et le soutien multi constructeurs.
- Il permet à différents types de matériel et de logiciel réseau de communiquer entre eux.

Il empêche les changements apportés à une couche d'affecter les autres couches, ce qui assure un développement plus rapide.

I.3.2 Le modèles TCP/IP : [8]

Le modèle TCP/IP, inspiré du modèle OSI, reprend l'approche modulaire (utilisation de modules ou couches) mais en contient uniquement quatre :

Comme on peut le remarquer, les couches du modèle TCP/IP ont des tâches beaucoup plus diverses que les couches du modèle OSI, étant donné que certaines couches du modèle TCP/IP correspondent à plusieurs couches du modèle OSI.

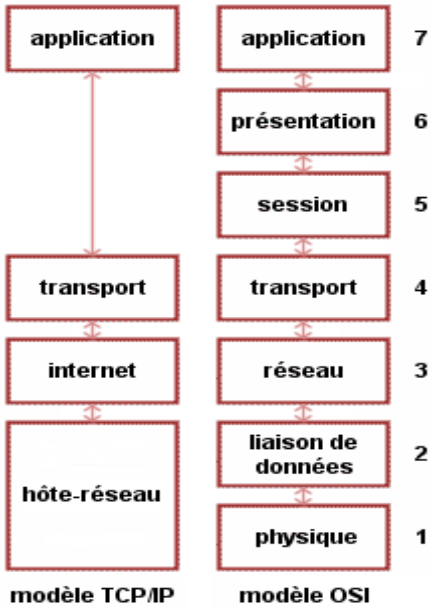

Figure I.8 : Comparaison entre le modèle OSI et le TCP/IP. [9]

I.3.2.1 Les rôles des différentes couches : [8]

- **Couche Accès réseau** : elle spécifie la forme sous laquelle les données doivent être acheminées quel que soit le type de réseau utilisé
- **Couche Internet** : elle est chargée de fournir le paquet de données (datagramme)
- **Couche Transport** : elle assure l'acheminement des données, ainsi que les mécanismes permettant de connaître l'état de la transmission
- **Couche Application** : elle englobe les applications standard du réseau (Telnet, SMTP, FTP, ...) Voici les principaux protocoles faisant partie de la suite TCP/IP :

I.4 Les protocoles :

C'est un ensemble de règles de communication qui permet à deux ou plusieurs entités (ordinateurs, applications logicielles, périphériques d'ordinateur, etc.) d'échanger des données entre elles.

I.4.1 Le protocole TCP / IP ? (Transmission Control Protocol/Internet Protocol) : [10]

Le protocole TCP/IP provient des noms des deux protocoles majeurs TCP et IP, il représente d'une certaine façon l'ensemble des règles de communication sur internet et se base sur la notion d'adressage IP, c'est-à-dire le fait de fournir une adresse IP à chaque machine du réseau afin de pouvoir acheminer des paquets de données. Etant donné que la suite de protocoles TCP/IP a été créée à l'origine dans un but militaire, elle est conçue pour répondre à un certain nombre de critères parmi lesquels :

- Le fractionnement des messages en paquets
- L'utilisation d'un système d'adresses
- L'acheminement des données sur le réseau (routage)
- Le contrôle des erreurs de transmission de données

I.4.2 Le protocole IP : [10]

Il assure la définition, la fragmentation, le réassemblage et le routage des datagrammes. Il transfert des données en mode datagramme, c'est-à-dire que les paquets sont traités indépendamment les uns des autres. Le but de l'IP est de pouvoir construire un réseau mondial en s'adaptant à tout type de support physique.

Sur Internet, les ordinateurs communiquent entre eux grâce au protocole IP, qui utilise des adresses numériques appelées adresses IP. Ces numéros permettent aux ordinateurs de ce reconnaître sur le réseau, et ces adresses sont uniques sur un même réseau.

I.4.3 Le protocole UDP (User Datagram Protocol) : [10]

UDP est un protocole de la couche transport, en mode datagramme ayant les caractéristiques suivantes :

- Sans connexion
- Support de transmission non fiable
- Perte de datagrammes possibles
- L'ordre des datagrammes peut être inversé pendant le trajet sur le réseau (s'ils ne prennent pas la même route)

Les données sont envoyées dès que l'application effectue une écriture. Chaque écriture de l'application génère un datagramme UDP, et les lectures des datagrammes depuis l'application se font sur des datagrammes complets.

I.4.3.1 Le protocole TCP (Transmission Control Protocol) : [10]

Contrairement à UDP, le protocole TCP offre un service de flux d'octets orienté connexion et fiable.

Il permet d'avoir une connexion entre programmes ayant des propriétés bien plus complexes que les datagrammes UDP. TCP propose, par l'intermédiaire d'acquittements, d'avoir un contrôle de perte de paquet avec délivrance des données à l'application dans l'ordre d'envoi.

- TCP temporise les données envoyées sous forme de segments dont la taille est adaptée aux conditions présentes sur le réseau
- Chaque segment est acquitté par le destinataire pour avoir un transport fiable
- La perte de paquets est contrôlée à l'aide de temporisations
- Les données sont transmises "en ordre" à l'application
- TCP propose un mécanisme de fenêtrage pour ne pas saturer le mémoire de l'application
- Les connexions TCP sont bidirectionnelles.

I.5 L'architecture client /serveur:

I.5.1 Présentation de l'architecture à deux niveaux :

L'architecture à deux niveaux (appelée aussi architecture 2-tier, tier signifiant étage en anglais) caractérise les systèmes client/serveur dans lesquels le client demande la ressource et le serveur la lui fournit directement (sans intermédiaire). Cela signifie que le serveur ne fait appel à une autre application afin de fournir le service.

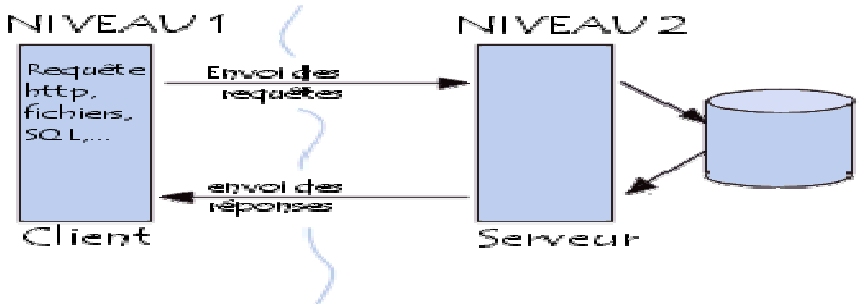

Figure I.9: Le client/serveur à deux niveaux. [11]

I.5.2 Présentation de l'architecture à trois niveaux :

Dans l'architecture à trois niveaux (appelé aussi architecture 3-tier) il existe un intermédiaire, cette architecture est généralement partagée entre :

- **Le client** : qui demande la ressource.
- **Le serveur d'application** (ou midlleware) : C'est le serveur chargé de fournir la Ressource mais faisant appel à un autre serveur.
- **Le serveur secondaire** : celui qui fournit le service au premier serveur (souvent un serveur de base de données).

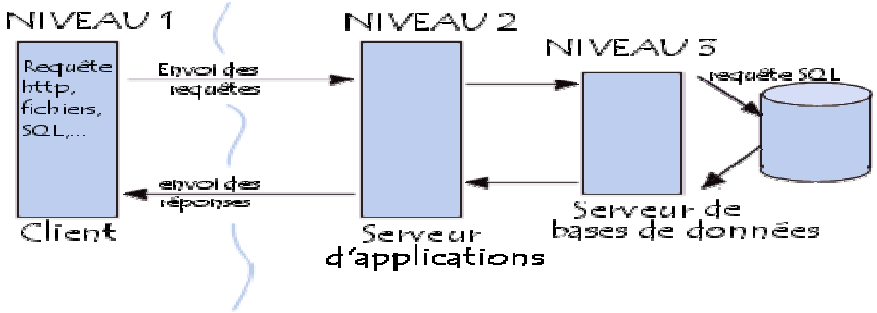

Figure I.10: Le client/serveur à trois niveaux. [11]

Remarque : l'architecture à deux niveaux est donc une architecture client/serveur dans laquelle le serveur est polyvalent, i.e qu'il est capable de fournir directement l'ensemble des ressources demandées par le client. Dans l'architecture à trois niveaux par contre les applications au niveau serveur sont délocalisées, i.e. Que chaque serveur est spécialisé dans une tâche (serveur Web/serveur de base de données par exemple). Ceci nous conduit à définir l'architecture multi-niveaux, qui est une architecture à N niveaux (tout comme l'architecture à 3-tier mais avec plusieurs serveurs intermédiaires).

Objectifs de cette architecture :

- Une plus grande flexibilité/souplesse.
- Une plus grande sécurité (la sécurité peut être définie pour chaque service).
- De meilleures performances (les tâches sont partagées).

Conclusion :

Les réseaux informatiques sont entrain de s'imposer comme la solution du partage d'information, grâce notamment à la minimisation des coûts et à l'augmentation des performances des systèmes.

Nous avons consacré ce chapitre à la présentation de quelques notions générale sur les réseaux informatiques, ainsi que les deux architectures les plus utilisé OSI et TCP/IP.

Le chapitre suivant sera consacrée à la sécurité informatique ainsi les différentes techniques de protection contre les menaces informatiques.

CHAPITRE II

SECURITE INFORMATIQUE

II. Introduction :

Les entreprises ouvrent leur système d'information à leurs partenaires et fournisseurs via internet. Cette merveilleuse ouverture, qui permet de faciliter la communication engendre malheureusement des risques importants dans le domaine de sécurité de l'entreprise.

Les ordinateurs connectés de l'entreprise ont des failles qui peuvent être exploitées par des **hackers**[1] pour réaliser ses attaques. Les conséquences de ces attaques peuvent être lourdes (perte d'information, vol d'information, perte financière, accès à des informations confidentielles, etc..). Par conséquent, il est important d'être conscient de ces menaces et de prendre des mesures adéquates afin de se protéger de ces attaques, pour cela, les administrateurs sécurisent de plus en plus leur système d'informations en utilisant diverses solutions comme : mot de passe, pare feu, la cryptographie, les scanners de vulnérabilités, antivirus, et les systèmes de détection d'intrusions. Nous détaillons dans la suite chacune de ces méthodes et nous soulignons leurs limites.

II.1 Objectifs de la sécurité informatique:

La sécurité informatique à plusieurs objectifs, bien sûr liés aux types de menaces ainsi qu'aux types de ressources, etc. Néanmoins, les points principaux sont les suivant :

- Empêcher la divulgation non autorisée des données.
- Empêcher la modification non autorisée des données.
- Empêcher l'utilisation non autorisée des ressources réseaux ou informatique de façon générale.

II.2 Services Principaux de la sécurité réseau :

Pour remédier aux failles et pour contrer les attaques, la sécurité informatique se base sur un certain nombre de services qui permettent de mettre en place une réponse appropriée à chaque menace. Les principaux services sont :

-La confidentialité : La confidentialité consiste à rendre l'information inintelligible à d'autres personnes que les seuls acteurs de la transaction.

-L'intégrité de données :Vérifier l'intégrité des données consiste à déterminer si les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).

-La disponibilité : Permettant de maintenir le bon fonctionnement du système informatique.

[1] Le terme « hacker » est souvent utilisé pour designer un pirate informatique.

-La non répudiation : Permettant de garantir qu'une transaction ne peut être niée.

-L'authentification : L'authentification consiste à assurer l'identité d'un utilisateur, c'est-à-dire de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être.

II.3 Objectifs des hackers :

Les motivations des hackers (Selon les individus) peuvent être multiples. On y retrouve :

- Vérification de la sécurité d'un système.
- Espionnage.
- L'attirance de l'interdit.
- Le désir d'argent (voler un système bancaire par exemple).
- Le besoin de renommées (impressionner des amis).
- L'envie de nuire.
- Pour apprendre.
- Etc.

II.4 Politique des hackers :

La Meilleure façon de protéger son système informatique est de procéder de la même manière que les pirates, afin de cartographier les vulnérabilités[2] du système :

II.4.1 Reconnaissance du système :

Avant qu'un hacker ne s'introduit dans le système informatique, il cherche dans un premier temps les failles c'est-à-dire, des vulnérabilités visibles à la sécurité du système : dans les protocoles, les systèmes d'exploitation, les applications, ou même le personnel d'une organisation. Pour cela, il utilise plusieurs moyens :

- **Reconnaissance passive** :

Le hacker partage librement ses découvertes et évite la destruction intentionnelle des données. Une des attaques les plus répandues l'écoute du trafic **Sniffing** .Le principe consiste à installer une sonde sur le réseau pour capter le trafic et le sauvegarder dans des fichiers journaux. L'analyse de ces fichiers permet de connaître les machines installées sur le réseau, et de déterminer les ports ouverts, et les systèmes d'exploitations utilisées.

- **Reconnaissance active** :

A ce niveau, l'attaquant ne se restreint pas à inspecter les données échangées entre les différents hôtes cependant, il initie lui même des connexions réseau pour tester le

[2] Vulnérabilité : trou de sécurité (en anglais security hote) désignant les failles de sécurité.

comportement des machines, il cherche des informations précises concernant les hôtes accessibles,l'emplacement des routeurs et des par feux. Parmi les techniques les plus utilisées pour acquérir ces informations, nous évoquons les utilitaires : **Ping , TraceRoute** et **Nmap.**

II.4.2 Exploitation du système :

Une fois le hacker a localisé les applications vulnérables, il exploite ensuite leurs faiblesses. L'intrus cherche à gagner un accès au réseau, cible en lançant diverses attaques. (Dans la suite nous détaillons quelques attaques).

II.4.3 Préservation d'accès :

Les attaquants installent des **portes dérobées**[3] pour pouvoir retourner facilement aux systèmes compromis. Par exemple, ils créent de nouveaux comptes et les utilisent lors des prochains accès. Cette procédure est facilement détectable si un administrateur vérifie constamment l'intégrité des fichiers.

II.4.4 Effacement des traces : [8] [10]

Une fois la porte dérobée est crée, l'attaquant cherche aussitôt à effacer ses traces. Il essaie de restituer les mêmes propriétés des fichiers (date de création, de modification, dernière utilisation, etc..), pour garder la même signature, ceci force les administrateurs à enregistrer les événements sur des machines distinguées pour mieux protéger les fichiers de sécurité.

II.5 Différents types d'attaques :

II.5.1 Les attaques réseaux :

Les attaques réseaux les plus connues aujourd'hui sont :

- **Spoofing IP :**

Le spoofing IP est une technique permettant à un pirate d'envoyer à une machine des paquets semblant provenir d'une adresse IP autre que celle de la machine du pirate. Le spoofing IP n'est pas pour autant un changement d'adresse IP. Plus exactement, il s'agit d'une mascarade de l'adresse IP au niveau des paquets émis, c'est-à-dire une modification des paquets envoyés afin de faire croire au destinataire qu'ils proviennent d'une autre machine.

- **Spoofing ARP :**

Le spoofing ARP est une technique qui modifie le cache ARP. Le cache ARP contient une association entre les adresses matérielles des machines et les adresses IP, l'objectif du

[3] Porte dérobée : faille créer par le pirate.

pirate est de conserver son adresse matérielle, mais d'utiliser l'adresse IP d'un hôte approuvé. Ces informations sont simultanément envoyées vers la cible et vers le cache. A partir de cet instant, les paquets de la cible seront routés vers l'adresse matérielle du pirate.

- **Spoofing DNS**

Le système DNS (Domain Name System) a pour rôle de convertir un nom de domaine en son adresse IP et réciproquement, à savoir : convertir une adresse IP en un nom de domaine. Cette attaque consiste à faire parvenir de fausses réponses aux requêtes DNS émises par une victime. Il existe deux types de méthode:

- **DNS ID spoofing** L'attaquant essaie de répondre à un client en attente d'une réponse d'un serveur DNS, avec une fausse réponse et avant que le serveur DNS ne réponde.
- **DNS Cache Poisoning** L'attaquant essaie d'empoisonner le cache (table de correspondance IP- nom _machine) du serveur DNS.

