RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

Université Mouloud Mammeri

Faculté de Génie Electrique et de l'Informatique

Département d'Informatique

Tizi-Ouzou

Mémoire de fin d'études

En vue de l'obtention d'un diplôme de Master en Informatique

Option : « ingénierie des systèmes d'information »

Thème :

Conception et réalisation d'un système de hachage basé sur la fonction SHA-1

Présenté par :

***M^r^:* MAMERI Ahcene**

M^r^: OUBELLIL Belaid

Dirigé par :

M^me^: HADAOUI.R

--- Année universitaire : 2013-2014---

Remerciement

Nous tenons à remercier vivement :

- Allah le tout puissant qui nous a donné la force, le courage, la santé et la volonté pour élaborer ce travail.
- M^{me}, Haddaoui pour tous ses conseils précieux, ses orientations et ses encouragements.
- Les membres de jury, pour l'honneur qu'ils nous ont fait en acceptant de juger notre travail.
- Tous ceux qui nous ont aidés de près ou de loin dans notre travail.

Belaid & Ahcene

DÉDICACES

Je dédie ce modeste travail à :

- ***Mes très chers parents qui m'ont donné la vie et qui ont été toujours la pour moi.***
- ***Mes frères et a Mes deux sœurs et ainsi la famille MAMERI.***
- ***A tous mes amis(es) des prés et de loin.***
- ***Mon ami et mon binôme BELAID avec qui j'ai réalisé ce travail ainsi qu'à toute sa famille.***

Toute la promotion Master 2 ISI

2013/2014.

AHCENE

DÉDICACES

Je dédie ce modeste travail à :

Ames très chers parents.

A mes frères et sœurs.

Tous mes amis de la promotion

Master 2 ISI

Mon binôme hakim

Je remercie tous ceux et celle qui m'ont aidé à réaliser ce travail

BEAD

Sommaire :

Chapitre V : la réalisation

Liste des figures

INTRODUCTION GÉNÉRALE

De tout temps, les services secrets ont utilisé toutes sortes de codages et de moyens cryptographiques pour communiquer entre agents et gouvernements, de telle sorte que les ennemis ne puissent pas comprendre les informations échangées.

La cryptographie a alors évolué dans ces milieux fermés qu'étaient les gouvernements, les services secrets et les armées. Ainsi, très peu de gens, voir personne n'utilisait la cryptographie à des fins personnelles. C'est pourquoi, pendant tant d'années, la cryptographie est restée une science discrète longtemps réservée au domaine diplomatique et militaire.

S'appuyant alors sur des principes mathématiques élémentaires, la cryptographie a commencé à évoluer vers le milieu du 20ème siècle avec le début des télécommunications en intégrant essentiellement des techniques de codage de l'information, mais il a fallu attendre les années 1970 pour qu'elle passe du secret des laboratoires militaires au domaine public, et s'installe comme une véritable science dans les domaines universitaires. Beaucoup d'articles ont alors été publiés et des conférences publiques ont été présentés .

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.

Il existe aussi plusieurs mécanismes de sécurité, dont le but est de garantir un écho des recommandations de la sécurité informatique, mais la nécessité de garantir un niveau de sécurité maximal, et l'importance de la cryptographie dans ce domaine nous ont motivés à choisir cette technique pour qu'elle soit l'objet principal de notre étude.

Dans cette étude, nous avons essayé d'analyser l'environnement de la sécurité informatique. Aussi, nous avons étudié la fonction de hachage SHA-1, et nous avons proposé un logiciel adoptant à cette fonction.

Cinq chapitres sont consacrés à ce travail :

Apres l'introduction générale, on trouvera :

Chapitre I :les réseaux informatique

Chapitre II : la sécurité informatique

Il est divisé en deux parties

Partie 1 : problématique de la sécurité informatique

Partie 2 : mécanismes de sécurisation

Chapitre III :les concepts fondamentaux de la cryptographe

Il est divisé en trios parties

Partie 1 :la cryptographie

Partie 2 :les fonctions de hachage cryptographiques

Partie 3 : la fonction de hachage SHA-1

Chapitre IV : analyse et conception

Chapitre V : la réalisation

I. Les réseaux informatiques

I.1. Introduction

Les technologies liées aux traitements de l'information continue de s'évoluer de jour en jour. En effet ces quinze dernières années ont été marquées par l'apparition des réseaux informatiques à travers leurs généralisations et leurs utilisations sur l'échelle mondiale.