II.5.2 Les attaques applicatives :

- **Injection SQL :**

Les attaques par **injection de commandes SQL** sont des attaques visant les sites web s'appuyant sur des bases de données relationnelles.

Dans ce type de sites, des paramètres sont passés à la base de données sous forme d'une requête SQL. Ainsi, si le concepteur n'effectue aucun contrôle sur les paramètres passés dans la requête SQL, il est possible à un pirate de modifier la requête afin d'accéder à l'ensemble de la base de données, voire à en modifier le contenu.

Le principe de cette attaque consiste à injecter du code supplémentaire au sein des requêtes a fin de leurrer l'interpréteur SQL.

Par exemple contourner les mécanismes d'authentification via les entrées imprévues. Considérons par exemple la requête SQL (II.1). Cette requête permet de vérifier le mot de passe de l'utilisateur *$varNom* en consultant la table utilisateur. Seulement un intrus peut s'identifier avec un nom d'un utilisateur légitime puis entrer un mot de passe de la forme **cmoi OR TRUE.**

La requête SQL (II.1) se transforme en (II.2) et sera toujours vérifiée si l'utilisateur $varNom existe dans la table utilisateur.

select * from utilisateur where nom = $varNom and mot2passe = $varPasse (II.1)

select * from utilisateur where nom = $varNom and True (II.2)

L'attaquant peut créer son propre compte, en utilisant le nom d'un utilisateur légitime ***SELECT * from client where mon='joe ;insert into utilisateurs values('mon_login,'mon_password')'*** ***(II.3)***

- **Les bugs**

Tout logiciel comporte des bogues dont certains représentent des trous de sécurité ou des anomalies qui permettent de violer le système sur lequel tourne le programme. Si c'est un programme d'application réseau, ces trous peuvent être exploités à distance via Internet. Les plus connus de ces bugs et les plus intéressants, en ce qui concerne leur exploitation sont les buffers overflows.

Buffer overflow : consiste à mettre plus d'informations (et surtout d'autres informations) en mémoire que celle-ci n'est disposée à en recevoir.

Conséquences :

- Le débordement du buffer peut écraser l'adresse de retour au programme appelant : plantage (attaque de type déni de service).
- L'adresse de retour peut être remplacée par l'adresse d'un code malicieux.

II.5.3 Les attaques par déni de service :

Les attaques par déni de service (souvent abrégé ***DOS***, en anglais Denial of service) consistent à paralyser temporairement (rendre inactif pendant un temps donné) des serveurs afin qu'ils ne puissent être utilisés et consultés. Le but d'une telle attaque n'est pas de récupérer ou d'altérer des données, mais de nuire à des sociétés dont l'activité repose sur un système d'information. En l'empêchant de fonctionner.

- **La technique dite du smurf :**

La technique du *smurf* est basée sur l'utilisation de serveurs broadcast pour paralyser un réseau. Un serveur broadcast est un serveur capable de dupliquer un message et de l'envoyer à toutes les machines présentes sur le même réseau que lui. Le scénario d'une attaque est le suivant :

La machine attaquante envoie un **ping** à un (ou plusieurs) serveurs broadcast en falsifiant sa propre adresse IP (l'adresse à laquelle le serveur devrait théoriquement répondre par un pong) et en fournissant l'adresse IP de la machine cible. Lorsque le serveur broadcast va dispatcher le ping sur tout le réseau, toutes les machines du réseau vont répondre par un pong, que le serveur broadcast va rediriger vers la machine cible. Ainsi lorsque la machine attaquante adresse le ping à plusieurs serveurs broadcast situés sur des réseaux différents,

l'ensemble des réponses de tous les ordinateurs des différents réseaux vont être reroutées sur la machine cible.

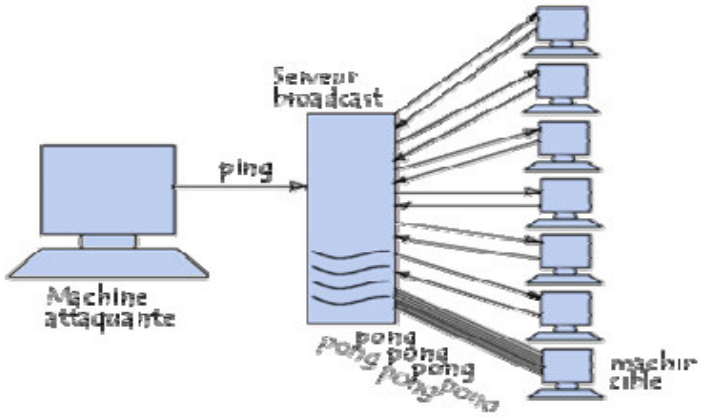

Figure II.1 : Exemple d'attaque.

- **SYN FLOOD :**

Son objectif est de rendre indisponible un service TCP offert sur une machine. Le principe de cette attaque est de créer des connexions TCP semi-ouvertes sur la machine cible afin de remplir la file d'attente où sont stockées les demandes d'ouverture de connexions. L'attaquant envoie un grand nombre de requêtes SYN à la machine cible et remplace son adresse source avec l'adresse d'une machine indisponible ou inexistante afin que les réponses SYN/ACK ne soient jamais reçue et que donc les messages ACK ne soient jamais générés, ce qui signifie que la file d'attente restera pleine. Les conséquences de cette attaque sont que toutes les requêtes arrivant sur le port TCP cible seront ignorées et de ce fait le service fourni sur ce port sera indisponible. Dans certains cas, la machine peut aussi devenir indisponible.

- **Fragmentation :**

L'attaquant sature la connexion en envoyant des fragmentations déclanchant des exceptions (faille de la pile TCP/IP de *Windows* 95 et 98).

II.5.4 Les attaques virales : [14]

Il existe principalement quatre types de menaces distinctes :

- **Virus** : Se reproduisent en infectant le corps de programmes hôtes
- **Vers** : Le vers se duplique et se propage à travers le réseau, par courrier électronique par exemple.

- **Chevaux de Troie** : Exécutent des tâches malignes en se cachant dans un programme sain. Il peut par exemple voler des mots de passe, copier des données, ou exécuter tout autre action nuisible.
- **Trappes (portes dérobées)** : Permet à un utilisateur externe de prendre le contrôle d'une application par des moyens détournés.

II.6 Outils de sécurité:

Nous avons constaté que les attaquants disposent de plusieurs moyens pour réussir leurs attaques. La disponibilité des outils d'attaques et la richesse des sources d'informations accentuent le risque des intrusions. Par conséquent, les administrateurs sécurisent de plus en plus leurs systèmes informatiques. Ils s'appuient sur diverses solutions comme les pares feux, la cryptographie, le mot de passe, les scanners de vulnérabilités et les systèmes de détection d'intrusions. Nous détaillons dans la suite chacune de ces méthodes et nous soulignons leurs limites.

II.6.1 Signature électronique et Certificat :

-Signature électronique :

Signature électronique est un code digital permet à la personne qui reçoit une information de contrôler l'authenticité de son origine , et également de vérifier que l'information en question est intacte. Aussi, les signatures électroniques permettent l'authentification et le contrôle de l'intégrité et également la non répudiation.

-Certificat :

Certificat est Document électronique, carte d'identité émis par une autorité de certification. Il valide l'identité des interlocuteurs d'une transaction électronique, associe une identité à une clé publique d'encryptions et fournit des informations de gestion sur le certificat et le détenteur.

II.6.2 Mots de passes :

Une personne peut être authentifiée par une combinaison d'une identification et d'un mot de passe, (code secret personnel).Le mot de passe doit posséder certaines caractéristiques qui sont : non trivial, difficile à deviner, régulièrement modifié. Cependant si l'attaquant accède au fichier de mot de passe, il pourra s'introduire dans le système sécurisé.

II.6.3 Firewall :

Un Firewall est un assemblage matériel (ordinateur) et des logiciels installés sur celui ci dont l'objectif principal est de protéger le réseau interne contre les accès et actions non autorisés en provenance de l'extérieur, en contrôlant le trafic entrant et sortant.

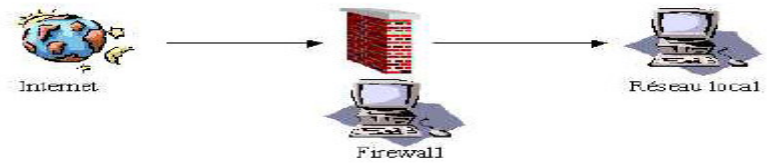

Figure II.2 : Placement d'un firewall.

Il existe plusieurs types de techniques de firewall :

- La technique de filtrage des paquets : chaque paquet d'informations entrant ou sortant est accepté ou rejeté selon des règles établies par l'utilisateur.
- La technique des serveurs Proxy : qui empêche l'extérieur de connaître les adresses internes du réseau.
- La technique des passerelles : qui fournissent des systèmes de sécurité pour établir des connexions TCP/IP entre l'extérieur et l'intérieur ou pour certains services comme *FTP* et *TELNET*.

Les inconvénients d'un firewall sont :

- Ne couvre pas tous les risques de sécurité. Par exemple il n'assure pas la confidentialité des informations, n'authentifie pas l'origine des informations, ne vérifie pas l'intégrité des informations, ne protège pas contre les attaques internes.
- Une très forte configuration de firewall augmente la sécurité mais peut alerter le fonctionnement du réseau.
- Le pirate peut détruire le système et ainsi permettre l'accès à tous les individus .C'est en général ce qui se passe.
- l'attaquant peut contourner le firewall

II.6.4 Scanners de vulnérabilités :

Les scanners de vulnérabilités automatisent la découverte des failles de sécurité. Ils sont utilisés par les attaquants pour localiser les faiblesses du réseau cible. De plus, les administrateurs peuvent en tirer profit pour corriger les vulnérabilités de leurs systèmes informatiques. Nous citons à titre d'exemple Nessus, Whisker et Saint.

Cependant les scanners présentent quelques limites qui peuvent être résumées en trois points : l'exhaustivité, la mise à jour et l'exactitude. En effet, malgré le grand nombre de

vulnérabilités détectées, les scanners d'aujourd'hui sont inaptes à déterminer toutes les faiblesses possibles.

De plus, la mise à jour de ces produits ne suit pas le rythme de la découverte des nouvelles vulnérabilités. Enfin, la modification des bannières des services scannés permet de dissuader facilement le scanner ce qui entraîne parfois un responsable de sécurité à chasser des vulnérabilités fantômes.

II.6.5 Réseau Privé Virtuel :

Un réseau VPN (*Virtuel Privat Network*) est un service qui permet d'établir des connexions sécurisées privées (c'est-à-dire faire un réseau privé) sur un réseau publique comme Internet.

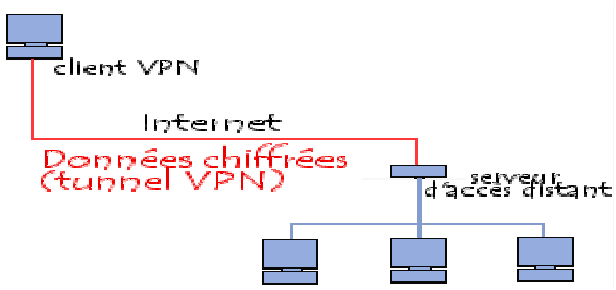

Figure II.3 : VPN.

VPN repose sur un protocole appelé protocole de ***tunneling***, le principe de tunneling consiste à construire un chemin virtuel après avoir identifié l'émetteur et le destinataire. Par la suite, la source chiffre les données et les achemine en empruntant ce chemin virtuel. Il existe trois types standard d'utilisation des VPN :

-Le VPN d'accès : permet à des utilisateurs itinérants d'accéder au réseau privé.

Figure II.4 : VPN d'accès.

-L'intranet VPN : est utilisé pour relier au moins deux intranet entre eux. Ce type de réseau est particulièrement utile au sein d'une entreprise possédant plusieurs sites distants.

Figure II.5 : VPN intranet.

-L'extranet VPN : une entreprise peut utiliser le VPN pour communiquer avec ses clients et ses partenaires. Elle ouvre alors son réseau local à ces derniers. Dans Ce cadre, il est fondamental que l'administrateur du VPN puisse tracer les clients sur le réseau et gérer les droits de chacun sur celui-ci.

Figure II.6 : VPN extranet.

II.6.6 Systèmes de détection d'intrusions (IDS) :

- **Contexte :**

Le concept de système de détection d'intrusions a été introduit en 1980 par *Anderson*. Mais le sujet n'a pas eu beaucoup de succès. Il a fallu attendre la publication d'un modèle de détection d'intrusions par *Denning* en 1987 pour marquer réellement le départ du domaine. En 1988, il existait au moins trois prototypes.

La recherche dans le domaine s'est ensuite développée, le nombre de prototypes s'est énormément accru. Le gouvernement des États-Unis a investi des millions de dollars dans ce type de recherches dans le but d'accroître la sécurité de ses machines.

- **Intrusion :**

Une intrusion est toute utilisation d'un système informatique à des fins autres que celles prévues, généralement dues à l'acquisition de privilèges de façon illégitime.

Actuellement les ***IDS*** sont très populaires à cause de :

-l'évolution continue des attaques.

-l'apparition de nouvelles attaques.

- la nécessite de détecter et réagir le plus vite possible aux attaques survenant dans le réseau.

Snort **[66]** est un exemple de ***IDS*** Open Source disponible au grand public basé sur l'approche par connaissance (nous détaillons cette approche dans le chapitre suivant).

Conclusion :

Nous avons constaté dans ce chapitre, que la sécurité réseau est un point primordial. Les administrateurs déploient des solutions de sécurité efficaces, capables de protéger le réseau de l'entreprise. Dans ce contexte, les IDS constituent une bonne alternative pour mieux protéger le réseau informatique. Dans le chapitre suivant nous détaillerons beaucoup plus la notion de la cryptographie.

CHAPITRE III

LA CRYPTOGRAPHIE

Introduction :

La Cryptographie est aujourd'hui essentielle pour le développement du commerce électronique, des cartes à puce, de la téléphonie mobile, et particulièrement cruciale dans le secteur bancaire.

Dans ce chapitre nous allons présenter les différents types algorithmiques de cryptage, a savoirs les méthodes de la cryptographie.

III.1 Cryptage et décryptage : [22]

Les données lisibles et compréhensibles sans intervention spécifique sont considérées comme du texte en clair. La méthode permettant de dissimuler du texte en clair en masquant son contenu est appelée le cryptage. Le cryptage consiste à transformer un texte normal en charabia inintelligible appelé texte chiffré.

Cette opération permet de s'assurer que seules les personnes auxquelles les informations sont destinées pourront y accéder. Le processus inverse de transformation du texte chiffré vers le texte d'origine est appelé le décryptage.

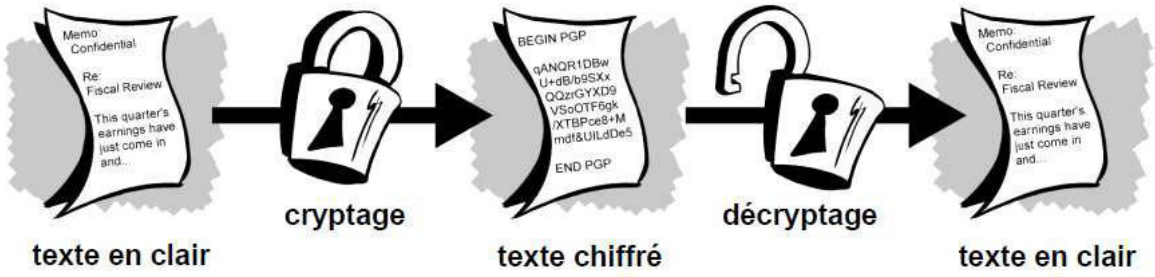

Figure III.1 Cryptage et décryptage [22]

III.1.1 Terminologie : [23]

- **Alphabet A :** ensemble fini de symboles utilisés pour écrire les messages.
- **Message clair m :** chaîne de caractères composée de lettres de l'alphabet A et dont on veut en général conserver la confidentialité. On note M l'ensemble de tous les messages clairs possibles.
- **Message crypté c :** chaîne de caractères composée de lettres de l'alphabet A, correspondant à un message clair, et dont la diffusion à des entités non autorisées ne

doit pas dévoiler pas d'information sur ce message clair. On note C l'ensemble de tous les messages cryptés

- **Cryptage :** transformation d'un message clair en un message crypté.
- **Décryptage :** transformation inverse du cryptage qui permet de retrouver à partir d'un message crypté, le message clair correspondant.
- **Signature s :** chaîne de caractères associées à un message donné (et aussi possiblement à une entité) et le caractérisant.
- **Transformation Tk :** fonction qui associe à un message clair ou crypté, une autre donnée qui peut être un message clair, crypté ou une signature. En général, ce sont des fonctions qui dépendent de clés.
- **Clé k :** donnée supplémentaire permettant de construire les fonctions de cryptage et de décryptage. Sans connaissance de la clé de décryptage, le décryptage doit être impossible. On note K l'ensemble de toutes les clés.
- **Protocole :** description de l'ensemble des données nécessaires pour mettre en place le mécanisme de cryptographie : ensemble des messages clairs, des messages cryptés, des clés possibles, des transformations...

III.1.2 La cryptologie :

La cryptologie est une science mathématique qui comporte deux branches : la cryptographie et la cryptanalyse.

III.1.2.1 La cryptographie :

La cryptographie st la science qui utilise les mathématiques pour le cryptage et le décryptage de données.

Elle vous 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.

A. **La cryptographie traditionnelle:** est l'étude des méthodes permettant de transmettre des données de manière confidentielle. Afin de protéger un message, on lui applique une transformation qui le rend incompréhensible ; c'est ce qu'on appelle le chiffrement, qui à partir d'un texte en claire donne un texte chiffré ou cryptogramme. Inversement, le déchiffrement est l'action qui permet reconstruire le texte en claire à partir du texte chiffré.

B. La cryptographie moderne: les transformations en question sont des fonctions mathématiques appelées algorithmes cryptographiques qui dépendent d'un paramètre appelé clef.

- **Méthode Symétrique (DES) :**

C'est un algorithme de chiffrement à clef secrète. La clef sert donc à la fois à chiffrer et à déchiffrer le message. Cette clef a ici une longueur de 64 bits, c'est-à-dire 8 caractères, mais dont seulement 56 bits sont utilisés. On peut donc éventuellement imaginer un programme testant l'intégrité de la clef en exploitant ces bits inutilisés comme bits de contrôle de parité. L'entière sécurité de l'algorithme repose sur les clefs puisque l'algorithme est parfaitement connu de tous. La clef de 64 bits est utilisée pour générer 16 autres clefs de 48 bits chacune qu'on utilisera lors de chacune des 16 itérations du D.E.S.. Ces clefs sont les mêmes quel que soit le bloc qu'on code dans un message.
Cet algorithme est relativement facile à réaliser matériellement et certaines puces chiffrent jusqu'à
1 Go de données par seconde. Pour les industriels, c'est un point important notamment face à des algorithmes asymétriques, plus lents, tels que l'algorithme R.S.A.

- **Méthode Asymétrique (RSA) :**

L'algorithme le plus célèbre d'algorithme à clef publique a été inventé en 1977 par Ron Rivest, Adi Shamir et Len Adleman, à la suite de la publication de l'idée d'une cryptographie à clef publique par Diffie et Hellman. Il fut appelé RSA, des initiales de ces inventeurs.
RSA est basé sur la difficulté de factoriser un grand nombre en produit de deux grands facteurs premiers.

- **Généralité des clés :**

1. Choisir deux nombre premiers de grande taille **p** et **q**

2. Calculer **n = p x q**

3. Calculer **Φ(n) = (p-1) (q-1)**

4. Choisir un entier aléatoire **e < Φ(n)** tel que **e** et **Φ(n)** soient premiers entre eux

PGCD(e, Φ(n)) = 1; 1 < e < Φ(n)

5. Trouver un entier **d** tel que **e x d ≡ 1 mod Φ(n)**

d= e^{-1} mod Φ(n)

6. La clé publique est **KU = {e, n}**

7. La clé privée est **KR = {d, n}**

8. Les nombres **p** et **q** ne sont plus utiles et peuvent être écartés (mais jamais révélés)

- **Exemple de RSA :**

Prendre deux nombres premiers *p* et *q*. En cryptographie réelle on choisira de très grands nombres, de 150 chiffres décimaux chacun. Nous allons donner un exemple avec *p* = 13 et *q* = 11.

1. Calculer **n = pxq,** soit dans notre exemple **n =13x11=143.**
2. Calculer **Φ(n) = (p −1) (q − 1) =12x10=120**
3. Prendre un petit entier **e<120**, et premier avec **120**, soit **e =17**. Dans la pratique.
4. Déterminer **d** tq **d.e = 1 mod 120 et d < 120**

Dans notre exemple **d = 113 car d.e = 113 x 17 = 1921 = 16 x 120 + 1**

5. La clé publique **KU = {17, 143}**
6. La clé privée **KR = {113, 143}**

III.1.2.2 Les quatre buts de la cryptographie : [23]

- **Confidentialité :** mécanisme pour transmettre des données de telle sorte que seul le destinataire autorisé puisse les lire.
- **Intégrité :** mécanisme pour s'assurer que les données reçues n'ont pas été modifiées durant la transmission.
- **Authentification :** mécanisme pour permettre d'identifier des personnes ou des entités et de certifier cette identité.
- **Non-répudiation :** mécanisme pour enregistrer un acte ou un engagement d'une personne ou d'une entité de telle sorte que celle-ci ne puisse pas nier avoir accompli cet acte ou pris cet engagement.

III.1.2.3 Cryptanalyse :

La cryptanalyse est l'étude des procédés cryptographiques dans le but de trouver des faiblesses et en particulier, de pouvoir décrypter des textes chiffrés.

❖ **La Cryptanalyse classique :**

Implique une combinaison intéressante de raisonnement analytique, d'application d'outils mathématiques, de recherche de modèle, de patience, de détermination et de chance. Ces cryptanalystes sont également appelés des pirates.

❖ **Attaques classiques : [23]**

L'attaquant connaît les algorithmes de cryptage et décryptage.

1. Attaque à texte crypté uniquement : l'attaquant ne dispose que d'un ou plusieurs messages cryptés qu'il souhaite décrypter. C'est le type d'attaque le plus difficile.

2. Attaque à texte clair connu : l'attaquant dispose d'exemples de messages clairs avec les messages cryptés correspondants, ou d'une partie claire d'un message crypté. Le but est d'obtenir de l'information sur la clé.

3. Attaque à texte clair choisi : l'attaquant peut obtenir la version cryptée d'un certain nombre de messages clairs choisis, soit avant l'attaque (attaque hors ligne), soit au fur et à mesure (attaque en ligne).
Le but est encore d'obtenir de l'information sur la clé.

4. Attaque à texte crypté choisi : l'attaquant peut obtenir la version cryptée d'un certain nombre de messages clairs choisis, et aussi la version claire d'un certain nombre de messages cryptés choisis. On distingue encore entre attaques hors ligne et en ligne.

5. Attaque par le paradoxe des anniversaires : Il s'agit d'obtenir des collisions (utilisation deux fois d'une même valeur) pour obtenir de l'information. Si on utilise 2^n valeurs possibles, on peut espérer la première collision avec environ $2^{n/2}$ valeurs.

6. Attaque par précalcul : Il s'agit pour l'attaquant de précalculer des informations et de s'en servir pour identifier des messages ou des clés, cela nécessite plus de travail mais permet aussi plus de flexibilité. Un cas extrême est la recherche exhaustive.

7. Attaque de différentiation : Il s'agit d'une attaque qui permet de différencier le protocole de cryptage utilisé d'un protocole de cryptage parfait. Cela couvre les attaques citées précédemment et toutes les attaques à venir.

III.1.3 Mécanismes de la cryptographie : [22]

Un algorithme de cryptographie ou un chiffrement est une fonction mathématique utilisée lors du processus de cryptage et de décryptage. Cet algorithme est associé à une clé (un mot, un nombre ou une phrase), afin de crypter le texte en clair. Avec des clés différentes, le résultat du cryptage variera également. La sécurité des données cryptées repose entièrement sur deux éléments : l'invulnérabilité de l'algorithme de cryptographie et la confidentialité de la clé.

Un système de cryptographie est constitué d'un algorithme de cryptographie, ainsi que de toutes les clés et tous les protocoles nécessaires à son fonctionnement.
PGP est un système de cryptographie.

III.1.3.1 Fonctionnement de PGP : [22]

PGP est une combinaison des meilleures fonctionnalités de la cryptographie de clé publique et de la cryptographie conventionnelle. PGP est un système de cryptographie hybride.

Lorsqu'un utilisateur crypte du texte en clair avec PGP, ces données sont d'abord compressées. Cette compression des données permet de réduire le temps de transmission par modem, d'économiser l'espace disque et, surtout, de renforcer la sécurité cryptographique. La plupart des cryptanalystes exploitent les modèles trouvés dans le texte en clair pour casser le chiffrement. La compression réduit ces modèles dans le texte en clair, améliorant par conséquent considérablement la résistance à la cryptanalyse. Toutefois, la compression est impossible sur les fichiers de taille insuffisante ou supportant mal ce processus.

PGP crée ensuite une clé de session qui est une clé secrète à usage unique. Cette clé correspond à un nombre aléatoire, généré par les déplacements aléatoires de votre souris et les séquences de frappes de touches. Pour crypter le texte en clair, cette clé de session utilise un algorithme de cryptage conventionnel rapide et sécurisé. Une fois les données codées, la clé de session est cryptée vers la clé publique du destinataire. Cette clé de session cryptée par clé publique est transmise avec le texte chiffré au destinataire.

III.1.4 Cryptographie conventionnelle : [22]

En cryptographie conventionnelle, également appelée cryptage de clé secrète ou de clé symétrique, une seule clé suffit pour le cryptage et le décryptage.
La norme de cryptage de données (DES) est un exemple de système de cryptographie conventionnelle largement utilisé par le gouvernement fédéral des
Etats-Unis. La Figure en-dessus est une illustration du processus de cryptage conventionnel.

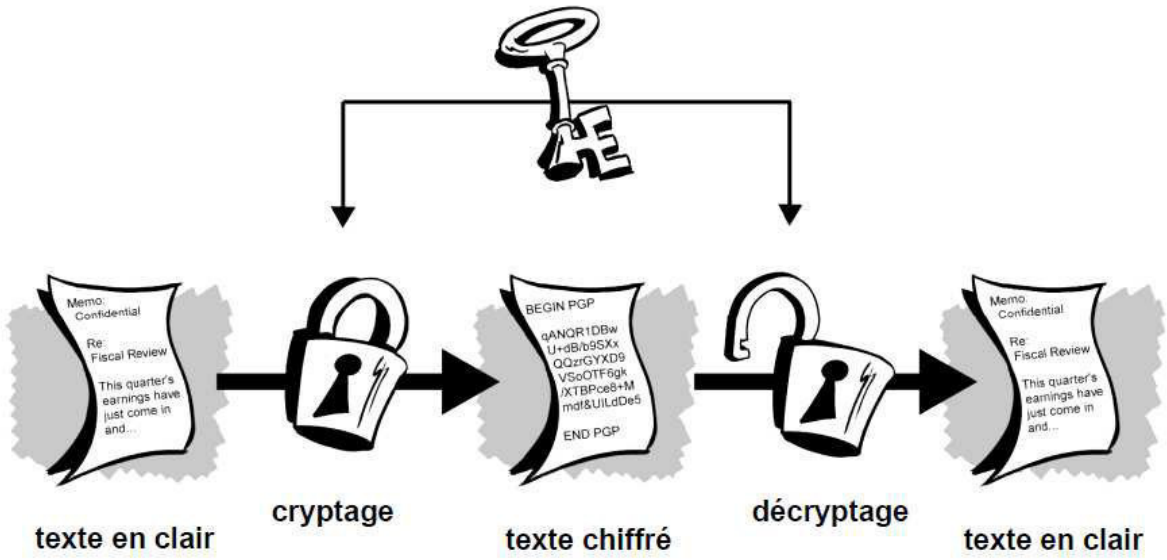

Figure III.2 Cryptage conventionnel [22]

III.1.5 Cryptographie de clé publique : [22]

La cryptographie de clé publique est un procédé asymétrique utilisant une paire de clés pour le cryptage : une clé publique qui crypte des données et une clé privée ou secrète correspondante pour le décryptage. Vous pouvez ainsi publier votre clé publique tout en conservant votre clé privée secrète. Tout utilisateur possédant une copie de votre clé publique peut ensuite crypter des informations que vous êtes le seul à pourvoir lire. Même les personnes que vous ne connaissez pas personnellement peuvent utiliser votre clé publique. D'un point de vue informatique, il est impossible de deviner la clé privée à partir de la clé publique. Tout utilisateur possédant une clé publique peut crypter des informations, mais est dans l'impossibilité de les décrypter. Seule la personne disposant de la clé privée correspondante peut les décrypter.

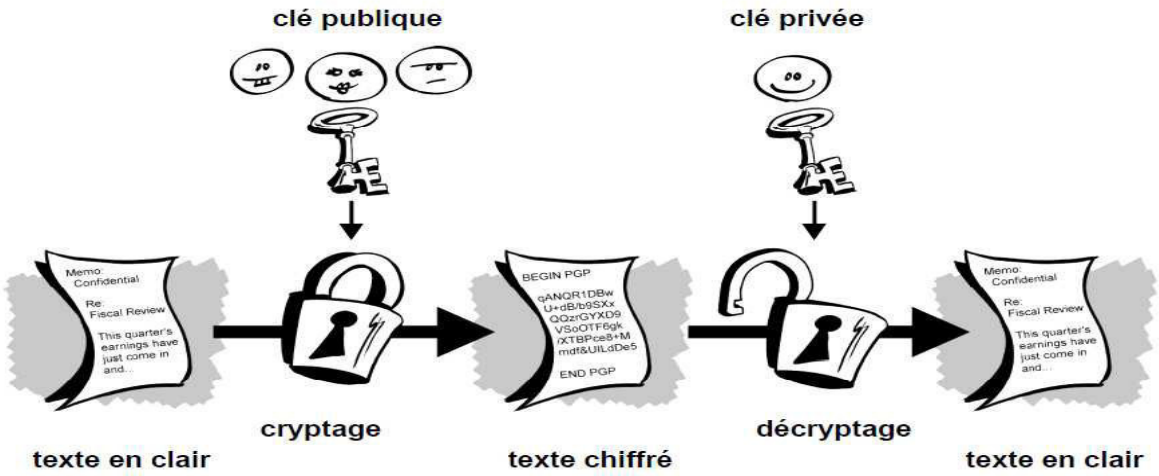

Figure III.3 Cryptage de clé publique [22]

III.1.6 La signature numérique : [22]

Les signatures électroniques font également partie de la panoplie des mécanismes indispensables à la transmission de documents dans un réseau. La signature a pour fonction d'authentifier l'émetteur. Celui-ci code le message de signature par une clé qu'il est le seul à connaître. La vérification d'une signature s'effectue par le biais d'une clé publique.

En utilisant l'algorithme RSA, l'émetteur signe le message M par M^e mod n, et le récepteur porte cette valeur à la puissance d pour vérifier que $(M^e)^d = M$. Si cette égalité se vérifie, la signature est authentifiée.

La méthode de base utilisée pour créer des signatures numériques est illustrée sur la figure suivante. Au lieu de chiffrer l'information en utilisant la clé publique d'autrui, vous la chiffrez avec votre propre clé privée. Si l'information peut être déchiffrée avec votre clé publique, c'est qu'elle provient bien de vous.