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. Aujourd'hui, l'intégration de la parole téléphonique et de la vidéo est généralisée dans les réseaux informatiques, même si cela ne va pas sans difficulté.

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

I.2. Objectifs d'un réseau [5]

Un réseau informatique permet:

- Le partage des fichiers et des périphériques, imprimante, application
- La communication entre personnes grâce au courrier électronique
- La communication entre processus (ordinateur et automate industriel)
- La garanti de l'unicité de l'information (base de données)
- Les jeux en réseau ou sur internet
- Les réseaux permettent ainsi de standardiser le développement des applications.

I.3. Classification des réseaux

I.3.1. Selon la taille [1]

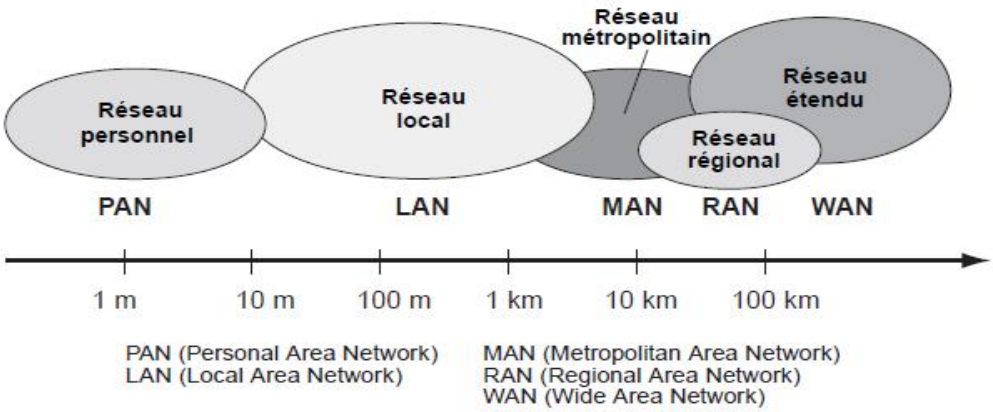

Figure I.1: Classification des réseaux informatiques selon leur taille. [1]

- **PAN**

Les réseaux personnels, ou PAN (Personal Area Network), interconnectent sur quelques mètres des équipements personnels tels que terminaux GSM, portables, organiseurs, etc., d'un même utilisateur.

- **LAN**

Les réseaux locaux, ou LAN (Local Area Network), correspondent par leur taille aux réseaux intra-entreprises. Ils servent au transport de toutes les informations numériques de l'entreprise. En règle générale, les bâtiments à câbler s'étendent sur plusieurs centaines de mètres. Les débits de ces réseaux vont aujourd'hui de quelques mégabits à plusieurs centaines de mégabits par seconde.

- **MAN**

Les réseaux métropolitains, ou MAN (Metropolitan Area Network), permettent l'interconnexion des entreprises ou éventuellement des particuliers sur un réseau spécialisé à haut débit qui est géré à l'échelle d'une métropole. Ils doivent être capables d'interconnecter les réseaux locaux des différentes entreprises pour leur donner la possibilité de dialoguer avec l'extérieur.

- **RAN**

Les réseaux régionaux, ou RAN (Regional Area Network), ont pour objectif de couvrir une large surface géographique. Dans le cas des réseaux sans fil, les RAN peuvent avoir une cinquantaine de kilomètres de rayon, ce qui permet, à partir d'une seule antenne, de connecter un très grand nombre d'utilisateurs.

Cette solution devrait profiter du dividende numérique, c'est-à-dire des bandes de fréquences de la télévision analogique, qui seront libérées après le passage au tout-numérique.

- **WAN**

Les réseaux étendus, ou WAN (Wide Area Network), sont destinés à transporter des données numériques sur des distances à l'échelle d'un pays, voire d'un continent ou de plusieurs continents. Le réseau est soit terrestre, et il utilise en ce cas des infrastructures au niveau du sol, essentiellement de grands réseaux de fibre optique, soit hertzien, comme les réseaux satellite.