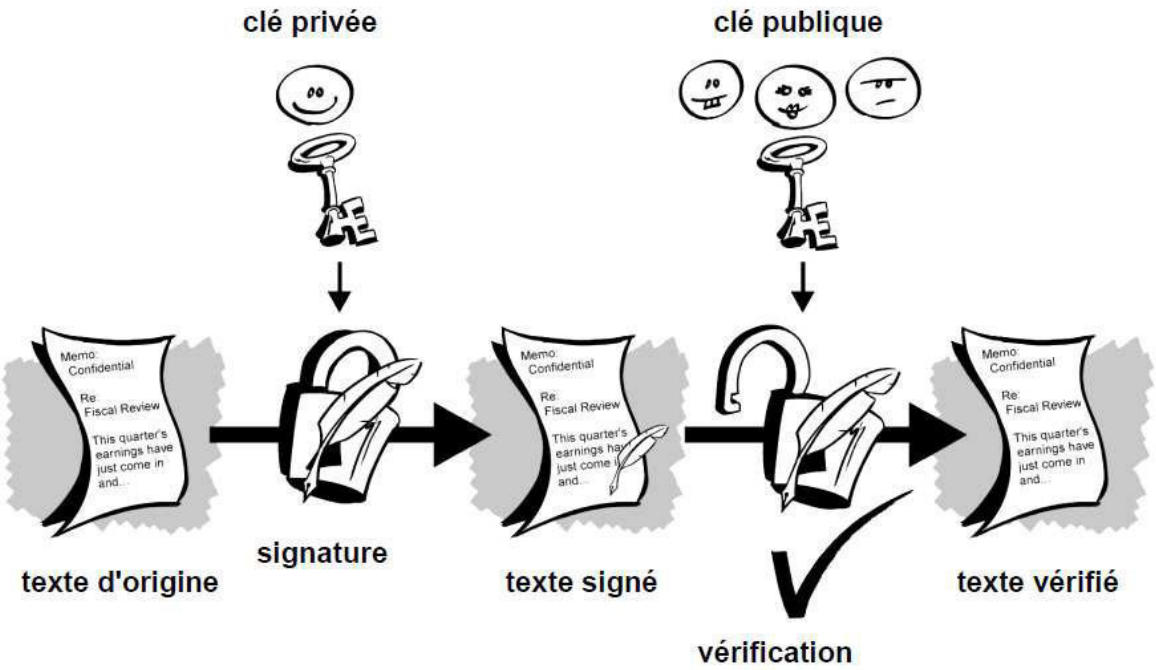

Figure III.4 Signatures numériques simples [22]

III.1.6.1 Principes de la signature numérique :

La signature numérique doit donc :

- Permettre d'identifier la personne expéditrice.
- Garantie que le document n'a pas été modifié entre l'expédition et réception il faut donc que :
    - La signature ne soit pas modifiable (identification).
    - La signature ne soit pas réutilisable (non répudiation et identification).
    - Le document signé soit inaltérable (intégrité de document).

Les plus célèbres techniques de signature sont les suivantes :

- MD5 (Message Digest #5), Ce sont des fonctions conçues par **Ron Rivest** qui produisent des empreintes de 128 bits.
- SHA-1 (Secure Hash Algorithm), de 1993, pour les fonctions de hachage. Cette technique permet de réaliser une empreinte de 160 bits.

Conclusion :

Dans ce chapitre nous avons traité les différentes méthodes de la cryptographie et son fonctionnement.

Dans Le chapitre suivant on va traiter le RSA d'une manière très détaillé à savoir ces points fort et ces points faibles.

CHAPITRE IV

L'ALGORITHME DE RSA

Introduction :

Le cryptage est historiquement l'une des premières applications de l'informatique. Ce domaine, qui était il y a encore quelques années, réservé aux militaires et au grandes entreprises, concerne aujourd'hui tous ceux qui souhaitent transmettre des données protégées, qu'ils soient professionnels ou particuliers. Pour cela, il existe de nombreuses méthodes de cryptage, mais peu d'entre elles sont reconnues comme sûres. La méthode RSA fait depuis longtemps partie de cette catégorie.

Dans ce mémoire, nous allons développer un petit programme de cryptage et décryptage basé sur ce chiffre. Le but ne sera pas de développer un programme au code « incassable », mais plutôt de comprendre comment fonctionne le cryptage RSA.

IV.1 L'algorithme RSA : [24]

Première étape : la création des clés de Bob[1] pour RSA :

Input *Bob: rien*

Output *Bob: Clé publique* ***(n, e)*** *; Clé privée* ***(n, d)***

begin

Choisir deux grands nombres premiers ***p*** *et* ***q***

Calculer N = p.q et l'indicateur d'Euler $\Phi(n) = (p-1)(q-1)$

Choisir un entier $1 < e < \Phi(n)$ *premier avec* $\Phi(n)$*, c.à.d.* $pgcd(e, \Phi(n)) = 1$

Calculer ***d*** *l'inverse de* $e \pmod{\Phi(n)}$

end

Algo.1 Algorithme RSA première étape.

Désormais, toute personne ayant accès à la clé publique de Bob peut lui envoyer des messages sécurisés. A aucun moment, Alice[2] n'est intervenue dans la création de la clé publique de Bob.

Deuxième étape : Protocole RSA :

Input *général: La clé publique* ***(n, e)***

Input *Bob: la clé privée* ***d***

Input *Alice: un message* ***x*** *avec* $x < n$

Output *Bob: le message* ***x*** *est reçu*

begin *Cryptage*

[1] Traditionnellement, on utilise ces deux prénoms en cryptographie
[2] Traditionnellement, on utilise ces deux prénoms en cryptographie

Alice calcule $y \equiv x^e \pmod N$

Alice envoie **y** *à Bob*

end *Cryptage*

begin *Décryptage*

Bob calcule $x \equiv y^d \pmod N$

end *Décryptage*

Algo.2 Algorithme RSA deuxième étape.

Le corollaire suivant donne une condition suffisante sur e et sur d pour faire en sorte que le cryptosystème RSA fonctionne pour tout x appartient a (Z/NZ)*.

IV.1.1. Fabrication des clés : [24]

- **La clé publique :**

Pour créer cette clé, il faut construire un quadruplet **(p,q,e,d)** tel que

- p et q sont deux grands nombres premiers (ils ne possèdent que deux diviseurs : 1 et eux-mêmes) distincts.
- On pose **n=p.q**
- **e** est un entier premier avec **Φ(n) = Φ(p) Φ(q) = (p-1)(q-1), Φ** étant appelée l'indicatrice d'Euler (c'est le nombre d'entiers inférieurs et premiers à n ; Φ(p.q)= Φ(p).Φ(q) car p et q sont premiers entre eux ; si p est premier alors Φ(p)=p-1).
- Voici l'algorithme servant à déterminer e :

Détermination_de_e*(entier* ***p****, entier* ***q****) : entier*

Début

Φ(n) ← (p-1)*(q-1)

*/** ***e > 2*** *on peut spécifier une valeur minimale de* ***e*** **/*

Pour ***e*** *allant de* ***2*** *à* ***Φ(n)-1***

si ***(pgcd(e, Φ(n)) = 1)*** *alors Retourner****(e)***

sinon ***e ← e +1***

finpour

Fin

Algo.3 Algorithme de détermination de e.

❖ **La clé secrète :**

- **d** est tel que **e.d=1 modulo Φ(n).** Autrement dit, **e.d-1** est un multiple de **Φ(n).** On peut fabriquer d à partir de e, p et q, en utilisant l'algorithme d'inversion modulaire basé sur l'algorithme d'Euclide étendu.
- Algorithme servant à déterminer d :

```
Inversion_modulaire(entier e, entier Φ(n)) : tableau d'entiers
Début
  entier r ← e modulo Φ(n)
  entier  s ← Φ(n) / e
  Bezout = tableau de 2 entiers
  si r = 0
       { Bezout [0] ← 0
Bezout [1] ← 1
retourne Bezout }
       sinon
       { tableau_transi ← inversion_modulaire(e,r)
Bezout [0] ← tableau_transi [1]
         Bezout [1] ← tableau_transi [0] - Bezout[0] * s }
       retourne Bezout; // d est égal à Bezout [0]
Fin
```

Algo.4 Algorithme de l'inversion modulaire.

IV.1.2. Utilisation des clés : [25]

Le cryptage :

- Bob veut transmettre un message secret à Alice. Il commence par le transformer en un nombre entier M, inférieur à n (ou en plusieurs, si nécessaire, car sinon RSA ne fonctionne pas), en codant par exemple chaque lettre du texte par son rang dans l'alphabet (a=01, b=02, etc…).
- Bob calcule ensuite C = M^e modulo n grâce à la méthode d'exponentiation modulaire (voir l'algorithme suivant) puis envoie C à Alice.
- **Algorithme d'exponentiation modulaire rapide :**

```
Exponentiation_modulaire(entier a, entier e, entier n) : entier
Début
entier p ←1
Tant que e >=1 faire
    si e modulo 2 ≠ 0
        p ←(p*a) modulo n
    fin si
        a ←(a*a) modulo n
        e ←e / 2
fin faire
retourner p // résultat de a^e modulo n
Fin
```

Algo.5 Algorithme d'exponentiation modulaire.

Le décryptage :

- Alice décode le message en calculant $\mathbf{C^d[n] = (M^e)^d [n] = M}$ ([n] signifie « modulo n »)
- Démonstration :

  On a **e.d = 1 modulo Φ(n)** avec **Φ(n)= (p-1)(q-1)**

  Donc il existe un entier k tel que **e.d = 1 + k(p-1)(q-1).**

  D'où, si $\mathbf{M \neq 0\ [p]}$: $\mathbf{M^{ed} \equiv M(M^{p-1})^{k\,(q-1)}\ [p]}$

  D'après le petit théorème de Fermat (**d'Euler**) **[1]** (que l'on retrouvera dans la suite) :

Si p est premier, $\mathbf{M^{p-1}-1}$ est divisible par p i.e. $\mathbf{M^{p-1} \equiv 1\ [p]}$ quelque soit M entier, premier à p (c'est le cas puisque $\mathbf{M \neq 0\ [p]}$)

D'où $M^{ed} = M(1)^{k\,(q-1)}$ [p] et donc $\mathbf{M^{ed} \equiv M\ [p]}$

Et enfin, si $\mathbf{M \equiv 0\ [p]}$, on a bien $\mathbf{M^{ed} \equiv M\ [p].}$

De même, avec **q** : $\mathbf{M^{ed} \equiv M\ [q].}$

- **Théorème de gausse :** Soit *a*, *b* et *c* trois entiers relatifs non nuls.

Si *a* divise le produit *bc* et si *a* et *b* sont premiers entre eux alors *a* divise *c*.

Donc d'après le **théorème de Gauss**, puisque p et q sont premiers entre eux, on a

$M^{ed} \equiv M\ [p.q]$, d'où finalement : $\mathbf{M^{ed} \equiv M\ [n]}$

Exemple:

Considérons le doublon (p, q) = (11, 19).

p et q sont bien premiers entre eux et leur produit vaut n = 319.

Φ(n) = (p-1)(q-1) = 10*18 = 280.

Choisissons e premier avec Φ(n), par exemple e = 3 puis calculons d :

Inversion modulaire (e, Φ(n)) retourne le tableau (-93,1) donc d ≡ -93 [280]

Ce qui revient à dire que d ≡ 280 -93 [280] ≡ 187 [280].

Cryptons le message suivant : « CNAM » transformé en nombre « 03140113 » en utilisant la position des lettres dans l'alphabet. Puisque 3140113 > n, nous devons le découper en tronçons inférieurs à n et de taille égale :

« 03 14 01 13 » puis calculer chaque $M_i^e = C_i$ pour i allant de 1 à 4 grâce à l'exponentiation modulaire ce qui donne « 27 192 01 283 ».
On pourrait vérifier que $C_i^d = M_i$, pour chaque i.

IV.2.Détails implémentation RSA :

IV.2.1. rechercher des nombres premiers : [24]

En RSA on recherche p et q premiers et aussi la clé publique e qui est relativement

première avec (p - 1) et (q - 1), c.à.d. pgcd(e, (p - 1)(q - 1)) = 1. Si p et q ne sont pas premiers on risque de pouvoir factoriser n facilement et de trouver la clé privée. Mais le pire est que l'algorithme RSA ne fonctionne plus. Les calculs pour la production de la clé privée doivent se faire modulo Φ(n). Si p et q ne sont pas premiers, alors la fonction indicatrice d'Euler $\Phi(n) \neq (p - 1)(q - 1)$.

Il y a deux philosophies pour rechercher les nombres premiers : les tests de primalité (probabiliste) et les algorithmes de primalité (déterministe). Le critère d'évaluation est celui de leur complexité.

Le crible d'Eratosthène est l'algorithme le plus simple qui cherche les premiers p tel que $p < \sqrt{(N)}$.

1. Algorithme d'Eratosthène : [24]

```
/* Le crible d'Ératosthène */
int Crible( int n, tableau prime[]: prime[1], ... , prime[n]){
pour tout 1<=k<=n
prime[k]= vrai /* au debut tous les nombres sont premiers*/
tant que (i<n et prime[n]==vrai){
si (prime[i]==vrai) alors /* on vient de trouver un nouveau premier */
pour tout 1<j<N { prime[j*i]=faux;}
i=i+1;
}
si (prime[n]== faux) alors i-1 est un facteur de n;
retourne i-1;
}
```

Algo.6 Algorithme d'Eratosthène.

Pour un module RSA n à 256 bits n = p.q est plus grand que 10^{75}. Donc au crible il faut au moins 0,36 x 10^{36} divisions pour décider. Trop !

En pratique nous utilisons des tests de primalité.

Le premier test était basé sur le petit théorème de Fermat (d'Euler).

2. Algorithme d'Euler : [24]

```
Soit n un nombre à tester
Tirer au hasard a tel que 1 < a < n
```

si ***pgcd(a, n) ≠ 1*** *(c.à.d.* ***a facteur de n****) alors* ***n pas premier***

sinon, si $a^{n-1} \neq 1$ ***(mod n), n pas premier***

sinon ***n est peut-être premier***

Algo.7 Algorithme d'Euler.

En essayant pour beaucoup d'entiers test a si $a^p = a \bmod p$ on peut déduire que ces entiers a ne sont pas des facteurs de p et que donc peut-être p est un premier.

Il existe des p non premiers (nombres composés) tels que pour tout a avec pgcd(a, p) = 1 et $a^{p-1} = 1 \bmod p$. Ils s'appellent pseudo-premiers pour le Test de Fermat et la base a.

Exemple : Vérifiez que 341 = 11 . 31 est un pseudo-premier pour le Test de Fermat et la base 2. Vérifiez que la base 3 est témoin que 341 est un nombre composé.

On vérifie en utilisant l'exponentiation rapide si $2^{340} = 1 \pmod{341}$.

$340 = 256 + 64 + 16 + 4 = 2^8 + 2^6 + 2^4 + 2^2$

On sait que $z^{b+c} = z^b \cdot z^c$. Alors

$$2^{340} = 2^{2\wedge 8} \cdot 2^{2\wedge 6} \cdot 2^{2\wedge 4} \cdot 2^{2\wedge 2}$$
$$2^{340} = (((((((2^2)^2)^2)^2)^2)^2)^2)^2 \cdot (((((2^2)^2)^2)^2)^2) \cdot 2^{2\wedge 4} \cdot 2^{2\wedge 2}$$
$$2^{340} = (((((256)^2)^2)^2)^2)^2 \cdot (((256)^2)^2)^2 \cdot 256^2 \cdot 16$$
$$2^{340} = (((((256)^2)^2)^2)^2)^2 \cdot (((256)^2)^2)^2 \cdot 65536 \cdot 16$$

Mais 65536 > 341 et on peut réduire modulo 341 ; $65536 = 192 \cdot 341 + 64 \equiv 64 \pmod{341}$

$$2^{340} \equiv ((((64)^2)^2)^2)^2 \cdot ((64)^2)^2 \cdot 64 \cdot 16 \pmod{341}$$
$$2^{340} \equiv (((4096)^2)^2)^2 \cdot (4096)^2 \cdot 1024 \pmod{341}$$

Mais $4096 = 12 \cdot 341 + 4 \equiv 4 \pmod{341}$ et $1024 = 3 \cdot 341 + 1 \equiv 1 \pmod{341}$

$$2^{340} \equiv ((4^2)^2)^2 \cdot (4)^2 \cdot 1 \pmod{341}$$
$$2^{340} \equiv (256)^2 \cdot 16 \pmod{341}$$
$$2^{340} \equiv 64 \cdot 16 \equiv 1 \pmod{341}$$

Donc 341 est un pseudo-premier pour le test Fermat et la base 2. Similairement on peut calculer $3^{340} \equiv 56 \neq 1 \pmod{341}$. La base 3 est témoin pour le test Fermat du fait que 341 est un nombre composé.

Un nombre n pseudo-premier pour toutes les valeurs de a tels que pgcd(a, n) = 1 est un pseudo-premier absolu pour le Test de Fermat. Les nombres de Carmichael sont des tels pseudo-premiers absolus.

Ils ont été la source de nombreux attaques sur les implémentations RSA utilisant le teste de Fermat/Euler. Maintenant, il y des testes de primalité amélioré comme Miller-Rabin et Solovay-Strassen.

3. Algorithme Miller-Rabin : [24]

Soit ***n*** *un nombre à tester*
Tirer au hasard ***a*** *tel que* ***1 < a < n***
si ***pgcd(a, n) ≠1*** *(****a*** *facteur de* ***n****) alors* ***n*** *pas premier*
sinon on écrit $\boldsymbol{n = 2^k.d}$ *, avec* ***d*** *impaire*
sinon, si $\boldsymbol{a^d \neq 1 \pmod n}$ *et pour tout r appartient à* $\boldsymbol{0,1,...., k-1, a^{(2\wedge r).d} \neq -1 \pmod n}$
alors ***n*** *pas premier*
sinon ***n*** *est peut être premier*

Algo.8 Algorithme de Miller-Rabin.

Exemple : Vérifiez que pour n = 561 = 3.11.17 la base a = 2 est un témoin pour le test Miller-Rabin du fait que 561 est un nombre composé.

Pour le test Miller-Rabin on écrit $n-1 = 2^k.d$ On remarque que

$560 = 16.35 = 2^4.35$, k = 4, d = 35

Il faut vérifier que $a^d \neq 1 \pmod n$ et pour tout r appartient à 0,1,...., k-1, $a^{(2\wedge r).d} \neq -1 \pmod n$.

On remarque que 35 = 32 + 2 + 1, donc on peut écrire

$$2^{35} = 2^{2\wedge 5} \cdot 2^2 \cdot 2$$
$$2^{35} = (((2)^{2\wedge 3})^2)^2 \cdot 2^2 \cdot 2$$
$$2^{35} = ((256)^2)^2 \cdot 2^2 \cdot 2$$
$$2^{35} = (65536)^2 \cdot 2^2 \cdot 2$$

Mais, 65536 > 561 et on peut réduire modulo 561, donc $65536 = 116 \cdot 561 + 460 \equiv 460 \pmod{561}$. Nous avons aussi $460^2 = 211600 = 377 \cdot 561 + 103$ Maintenant on fait le test

$$2^{35} \equiv 103 \cdot 8 \equiv 824 \equiv 263 \neq 1 \pmod{561}$$
$$2^{2.35} \equiv 263^2 \equiv 166 \neq -1 \pmod{561}$$

$2^{4.35} \equiv 263^4 \equiv (263^2)^2 \equiv 166 .166 \equiv 67 \neq -1 \pmod{561}$

$2^{4.35} \equiv 263^8 \equiv (263^4)^2 \equiv 67 . 67 \equiv 1 \neq -1 \pmod{561}$

Donc, 561 n'as pas passée le test Miller-Rabin.

Exemple. Le même problème pour 41041 = 7 . 11 . 13 . 41.

4. L'algorithme déterministe d'Agrawal, Kayal et Saxena (AKS) : [25]

Cet algorithme date d'août 2002 et est le premier à s'exécuter dans un temps polynomial. Jusqu'alors, le meilleur algorithme déterministe était la version de Cohen-Lenstra du test publié par Adleman, Pomerance et Rumely en 1983 (« Sommes de Jacobi »). Sa complexité était en $\log_2(n)^{O(\log_2(\log_2(\log_2(n))))}$, ce qui est très légèrement supérieur à un temps polynomial.

L'idée repose sur une généralisation du petit théorème de Fermat :

Soit $a \in Z$, $n \in N$ avec $n \geq 2$ et pgcd(a,n) = 1 alors n est premier si et seulement si :

$(X-a)^n = X^n - a$ modulo n.

Evaluer cette expression pour un n donné et différents prendrait un temps en O(n). L'idée qu'ont eue ces trois chercheurs pour rendre ce test effectif a été d'évaluer cette identité modulo un polynôme $\mathbf{X^r-1}$, c'est-à-dire tester si

$(X-a)^n = X^n - a$ modulo (X^r-1, n).....(1)

La mention « modulo X^r-1,n » signifie « à un multiple X^r-1 près et à un multiple de n près ». Ainsi **(1)** est vrai s'il existe un polynôme **q(X)** à coefficients entiers tel que, après développement et simplification, tous les coefficients du polynôme $\mathbf{(X-a)^n-(X^n-a)-q(X)(X^r-1)}$ sont des multiples de n. Par exemple :

$(X-1)^3=X^3-1$ modulo (X-1, 3). En effet, en prenant q(X)=-9X, on obtient :

$(X-1)^3-(X^3-1)-(-9X)(X-1) = X^3-3X^2+3X-1-X^3+9X^2-9X = 6X^2-6X$ qui est un polynôme dont les coefficients sont des multiples de 3.

Voici cet algorithme (cf leur article « Primes is in P »):

Soit n > 1, l'entier dont on doit tester la primarité :

1. Si n est de la forme a^b avec b>1, alors répondre « n est composé » et s'arrêter.

2. Trouver le plus petit r tel $o_r(n) > 4 \log_2^2(n)$, où $o_r(n)$ représente l'ordre de n [r], ie le plus petit nombre k tel que $n^k = 1[r]$. Ce qui peut se faire de la façon suivante :

***Détemination_de_r** (entier **n**)*

Début

```
entier ordre = 1
entier r = 1
Tant que ordre <= 4*log2²(n) faire
    r ← r+1
    tant que pgcd (r,n) ≠ 1 faire r ← r+1
    ordre = 1;
    tant que exponentiation_modulaire (n,ordre,r) ≠ 1 faire ordre ← ordre+1
fin tant que
retourne r
Fin
```

3. Si 1< pgcd(a,n) < n, pour a ≤ r alors retourner « n est composé »

4. Si n ≤ r alors retourner « premier »

5. Pour tout nombre a entier entre 1 et $2\log_2(n)\sqrt{r}$ faire

si $(X-a)^n \neq X^n$ – a modulo (X^r-1, n) alors retourner « n est composé ».

Fin pour

6. Retourner « n est premier ».

Algo.9 Algorithme déterministe d'Agrawal, Kayal et Saxena (AKS).

IV.2.2. Exemples RSA :

- **Exemple (codage RSA simple) :**

Soit les nombres premiers p = 11 et q = 5.

1. n = p.q = 11 . 5 = 55. $\Phi = (p-1)(q-1) = 10 . 4 = 40$

2. Choisissons e = 3.

3. Vérifions pgcd(e, p-1) = pgcd(3, 10) = 1. Vérifions pgcd(e; q-1) = pgcd(3,4) = 1.

Par conséquent, pgcd(e, Φ) = pgcd(e, (p-1)(q-1)) = pgcd(3,40) = 1

4. Calculons d tel que $e.d \equiv 1 \pmod{\Phi}$, c.à.d. $d \equiv e^{-1} \pmod{\Phi} \equiv 3^{-1} \pmod{40}$. Trouver une valeur pour d telle que Φ divise (e.d -1), c.à.d. trouver d tel que 40 divise 3d-1.

Une suite de tests simples (d = 1, d = 2, . . .) donne d = 27. Vérification : e.d - 1 = 3 . 27 - 1 = 80, qui est divisible par Φ.

5. La clé publique est (n, e) = (55, 3)

6. La clé privée est (n, d) = (55 , 27)

Soit le message à chiffrer x = 3. Alors $y \equiv x^e \pmod n \equiv 3^3 \pmod{55} \equiv 27 \pmod{55} \equiv 27$.

Donc le message chiffrer est y = 27.

Pour vérifier le déchiffrement on calcule $y' \equiv y^d \pmod n \equiv 27^{27} \pmod{55}$.

Nous n'avons pas besoin de calculer 27^{27} en entier ici. Nous avons déjà vu une manière de calculer y' par exponentiation rapide en utilisant :

$$a \equiv bc \bmod n \equiv (b \bmod n) \,.\, (c \bmod n) \bmod n$$

Donc :

$$y' \equiv 27^{27} \bmod 55 \equiv 27^{9+9+9} \bmod 55$$
$$\equiv (27^9 \bmod 55) \,.\, (27^9 \bmod 55) \,.\, (27^9 \bmod 55)$$

$$27^9 \bmod 55 \equiv 27^{(2+2+2+2+1)} \bmod 55 \equiv 27^2.27^2.27^2.27^2.27 \bmod 55$$
$$\equiv (27^2 \bmod 55)^4 \,.\, (27 \bmod 55) \bmod 55$$
$$\equiv (729 \bmod 55)^4 \,.\, (27 \bmod 55) \bmod 55$$
$$\equiv 14^4 \,.\, 27 \bmod 55 \equiv 38416 \,.\, 27 \bmod 55$$
$$\equiv 26 \,.\, 27 \bmod 55$$
$$\equiv 702 \bmod 55$$
$$\equiv 42 \bmod 55.$$

$$y' \equiv 27^{27} \bmod 55 \equiv (42 \bmod 55) \,.\, (42 \bmod 55) \,.\, (42 \bmod 55)$$
$$\equiv (42^3 \bmod 55) \equiv 74088 \bmod 55 \equiv 3$$

- **Codage d'un message :**

Voici un exemple de l'utilisation de RSA, avec des petits nombres :

je veux envoyer a mon ami un message crypter avec l'algorithme RSA

Mon ami a choisit deux nombre **p** et **q** tel que **p = 193** et **q = 181**.

Il est déduit **n = 34933** , et **Φ(n) = 192 . 180 = 34560**

Il a choisit ensuite **e = 23803**, qui est premier avec **34560**. L'inverse de **23803** modulo 34560 est **d = 7987**.

Il a publier ses clés publiques, par exemple sur son site internet : e = 23803 **et n = 34933** mois, je dois utiliser ces clés pour crypter un message, mais avant de convertir le texte en une suite des nombres d'un fichier informatique, le mieux est d'utiliser le code ASCII. Ce code attribue un nombre entre 0 et 255 pour chaque caractère de base utilisable sur un ordinateur.

le message est «**rdv ce soir**» devient :

Le code ASCII :

r	d	v		c	e		s	o	i	r
114	100	118	32	99	101	32	115	111	105	114

Il suffit de coder chaque nombre comme expliqué ci-dessus. On obtient :

114^{23803} [34933] = 16838 ; 100^{23803} [34933] = 29145 ; etc…

114	100	118	32	99	101	32	115	111	105	114
16838	29145	2800	14419	27579	5572	14419	33408	11570	26142	16838

J'envoie cette suite de nombres à mon ami, qui va le décrypter avec sa clé d. Il va pouvoir retrouver le message original :

16838^{7987} [34933] = 114 ; 291458^{7987} [34933] = 100 ; etc…

16838	29145	2800	14419	27579	5572	14419	33408	11570	26142	16838
114	100	118	32	99	101	32	115	111	105	114
r	d	v		c	e		s	o	i	r

❖ **Autre exemple :**

Voici un exemple de l'utilisation de RSA, avec des petits nombres :

Saddam souhaiterait envoyer le message suivant à George :

« Kisses from Iraq ». Malheureusement, Vladimir les espionnes, et pourrait intercepter ce message. Nos deux compères vont donc crypter leurs échanges avec la méthode RSA.

George a choisi **p = 37** et **q = 43**. Il en déduit n = 37 . 43 = 1591, et $\Phi(n) = 36 . 42 = 1512$.

Il choisit ensuite **e = 19**, qui est premier avec 1512. L'inverse de 19 modulo 1512 est **d = 955**.

George peut donc maintenant publier ses clés publiques, par exemple sur son site internet : e = 19 et n = 1591

Saddam va utiliser ces clés pour crypter son message, mais il doit avant tout convertir son texte en une suite de nombres. Comme Saddam veut envoyer le message sous forme d'un fichier informatique, le mieux est d'utiliser le code ASCII[3].Ce code attribue un nombre entre 0 et 255 pour chaque caractère de base utilisable sur un ordinateur.

En ASCII, « Kisses from Iraq » devient :

k	**i**	**s**	**s**	**e**	**s**		**f**	**r**	**o**	**m**		**i**	**r**	**a**	**q**
107	105	115	115	101	115	32	102	114	111	109	32	105	114	97	113

Il suffit à Saddam de coder chaque nombre comme expliqué ci-dessus. Il obtient :

107^{19} [1591] = 477 ; 105^{19} [1591] = 1338 ; etc…

107	**105**	**115**	**115**	**101**	**115**	**32**	**102**	**114**	**111**	**109**	**32**	**105**	**114**	**97**	**113**
477	1338	1410	1410	1174	1410	930	1397	632	703	483	930	1405	632	532	1441

Saddam envoie cette suite de nombres à George, qui va le décrypter avec sa clé **d**. Il va pouvoir retrouver le message original :

477^{955} [1591] = 107 ; 1338^{955} [1591] = 105 ; etc…

477	**1338**	**1410**	**1410**	**1174**	**1410**	**930**	**1397**	**632**	**703**	**483**	**930**	**1405**	**632**	**532**	**1441**
107	105	115	115	101	115	32	102	114	111	109	32	43	114	97	113
k	**i**	**s**	**s**	**e**	**s**		**f**	**r**	**o**	**m**		**i**	**r**	**a**	**q**

[3] ASCII =Amercian Standard Code for Information Interchange

En recodant en ASCII, George va pouvoir lire le message de son ami, sans que Vladimir n'ait pu le déchiffrer.

IV.2.3. Sécurité de RSA :

RSA est sûr dans les conditions suivantes:

- Les valeurs de **e** et **d** sont très grandes.
- Les valeurs de **p** et **q** restent inconnues.
- Des valeurs sur 1024 bits à 2048 bits semblent être suffisantes pour rendre l'algorithme incassable.
- Des recherches ont montré qu'il est possible de casser un code RSA avec une clé de 1024 bits en huit mois en faisant coopérer 1600 ordinateurs conventionnels.

IV.2.4 Avantages et inconvénients de cryptage asymétrique (RSA) :

Avantages :

- Une seule clé secrète à enregistrer
- Très utile pour échanger les clés pour ouvrir un tunnel de communication chiffré.

Inconvénients :

- Lenteur.
- Pas d'authentification de la source.
- Attaque Man-In-The-Middle.

IV.3 Principe de fonctionnement du RSA :

IV.3.1 Coder le message et la signature : [25]

Si Bob souhaite qu'Alice soit sûre qu'il s'agit bien de lui, il peut coder sa signature en utilisant sa clé secrète S_B : S_B(signature). Il code ensuite le message et S_B(signature) avec la clé publique d'Alice P_A. Alice décode le tout avec sa clé secrète S_A, obtient le message = $S_A(P_A(\text{message}))$ et S_B(signature) = $S_A(P_A(S_B(\text{signature}))$. Elle vérifie la signature de Bob en appliquant $P_B(S_B(\text{signature}))$=signature, P_B étant la clé publique de Bob.

IV.3.2 Combiner RSA avec un algorithme à clé privée : [25]

Les algorithmes à clé publique sont assez lents (à cause des nombreuses exponentiations modulaires à calculer). La méthode généralement utilisée pour envoyer un message volumineux, est de tirer au hasard une clé secrète, chiffrer le message avec un algorithme à clé privée en utilisant cette clé, puis chiffrer cette clé aléatoire elle-même avec la clé publique du destinataire. Ceci permet d'avoir la sécurité des systèmes à clé publique, avec la performance des systèmes à clé privée. Il existe des

logiciels qui effectuent toutes ces opérations de manière transparente, et qui, de plus, sont gratuits et téléchargeables à partir de dizaines de sites de par le monde comme le **PGP** que nous décrivons ci-après.