I.3.2. Selon la topologie

On distingue quatre topologies qui sont :

Topologie en bus [6]

C'est une architecture très facile à mettre en œuvre. Un câble relie les stations directement les unes aux autres, comme dans un réseau de distribution d'eau, il faut donc une terminaison à l'extrémité du bus. L'information diffusée par un poste est envoyée en même temps vers tous les postes de travail, mais seul le poste destinataire est censé la traiter. La communication sur le réseau devient impossible, une fois le câble coupé. Cette topologie est utilisée dans les réseaux Ethernet.

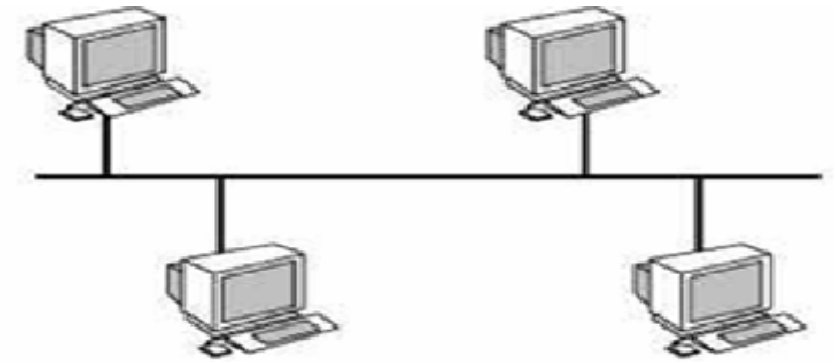

Figure I.2: Topologie en bus

v Topologie en anneau [6]

Dans ce type de réseaux, les équipements sont reliés entre eux pour former une boucle. Un jeton circule en permanence entre les stations. Une station qui veut émettre un message, remet le jeton en position " occupé ", puis transmet son message d'une station à une autre jusqu'à son destinataire, qui reconnaît son adresse dans l'entête, lit le message et remet le jeton à l'état ''libre''. Au bout d'un tour, la station émettrice, voit passer son message avec le jeton libre et sait ainsi que son message a été reçu.

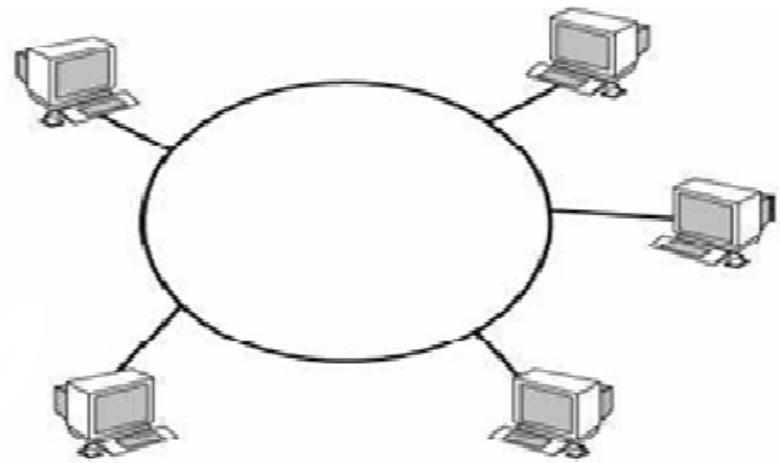

Figure I.3: Topologie en anneau

v Topologie en étoile [5]

Dans cette architecture, les câbles sont tous concentrés en un point central : le « HUB » ou « nœud central » qui est chargé de diffuser l'information à transmettre vers d'autres stations. Son inconvénient réside dans le câblage car il faut plus de câbles que pour les autres topologies, et si le concentrateur tombe en panne, tout le réseau est anéanti. L'avantage de cette topologie est la facilité

de localiser la station affectée, cette dernière ne remet pas en cause l'ensemble du réseau.

Hub : c'est un équipement électronique auquel sont reliés plusieurs ordinateurs.

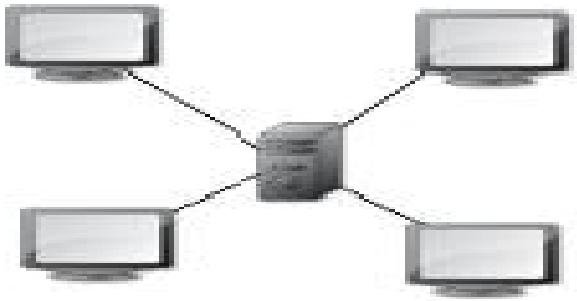

Figure I.4: Topologie en étoile

v Topologie maillée [7]

Dans cette topologie les stations sont toutes reliées les unes aux autres. Elle offre une redondance, et une meilleure fiabilité. Le câblage se fait par des câbles distants. Si un câble tombe en panne un autre câble prend son rôle.

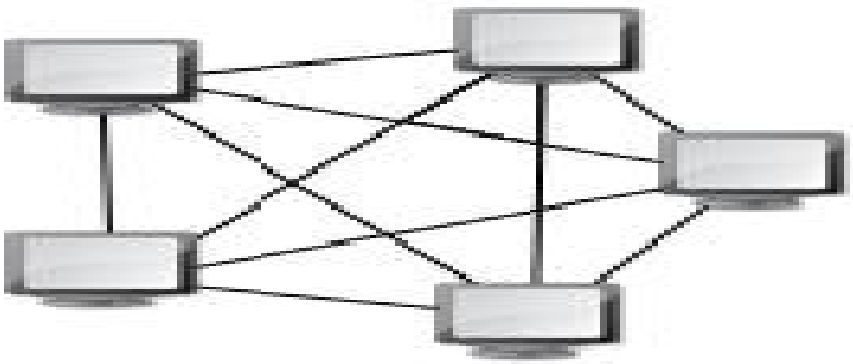

Figure I.5: Topologie en maillée

I.4. 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.4.1. Le modèle OSI (Open System Interconnection) [8]

1.4.1.1. 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.

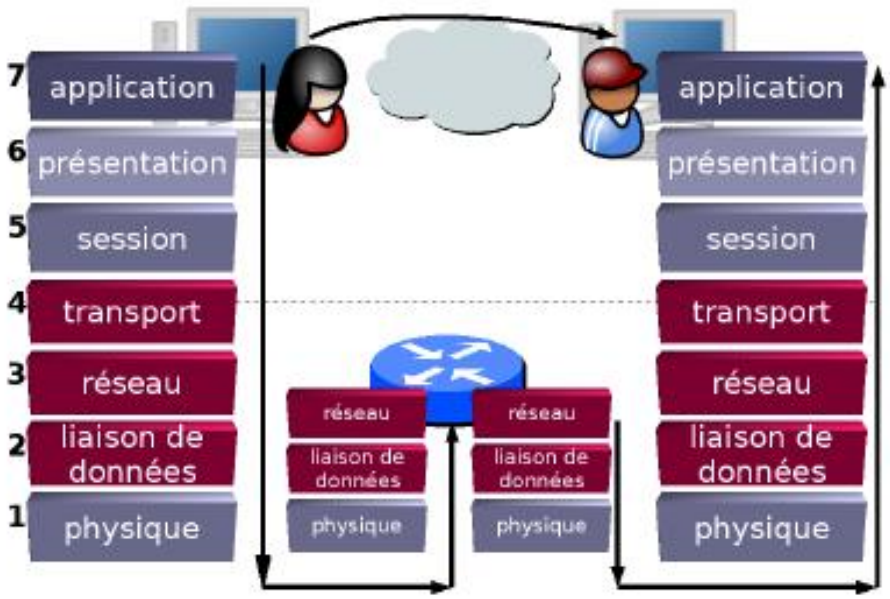

Figure I.6 : Architecture du modèle OSI [10]

I.4.1.2. Les rôles des différentes couches

- **La couche physique :**

S'occupe du flot binaire; transmission des bits. Définit la façon dont les données sont physiquement converties en signaux numériques sur le média de communication (impulsions électriques, modulation de la lumière, etc.).