- **Un exemple : le PGP**

Le PGP (Pretty Good Privacy) est un algorithme de chiffrement à destination des particuliers. Il est surtout utilisé pour chiffrer des messages envoyés par courrier électronique, même s'il peut aussi être utilisé pour chiffrer tous les fichiers. PGP a été mis au point en 1991 par l'informaticien américain Philip Zimmermann, et ceci lui valut divers problèmes avec la justice. D'une part, le PGP utilise l'algorithme RSA, qui était à l'époque breveté aux Etats-Unis (le brevet a expiré en septembre 2000). D'autre part, la NSA (national security agency) a tout fait pour tenter d'empêcher la diffusion du PGP. En effet, la puissance de ce programme met à la disposition de chacun un moyen de cacher ses échanges électroniques qui résiste même aux assauts de la plus puissante des agences de renseignements du monde (c'est d'ailleurs pour cette raison que son utilisation fut formellement interdite en France jusqu'en 1999).

PGP utilise le meilleur de la cryptographie symétrique (1000 fois plus rapide que le cryptage à clé publique, selon N.Dausque du CNRS « Certificats (électroniques) : Pourquoi? Comment ? ») et de la cryptographie asymétrique (sécurité de l'échange de clés). Il fonctionne comme suit :

Le logiciel commence par compresser le texte ce qui permet de réduire le temps de transmission des données, et d'améliorer également la sécurité. En effet, la compression détruit les modèles du texte (fréquences des lettres, mots répétés) or ces modèles sont souvent utilisés dans les analyses cryptographiques.

Ensuite PGP génère une clé de session (nombre aléatoire) qui ne sera utilisé qu'une seule fois et qui va servir à chiffrer le message.

La clé de session est chiffrée en utilisant la clé publique du destinataire avec l'algorithme RSA.

Enfin, le message ainsi que la clé de session cryptés sont envoyés au destinataire. Celui-ci récupère d'abord la clé de session, en utilisant sa clé privée, puis il décrypte le message grâce à la clé de session.

IV.3.3 Le problème du choix des nombres: [26]

- **L'exposant e :**

MM Boneth et Venkatesan ont établi en 1998 que casser le RSA, lorsqu'il est utilisé avec des exposants e trop petits, est moins difficile que de factoriser n (cf exemple ci-dessous).

MM K.Lestra et R.Veerheu ont recommandé en 1999 que l'exposant public soit choisi égal à $2^{16}+1 = 65537$ (nombre de Fermat) plutôt que les valeurs premières 3 ou 17 longtemps utilisées mais considérées aujourd'hui comme facteur de vulnérabilité.

Afin d'améliorer l'efficacité du cryptage, il est tentant de choisir un petit exposant, par exemple e=3. Supposons que 3 personnes aient choisi le même e mais 3 n différents. Si une autre personne souhaite leur envoyer le même message, elle devra crypter m comme ceci :

$c_i = m^3 \bmod n_i$, for i = 1, 2, 3. Si les n_i sont premiers deux à deux, une oreille indiscrète pourrait intercepter c_1, c_2, c_3 et utiliser l'algorithme de Gauss pour trouver une solution x, $0 \leq x < n_1 n_2 n_3$, au système suivant :

$x = c_1$ modulo n_1 ; $x = c_2$ modulo n_2 ; $x = c_3$ modulo n_3

Si $m^3 < n_1 n_2 n_3$, le théorème des restes Chinois nous dit que $x = m^3$. Il ne reste plus qu'à calculer la racine cubique de m^3 et le tour est joué. Evidemment ça fait beaucoup de « si » (cette dernière phrase ne figurait pas dans le livre). Je l'ai essayé avec m=69, n_1=15, n_2=22 et n_3=1081. On a bien $69^3 = 328509 < n_1 n_2 n_3 = 356730$ et $x = 69^3$ est bien solution du système :

x=9 mod 15 ; x=5 mod 22 ; x=966 mod 1081

Remarque : Cela nous montre également qu'il vaut mieux éviter de chiffrer le même message avec plusieurs clés différentes…

- **Les facteurs p et q :**

JP.Delahaye conseille de prendre p et q de taille proche et d'éviter que les nombres p+1, p-1, q+1, q-1 soient trop faciles à factoriser car alors n risquerait de l'être aussi par l'utilisation d'algorithmes spécialisés de factorisation sachant en tirer parti.

- **Le nombre n :**

Il ne doit pas être choisi par plus d'une entité car sinon, l'utilisation des diverses clés publiques des utilisateurs du même n risquerait de donner accès aux facteurs de n.

Il ne doit pas être trop petit (au minimum 1024 bits). On se souvient qu'en 1998, l'affaire Humpich a fait la une des journaux (cet informaticien a montré qu'il était possible de fabriquer de toute pièce une fausse carte qui permettait de payer chez un commerçant). Serge Humpich avait contourné deux systèmes de sécurité :

D'une part, il a fabriqué des "yes card", c'est-à-dire des cartes à puce qui, quelque soit le code secret entré, renvoie "code bon".

D'autre part, il a contourné l'authentification hors-ligne RSA : en 1998, le n utilisé par le GIE (groupement interbancaire) avait pour taille 320 bits (taille inchangée depuis 1990). A cette époque, factoriser un tel entier n'était plus impossible (le record se situait à 432 bits), et Humpich, en utilisant simplement un logiciel japonais de factorisation, a réussi à factoriser le n du GIE, et à découvrir la clé secrète. Depuis, le n a changé, et est désormais long de 768 bits.

IV.3.4. Les garanties du RSA : [25]

- **Est « facile » de fabriquer des clés ?**

Nous avons vu que pour fabriquer les clés, il fallait choisir deux grands nombres premiers. Après un bref aperçu de la problématique, nous verrons deux tests permettant de savoir si un grand nombre est premier : l'un stochastique et l'autre déterministe.

- **Quelques observations à propos des nombres premiers :**

Nombre de mathématiciens se sont depuis fort longtemps penchés sur le problème de la primarité des nombres. Le crible d'Ératosthène (240 av JC) était une méthode déterministe qui consistait à diviser un nombre n par tous les nombres entiers inférieurs à $\sqrt{n}$. Mais ce test est inefficace pour de grands nombres puisque sa complexité est en $O(\sqrt{n})$ (les spécialistes considèrent que seules les tâches traitées en temps polynomial sont faciles). C'est Fermat (1640) qui va être à l'origine de tests efficaces…

- **Mais difficile, voire impossible de les « casser » :**

« L'inviolabilité » de la méthode RSA vient de la difficulté de factoriser n pour en déduire p, q puis d. Cette idée d'inviolabilité vient de l'expérience, ce n'est ni un fait démontré, ni même pour certains spécialistes un fait démontrable. L'amélioration des méthodes de factorisation et /ou le progrès technique pourraient remettre sérieusement en cause les petites clés de RSA, comme nous le verrons après avoir décrit le principe de l'algorithme GNFS (crible général de corps de nombres) qui a conduit au record de la plus grande clé cassée.

- **Quelques records :**

Depuis des années la société RSA organise des challenges pour casser des très grandes clés. Le 22 août 1999 une équipe de l'institut national de recherche en mathématiques et sciences informatiques d'Amsterdam a réussi le challenge RSA-155 ("cassage" d'une clé de 512 bits) en factorisant un nombre de 155 chiffres en 2 nombres de 78 chiffres.

L'expérience a été réalisée avec un réseau de 300 ordinateurs pour un cumul de 8 000 Mips[4] année (équivalent à environ 40 PC Pentium III à 500 Mhz pendant 1 an), l'opération a duré 3 mois et demi et a donc utilisé le GNFS. (source : JP.Delahaye, Pour la Science)

Une première phase a consisté à chercher un bon jeu de paramètres pour la méthode : les spécialistes évaluent à 100 Mips année la quantité de calculs faite pour elle, qui a duré 9 semaines en utilisant une seule machine. La seconde phase, qui a été distribuée sur un réseau de 300 machines réparties dans le monde entier, est aussi la plus coûteuse en calcul. Elle a fourni les relations qui sont au cœur de la méthode.

Un total de 124 millions de relations a ainsi été engendré, dont 39 millions apparaissaient en double à cause de la méthode de calcul distribué. Le tri des relations et la préparation de la matrice du système linéaire à résoudre ont demandé un mois de travail. La matrice résultante comportait 6 699 191 lignes et 6 711 336 colonnes. La résolution du système d'équations a été faite par un ordinateur vectoriel possédant 2 giga-octets de mémoire centrale et a demandé 224 heures. Enfin, une légère amélioration de la méthode utilisée par le précédent record (140 chiffres) et réalisé 6 mois auparavant a permis d'économiser 6 000 Mips (un gain de 40%).

Tout cela montre que si casser une clé RSA de 512 bits est faisable, cela demande des moyens et des compétences qui ne sont pas à la portée de n'importe qui.

- **RSA assurerait quand même une sécurité à 99,8% : [27]**

Si la proportion est réduite, cela compromet néanmoins la fiabilité du commerce sur internet, d'autant plus que des millions d'achats se font en ligne chaque année. Cet algorithme est utilisé pour protéger les services de banques en ligne, de commerce en ligne, d'e-mail et de transactions en ligne. En pratique, si un utilisateur se connecte à un site de e-commerce, les transactions sont cryptées par la clé publique du site. Et elles ne peuvent être déchiffrées par le propriétaire du site que si ce dernier possède la clé privée correspondante. Les clés publiques sont émises par les autorités de certification, intégrées dans le certificat numérique. Et, en théorie, il est impossible de deviner le code d'une clé privée et, de plus, aucune paire de clés identiques ne jamais fabriquées.

[4] MIPS = Million d'instructions par seconde

D'après le site Network World, la division de la sécurité de RSA aurait répliqué en affirmant que les résultats de l'étude n'indiquaient pas un défaut fondamental dans l'algorithme. En effet, leur propre analyse leur a révélé que "les données ne pointent pas vers une faille dans l'algorithme, mais souligne plutôt l'importance de sa bonne mise en œuvre". L'équipe a tout de même précisé que "RSA assure au mieux, la sécurité à 99,8%".

Cependant, le chercheur Nadia Heninger[5] tient à rassurer les acheteurs en ligne et les banquiers. Selon elle, pas besoin de paniquer : "le problème touche principalement les systèmes embarqués comme les routeurs ou les dispositifs VPN[6], mais pas les serveurs web".

Conclusion :

Dans ce chapitre nous avons étudié l'algorithme RSA d'une manière très détaillés, a savoir ses différents méthodes pour le cryptage et le décryptage des donnés par rapport a ses propre clef.

[5] Le chercheur Nadia Heninger, cryptographe de l'université de Pennsylvanie avec l'aide de l'INRIA en France et de quelques autres labos

[6] VPN = Virtual Private Network

CHAPITRE V

CONCEPTION ET REALISATION

Introduction :

Nous allons détailler notre algorithmes RSA tout d'abord, et on va commencer par la description de l'environnement de développement et d'implémentation de notre application, puis nous nous focaliserons sur la présentation de cette dernière, tout en illustrant les différentes interfaces qu'elle contient.

V.1Conception de la base de données :

Nous intéressons maintenant aux données nécessaires pour le fonctionnement de l'application. Pour les obtenir, nous suivons de près le déroulement de l'utilisation sur les données nécessaires, qui seront stockées dans les tables de la base de données.

V.1.1 Les tables:

Elle contient les tables suivantes : emp, message, crypter.

Table emp (Employé) :

Nom du champ	Type de donnée	Description	Clé
Id_emp	Int(11)	L'identifiant de l'employé	Primaire
nom_emp	Varchar(25)	Nom du l'employé	
prenom_emp	Varchar(25)	Prénom de l'employé	
nai_emp	Varchar(15)	Date de naissance de l'employé	
adr_emp	Varchar(50)	Adresse de l'employé	
clepublic	Varchar(2048)	Clé public de l'employé	
cleprive	Varchar(2048)	Clé privé de l'employé	
randomN	Varchar(4100)	La valeur de N	
login	Varchar(25)	Login	
pass_emp	Varchar(25)	Mot de passe	
taille	Int(11)	Taille de la clé	

Table message (les messages claire) :

Nom du champ	Type de donnée	Description	Clé
id_mess	Int(11)	L'identifiant de message	Primaire

msg	Varchar(25)	Le message qu'on va crypter	
id_emp	Int(11)	L'identifiant de l'employé	Etranger

Table crypter (les message crypter) :

Nom du champ	Type de donnée	Description	Clé
Id_cry	Int(11)	L'identifiant de message crypté	Primaire
Message_cry	Mediumtext	Le message crypté	
id_emp	Int(11)	L'identifiant de l'employé	Etranger

V.2 Description de notre algorithme RSA :

Voici le code de RSA que nous avons implémenté dans notre application :

```
public class RSA
{
        //La longueur en bits de chaque nombre premier.
        int primeSize ;
        //Deux distincte grands nombres premiers p et q.
        BigInteger p, q ;
        //Modulus N.
        BigInteger N ;
```

```
        //r = ( p - 1 ) * ( q - 1 )
        BigInteger r ;
        // Exposant public E et un exposant privé D
        BigInteger E, D ;

        String nt,dt,et;

        String publicKey;
        String privateKey;
        String randomNumber;

        BigInteger[] ciphertext;
        int m[]  = new int[1000];
        String st[]  = new String[10000];
        String str = "";
        String sarray1[] = new String[100000];

        StringBuffer sb1 = new StringBuffer();

public RSA( int primeSize )
        {
                this.primeSize = primeSize ;
                // Générer deux distincte grands nombres premiers p et q.
                generatePrimeNumbers() ;
                // Générer les clés publics et privés.
                generatePublicPrivateKeys() ;
                BigInteger publicKeyB = getE();
                BigInteger privateKeyB = getD();
                BigInteger randomNumberB = getN();
                publicKey = publicKeyB.toString();
                privateKey = privateKeyB.toString();
                randomNumber = randomNumberB.toString();

}

        // Générer deux distincte grands nombres premiers p et q.
        public void generatePrimeNumbers()
        {
                p = new BigInteger( primeSize, 10, new Random() ) ;
                do
                {
                        // Construire un nombre probablement premier q avec la probabilité que  q est
premier dépasse  1-1/2^10=99.9%
                        q = new BigInteger( primeSize, 10, new Random() ) ;
                }
                while( q.compareTo( p ) == 0 ) ;
        }
// Générer les clés publics et privés.
        public void generatePublicPrivateKeys()
        {
                // N = p * q
                N = p.multiply( q ) ;
                // r = ( p - 1 ) * ( q - 1 )
                r = p.subtract( BigInteger.valueOf( 1 ) ) ;
                r = r.multiply( q.subtract( BigInteger.valueOf( 1 ) ) ) ;  //(p-1)(q-1)
```

```
                // Choisissez A, premier et moins avec r
                do
                {
                        E = new BigInteger( 2 * primeSize, new Random() ) ;
                }
          while( ( E.compareTo( r ) != -1 ) || ( E.gcd( r ).compareTo( BigInteger.valueOf( 1 ) ) != 0
) ) ;
                // calcule D, l'inverse de E mod r
                D = E.modInverse( r ) ;  //d ≡ e^-1 mod r
        }
}

//  Encryption


        public String RSAencrypt(String info) {

                E = new BigInteger(publicKey);
                N = new BigInteger(randomNumber);
                try {
                ciphertext = encrypt( info ) ;
                for( int i = 0 ; i < ciphertext.length ; i++ )
                {
                        m[i] = ciphertext[i];
                        st[i] = String.valueOf(m[i]);
                        sb1.append(st[i]);
                        sb1.append(" ");     //séparation de message a encrypter avec des  espaces
                        str = sb1.toString();

                }
                }
                catch (Exception e) {
                        System.out.println(e);
                }

                return str;
        }
        public BigInteger[] encrypt( String message )
        {
                        int i ;
                        byte[] temp = new byte[1] ;
                        byte[] digits = new byte[8];
                        try {
                        digits = message.getBytes() ;
                        String ds = new String(digits); // on va récupérer le message a encrypter

                        System.out.println("ds="+ds);

                        }
                        catch (Exception e) {
                                System.out.println(e);
                        }
                        BigInteger[] bigdigits = new BigInteger[digits.length] ;
                        for( i = 0 ; i < bigdigits.length ; i++ )
                        {
```

```
                    temp[0] = digits[i] ; //le code ascii de iéme élément
                    bigdigits[i] = new BigInteger( temp ) ;
                }
                BigInteger[] encrypted = new BigInteger[bigdigits.length] ;
                for( i = 0 ; i < bigdigits.length ; i++ )
                    encrypted[i] = bigdigits[i].modPow( E, N ) ;//on va crypter le iéme
élement avec E et N

                return( encrypted ) ;
        }


        // Decrption

        public String RSAdecrypt() {
                D = new BigInteger(decrypter.dd.getText());
                N = new BigInteger(decrypter.nnn.getText());

                System.out.println("D = " + D);
                System.out.println("N = " + N);
          String encryptedData1 = decrypter.textcrypter2015.getText();//textcrypter2015 : est la
zone de texte pour écrire un message
            encryptedData = encryptedData1.toString();
            System.out.println("arzeee:" +encryptedData);
                int k1= 0;
                //Tokenizer permet de facilité le traitement des chaines de caractères c-a-d découper la
chaine en morceaux appelé les tokens  pou ce fait ca , il considère qu'une chaine est constituée d'une
suite de tokenssepares par des caracteres speciaux ici espace
                StringTokenizer st = new StringTokenizer(encryptedData);
                while (st.hasMoreTokens()) {
                    sarray1[k1] = st.nextToken(" ");
                    k1++;
                }
                 // placer les blocs de messages crypter dans un tableaux ciphertext1
                BigInteger[] ciphertext1 = new BigInteger[100000];

                for( int i = 0 ; i <k1 ; i++ ) {
                    ciphertext1[i] = new BigInteger(sarray1[i]);
                }
                String recoveredPlaintext = decrypt( ciphertext1,D,N,k1) ;
                System.out.println("arzeee:" +recoveredPlaintext);
            decrypter.textclaire2015.setText(recoveredPlaintext);
                return recoveredPlaintext;
        }
        public String decrypt( BigInteger[] encrypted,BigInteger D,BigInteger N,int size )
        {       D = new BigInteger(decrypter.dd.getText());
                N = new BigInteger(decrypter.nnn.getText());
                int i ;
                String rs="";
                BigInteger[] decrypted = new BigInteger[size] ;
                for( i = 0 ; i < decrypted.length ; i++ ) {
                    decrypted[i] = encrypted[i].modPow( D, N ) ;
                }
                char[] charArray = new char[decrypted.length] ;
                byte[] byteArray = new byte[decrypted.length] ;
```

```
            for( i = 0 ; i < charArray.length ; i++ ) {
                    charArray[i] = (char) ( decrypted[i].intValue() ) ;
                    Integer iv = new Integer(0);
                    iv=decrypted[i].intValue() ;
                    byteArray[i] = iv.byteValue();
            }
            try {
                    rs=new String( byteArray );
            }
            catch (Exception e) {
                    System.out.println(e);
            }
            return(rs) ;

    }
```