- **La couche liaison de données :**

S'occupe de la détection/correction d'erreurs règle de partage du support physique. Définit l'interface avec la carte réseau et le partage du média de transmission.

- **La couche réseau :**

Permet de gérer l'adressage et le routage des données, c'est-à-dire leur acheminement via le réseau.

- **La couche transport** :

Est chargée du transport des données, de leur découpage en paquets et de la gestion des éventuelles erreurs de transmission.

- **La couche session** :

Définit l'ouverture et la destruction des sessions de communication entre les machines du réseau.

- **La couche présentation :**

Définit le format des données manipulées par le niveau applicatif (leur représentation, éventuellement leur compression et leur chiffrement) indépendamment du système.

- **La couche application :**

Assure l'interface avec les applications. Il s'agit donc du niveau le plus proche des utilisateurs, géré directement par les logiciels.

I.4.1.3. Avantages du modèle OSI [9]

- 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.4.2. Le modèles TCP/IP [14]

TCP/IP représente d'une certaine façon l'ensemble des règles de communication sur Internet et se base sur la notion 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 crée à l'origine dans un but militaire. Car l'origine de TCP/IP remonte au réseau ARPANET. ARPANET est un réseau de télécommunication conçu par l'ARPA (Advanced Research Projects Agency), l'agence de recherche du ministère américain de la défense (le DoD : Department of Defense).

Outre la possibilité de connecter des réseaux hétérogènes, ce réseau devait résister à une éventuelle guerre nucléaire, contrairement au réseau téléphonique habituellement utilisé pour les télécommunications mais considéré trop vulnérable. Il a alors été convenu qu'ARPANET utiliserait la technologie de commutation par paquet (mode datagram), une technologie émergeante promettant. C'est donc dans cet objectif et ce choix technique que les protocoles TCP et IP furent inventés en 1974.

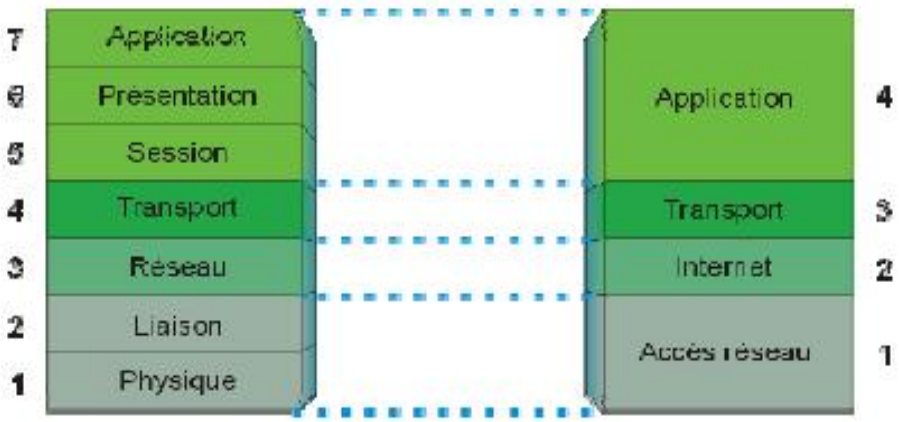

Figure I.7 : Comparaison entre le modèle OSI et le TCP/IP [14]

Le terme de couche est utilisé pour évoquer le fait que les données qui transitent sur le réseau traversent plusieurs **niveaux de protocoles**. Ainsi, les données (paquets d'informations) qui circulent sur le réseau sont traitées successivement par chaque couche, qui vient rajouter un élément d'information (appelé *en-tête*) puis sont transmises à la couche suivante. Ainsi les appellations changent suivant les couches :

- Le paquet de données est appelé **message** au niveau de la couche Application
- Le message est ensuite encapsulé sous forme de **segment** dans la couche Transport
- Le segment une fois encapsulé dans la couche Internet prend le nom de **datagramme**
- Enfin, on parle de **trame** au niveau de la couche Accès réseau

I.4.2.1. les différentes couches du modèle TCP / IP [15]

On peut définir les couches de ce modèle de la façon suivante en partant des couches les plus basses :