V.3 Environnement de développement :

V.3.1 Présentation de l'environnement :

V.3.1.1. Langage de programmation utilisé :

Langage JAVA :

JAVA est un langage de programmation orienté objets développé par Sun Microsystems. Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Un objet est une représentation simplifiée d'une entité du monde réel : entité concrète (ex : ma voiture) ou non (ex : la date d'aujourd'hui). Un objet se caractérise par son état et son comportement. Un objet stocke son état dans des variables appelées champs (ou attributs) et présente son comportement a travers de fonctionnalités appelées méthodes. Il est caractérisé par les points suivants :

- **Java est indépendante de toute plate-forme :** Une application en java fonctionne sur n'importe quel environnement (Unix, Windows, …) disposant d'une JVM (Java Virtual Machine).
- **Java est extensible à l'infini :** Idéalement, toutes les catégories d'objets (appelées classes) existantes en java sont définies par extension d'autres classes, en partant de la classe de base la plus générale : la classe Object. Pour étendre le langage il suffit donc de développer de nouvelles classes.
- **Java est un langage de haute sécurité :** Java a été développée dans un souci de sécurité maximale. L'idée maitresse est qu'un programme comportant des erreurs ne doit pas pouvoir être compilé. Ainsi les erreurs ne risquent pas d'échapper du

programmeur et de passer les procédures de tests. En détectant les erreurs à la source, on évite qu'elles se propagent en s'amplifiant.

- **Java est un langage compilé :** C'est-à-dire qu'avant d'être exécuté, il doit être traduit dans le langage de la machine sur laquelle il doit fonctionner. Cependant, contrairement à de nombreux compilateurs, java traduit le code source dans le langage de sa JVM. Le code traduit appelé byte code, ne peut pas être exécuté directement par le processeur d'une machine.
- **Java est doté de standard de bibliothèques de classes :** ces classes sont très riches et elles comprennent la gestion des interfaces graphiques (fenêtres, boites de dialogue, contrôles, menus, graphisme), la programmation multi-threads (multitâches), la gestion des exceptions, les accès aux fichiers et au réseau… l'utilisation de ces bibliothèques facilite grandement la tâche du programmeur lors de la construction d'applications complexes.

  On a utilisé ces bibliothèques :

  import java.util.*; contient le cadre des collections, classes de collection de l'héritage, modèle d'événement, date et heure des installations, l'internationalisation, les classes d'utilitaires divers (un tokenizer de chaîne, un générateur de nombres aléatoires (Random)).

  import java.io.* ; Système entrée et sortie par flux de données, la sérialisation et le système de fichiers.

  import java.math.* ; précision arbitraire entier (BigInteger) et décimal (BigDecimal) arithmétique.

  import java.sql.*; pour les données d'accès et de traitement mémorisées dans une source de données (par exemple, une base de données relationnelle).

V.3.1.2. Présentation de NetBeans et WampServer :

- **NetBeans** :

Est à l'origine un environnement de développement intégré (EDI) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and

Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS.

NetBeans permet de programmer et concevoir les interfaces utilisateur de manière visuelle. Pour ce faire, il offre de nombreux outils de conception visuelle qui permettent de concevoir les interfaces utilisateur avec rapidité et efficacité en attachant des événements et en modifiant les dispositions.

Pour notre réalisation nous avons utilisés la version 8.0.2 (NetBeans 8.0.2).

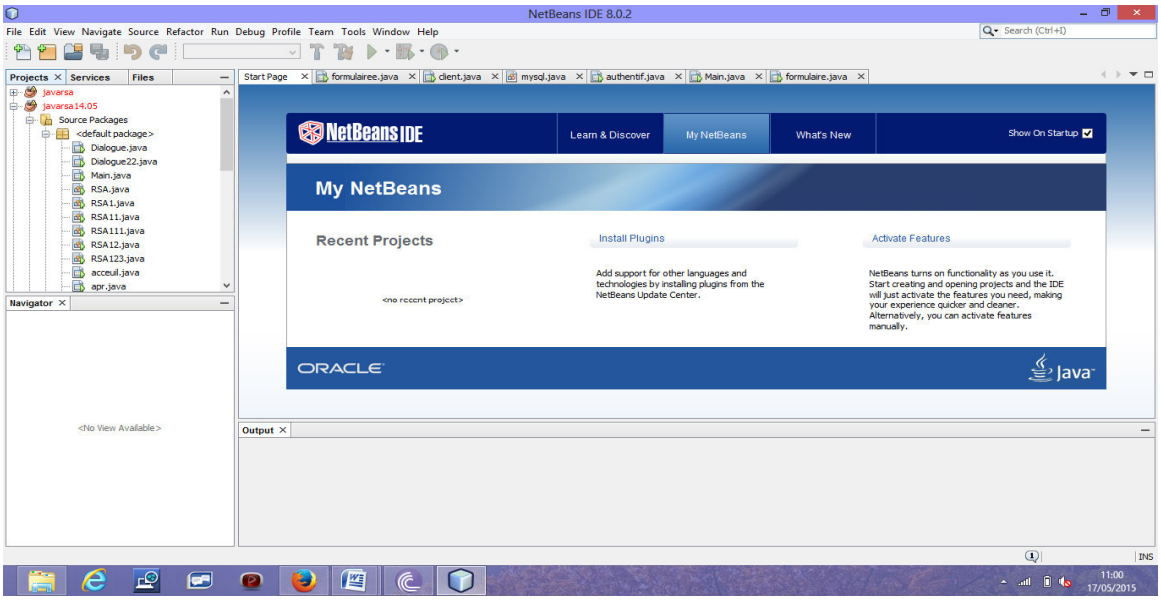

Figure V.2: Interface d'environnement de développement NetBeans.

Wamp Server :

WampServer 2 (anciennement WAMP5) est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans se connecter à un serveur externe) des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi qu'une administration pour les deux bases SQL PhpMyAdmin et SQLiteManager.
Il dispose d'une interface d'administration permettant de gérer et d'administrer ses serveurs au travers d'un tray icon (icône près de l'horloge de Windows).

La grande nouveauté deWampServer 2 réside dans la possibilité d'y installer et d'utiliser n'importe quelle version de PHP, Apache ou MySQL en un clic. Ainsi, chaque développeur peut reproduire fidèlement son serveur de production sur sa machine locale.

V.4 Présentation de l'application :

V.4.1 La page menu :

Cette interface est conçue de sorte qu'elle offre une simplicité d'utilisation. Et permet l'accès aux différentes opérations du logiciel (s'inscrire, entrer, a propos, quitter).

- Le bouton «s'inscrire» : permet de l'inscription pour faire des opération de cryptage.
- Le bouton «Entrer» : pour de rentrer a l'application.
- Le bouton «À propos» : permet le lancement de la fenêtre à propos.
- Le bouton «Quitter» : pour quitter l'application.

Figure V.3 : Interface de la page menu.

V.4.2 La page inscription:

D'abord on va choir la taille de la clé qu'on a besoin, et de remplir tout les champs de formulaire et après on va cliquer sur le bouton générer des clés pour avoir notre propre clé privé et clé public, et pour terminer l'inscription on tape sur valider.

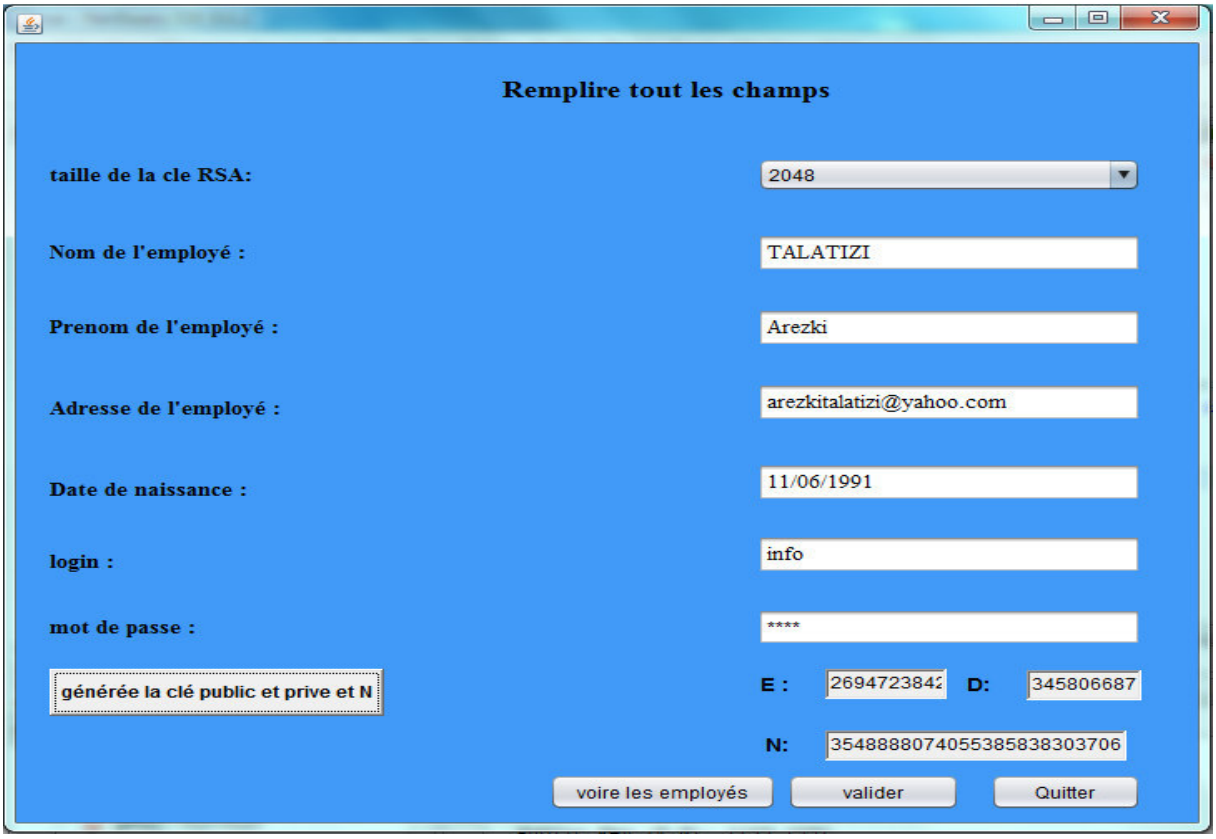

Figure V.4 : Interface de formulaire d'inscription.

V.4.3 La page authentification:

Pour accéder a l'interface principale il faut s'authentifier.

Figure V.5 : Interface de l'authentification.

V.4.4La page d'accueil:

Dans la page d'accueil en peut accéder a tout les fonctionnalités de l'application crypter, décrypter, modifier votre profil, voir tout les messages claire.

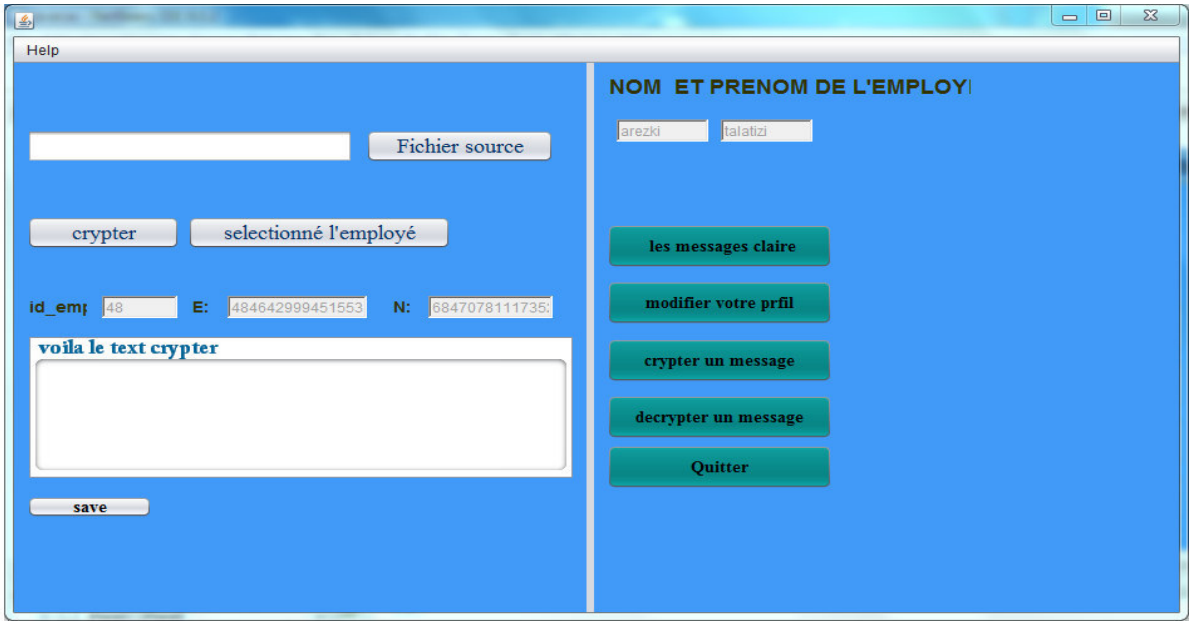

Figure V.6 : Interface de la page d'accueil.

V.4.5 La page cryptage:

Pour crypter un message il faut choisir a quelle employée vous crypter le, et après ecrire votre message et enfin cliquer sur crypter.

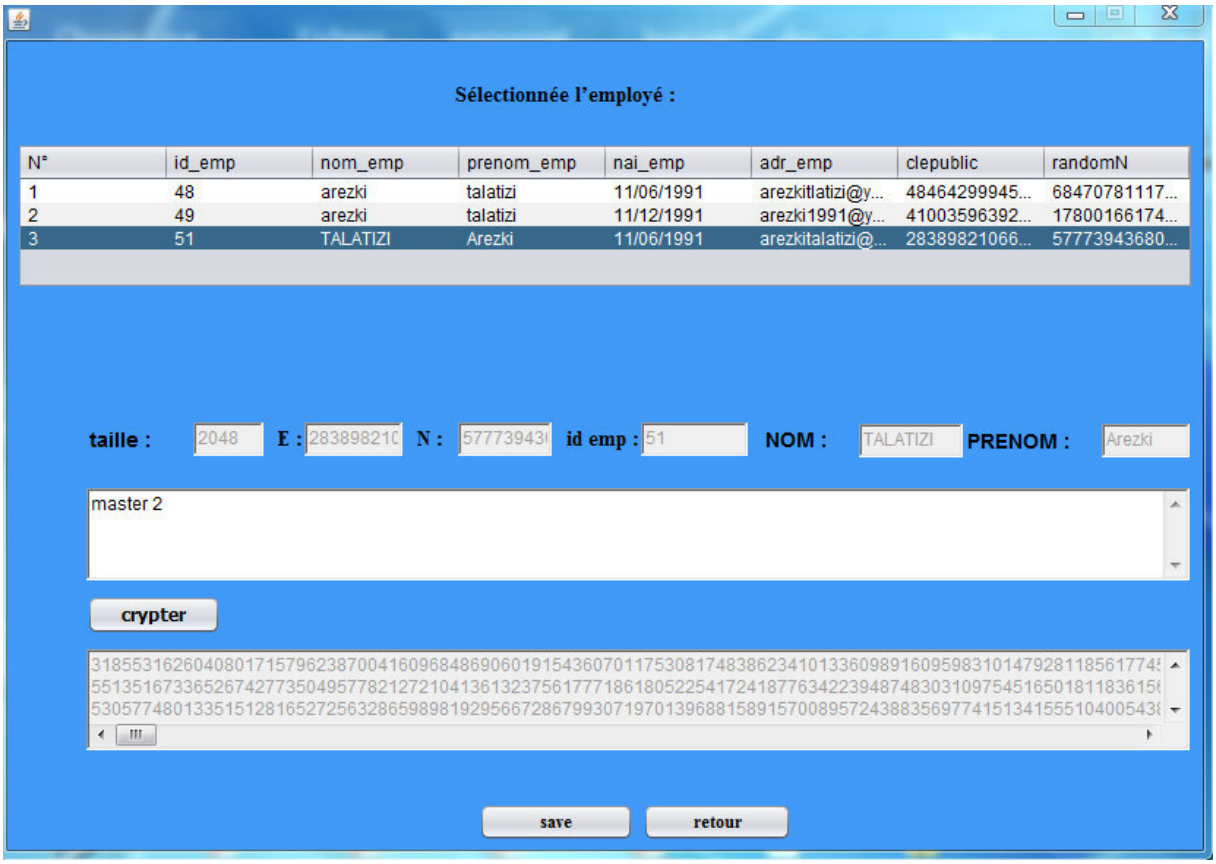

Figure V.7 : Interface pour crypter un message.

V.4.6 La page décryptage:

Pour décrypter un message il faut d'abord choisir le message qu'est reçu a travers de son profile et puis cliquer sur décrypter pour voir le message en claire et même vous pouvez sauvegarder le message.

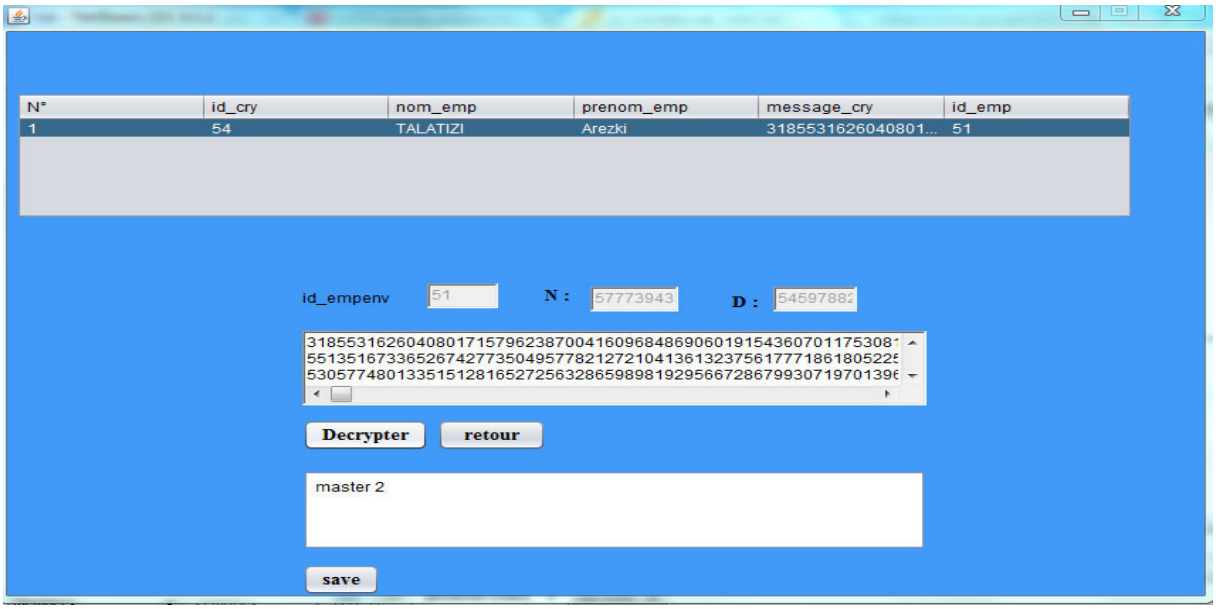

Figure V.8 : Interface pour décrypter un message.

V.4.7 La page de crypter un fichier txt:

Pour vous crypter les données d'un fichier il doive choisir d'abord l'employée qu'on va crypter le texte et après choisir l'emplacement de texte et enfin cliquer sur crypter.

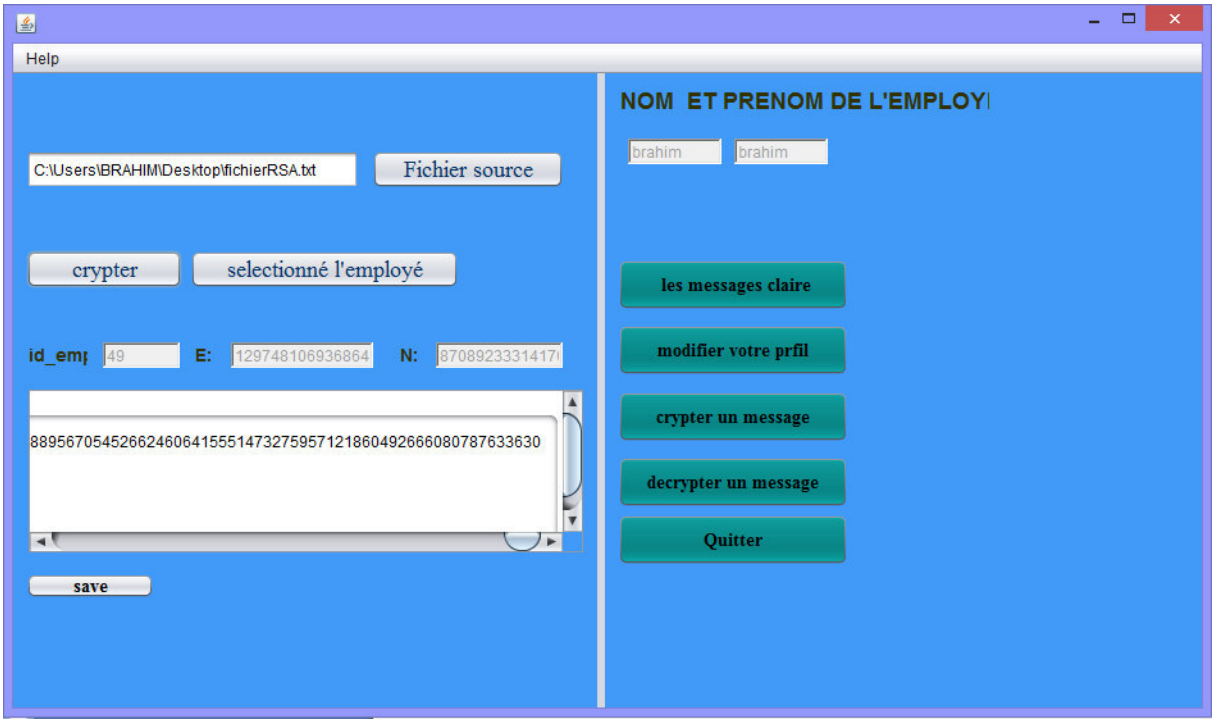

Figure V.9 : Interface pour crypter un message de forme fichier txt.

V.4.8 La page d'aide:

Cliquer sur le titre pour visualiser son contenu.

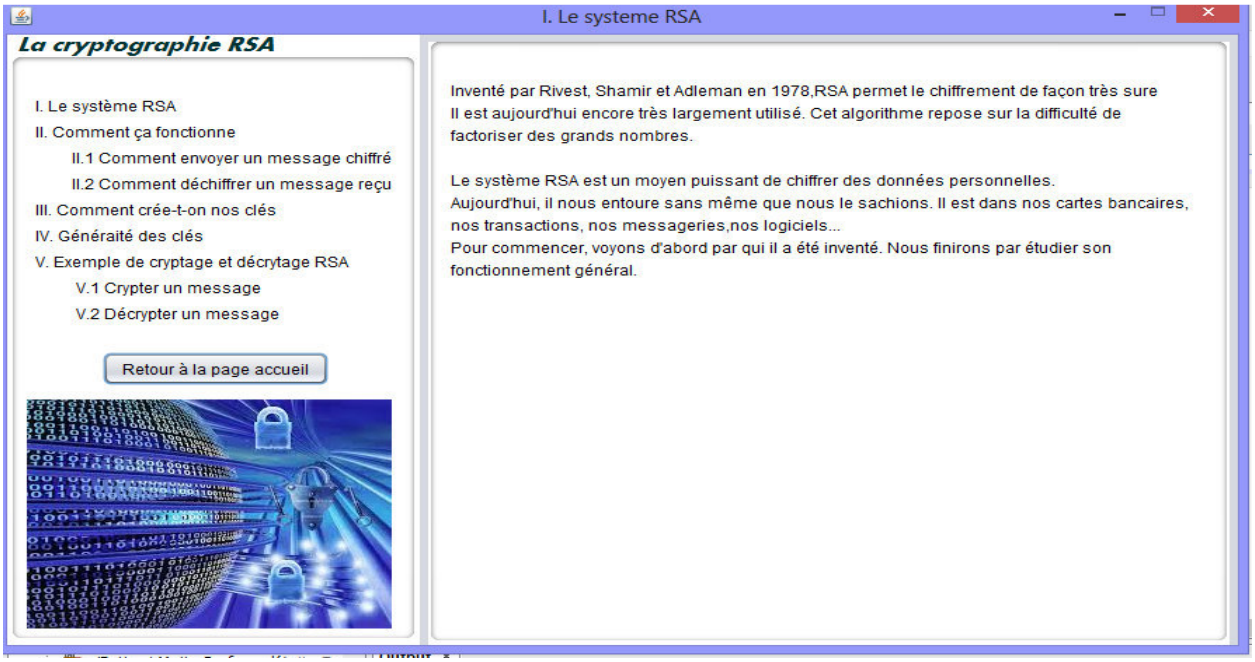

Figure V.10 : Interface pour help sur le système RSA.

Conclusion :

Dans ce chapitre, nous avons présenté l'environnement d'implémentation et de développement de notre application, ainsi que les outils utilisés pour réaliser notre travail, et les différentes interfaces que nous avons créé dans notre application.

Ce programme de cryptage nous a permis d'avoir une nouvelle approche de la conception

Conclusion Générale :

Aujourd'hui, la cryptographie tente de concevoir des systèmes plus sûrs et plus efficaces comme par exemple des systèmes garantissant intégrité et confidentialité en passant une seule fois sur le message clair au lieu de passer une première fois pour le chiffrer et de repasser pour garantir l'intégrité du chiffré. Enfin, tous les modèles d'adversaires n'ont pas encore été définis. Aujourd'hui, les attaques du monde réel sont réalisées soit sur des protocoles qui n'ont pas été prouvés sûrs, soit parce que le modèle d'adversaire ne reproduit pas le monde réel.

Ce travail avait pour le but de créer une application qui crypte et sécurise des informations transmises a travers les clients, et pour cela nous avons implémenté l'algorithme RSA pour plus de sécurité.

Notre projet a été partagé par plusieurs étapes et on peut distinguer deus étapes essentiels la première partie c'est la théorique qui consiste à comprendre mieux le fonctionnement de la sécurité informatique, et la base de la cryptographie ; et la deuxième partie c'est la réalisation ou on a étudié et développé notre application en utilisant les outils et le langage nécessaire à savoir l'outil de développement NETBEANS.

Enfin, la réalisation de ce travail nous a permis d'acquérir des connaissances très importantes et très utiles dans le domaine de développement des applications en JAVA.

BIBLIOGRAPHIE

Références bibliographiques :

[1] : http://www.fsr.ac.ma/cours/informatique/rziza/cours/Introduction_Reseau.pdf

[2] : http://www.courstechinfo.be/Reseaux/Classif.html

[3] : http://www.fil.univ-lille1.fr/~sedoglav/RSX/Introduction.pdf

[4] : http://static.ccm2.net/www.commentcamarche.net/contents/pdf/topologie-des-reseaux-512-kjmscd.pdf

[5] : http://www.frameip.com/osi/osi.gif

[6] : http://www.fil.univ-lille1.fr/~sedoglav/RSX/Introduction.pdf

[7] : Guy pujolle.les réseaux Eyrololles, 1995,1997.

[8] : http://www.commentcamarche.net/contents/539-tcp-ip

[9] : http://www.frameip.com/tcpip/tcpip.gif

[10] : J.Hulaas, cours technologie internet, Université de Genève, 2001.

[11] : http://www.commentcamarche.net/contents/221-reseaux-architecture-client-serveur-a-3-niveaux

[12] : http://www.irisa.fr/prive/bcousin/Cours/1-Securite-des-reseaux.2P.pdf

[13] : http://www.ducrot.org/securite.pdf

[14] : http://ylescop.free.fr/mrim/cours/securite.pdf

[15] : https://encrypted tbn3.gstatic.com/images?q=tbn:ANd9GcQVAHrpJKjTZC224maKGjelpR0aCn-y9YLdPlxdenQFac4vuHW9fA

[16] : www.irisa.fr/prive/bcousin/Cours/1-Securite-des-reseaux.2P.pdf

[17] : http://www.awt.be/web/sec/index.aspx?page=sec,fr,fic,045,004

[18] : http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-fr-4/ch-detection.html

[19] : http://www.eyrolles.com/Chapitres/9782212132335/Chap-1_Bloch.pdf

[20] : ylescop.free.fr/mrim/cours/securite.pdf

[21] : www.irisa.fr/prive/bcousin/Cours/1-Securite-des-reseaux.2P.pdf

[22] : ftp://ftp.pgpi.org/pub/pgp/6.5/docs/french/IntroToCrypto.pdf

[23] : http://math.univ-lyon1.fr/~roblot/resources/masterpro_chapitre_1.pdf

[24] : http://pageperso.lif.univ-mrs.fr/~andreea.dragut/enseignementCLAA/CryptoChap3RSA2012.pdf

[25] : jeremy.leclert.free.fr/Projet_Final/RSA.DOC

[26]:http://labit501.upct.es/~fburrull/docencia/SeguridadEnRedes/teoria/bibliography/HandbookOfAppliedCryptography_AMenezes.pdf

[27] : http://www.maxisciences.com/rsa/rsa-le-systeme-de-cryptage-le-plus-securise-a-un-defaut_art21831.html