- **La couche accès réseaux :**

La couche accès réseau est la première couche de la pile TCP/IP, elle offre les capacités à accéder à un réseau physique quel qu'il soit, c'est-à-dire les moyens à mettre en œuvre afin de transmettre des données via un réseau. Ainsi, la couche accès réseau contient toutes les spécifications concernant la transmission de données sur un réseau physique, qu'il s'agisse de réseau local (Anneau à jeton - token ring, Ethernet, FDDI), de connexion à une ligne téléphonique ou n'importe quel type de liaison à un réseau. Elle prend en charge les notions suivantes :

- **ü** Acheminement des données sur la liaison.
- **ü** Coordination de la transmission de données (synchronisation).
- **ü** Format des données.
- **ü** Conversion des signaux (analogique/numérique).
- **ü** Contrôle des erreurs à l'arrivée.

- **La couche Internet :**

Cette couche est responsable de l'adressage logique du réseau, de l'acheminement de l'information d'un nœud du réseau à un autre. Les unités logiques d'information véhiculées par cette couche sont appelées datagramme.

- **La couche transport :**

Cette couche parfois appelée couche hôte ou service provider layé ou l'on trouve deux protocoles TCP et UDP, est responsable du service de transmission fiable de données.

Le terme segment est utilisé pour désigner les paquets d'informations.

- **La Couche Application :**

La couche application est la couche située au sommet des couches de protocoles TCP/IP. Celle-ci contient les applications réseaux permettant de communiquer grâce aux couches inférieures. Les logiciels de cette couche communiquent donc grâce à un des deux protocoles de la couche inférieure (la couche transport) c'est-à-dire TCP ou UDP. Les applications de cette couche sont de différents types, mais la plupart sont des services réseau, c'est-à-dire des applications fournies à l'utilisateur pour assurer l'interface avec le système d'exploitation. On peut les classer selon les services qu'ils rendent :

- ü Les services de connexion à distance.
- ü Les utilitaires Internet divers.
- ü Les services de connexion au réseau.
- ü Les services de gestion (transfert) de fichier et d'impression

I.4.3. 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.4.3.1. Le protocole TCP / IP ? (Transmission Control Protocol/Internet Protocol) [16]

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.4.3.2. Le protocole IP [16]

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.4.3.3. Le protocole UDP (User Datagram Protocol) [16]

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.4.3.4. Le protocole TCP (Transmission Control Protocol) [16]

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. Architecture client-serveur

I.5.1. Fonctionnement d'un système Client / Serveur

L'architecture client/serveur désigne un mode de communication entre plusieurs ordinateurs d'un réseau qui distingue un ou plusieurs clients du serveur, chaque logiciel client peut envoyer des requêtes à un serveur. Un serveur peut être spécialisé en serveur d'applications, de fichiers, de terminaux, ou encore de messagerie électronique.

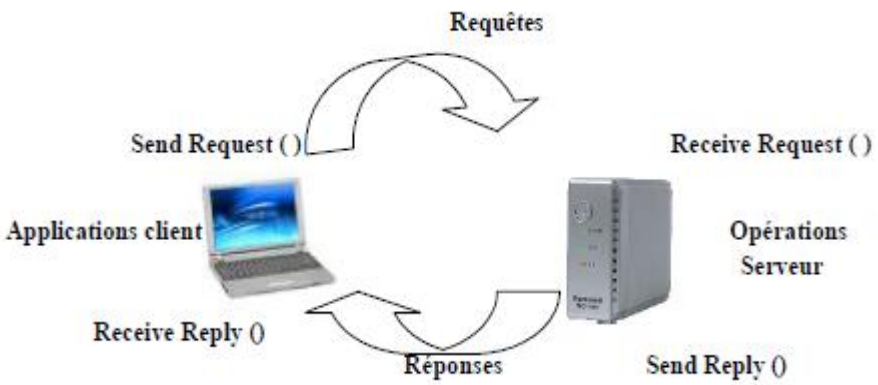

Figure I.8 : Dialogue entre client et serveur [12]

I.5.2. Type d'architecture client / serveur

I.5.2.1. Architecture client/serveur à 2-tiers

Architecture client/serveur à 2 tiers (2 niveaux) est l'architecture la plus classique, elle décrit les systèmes client/serveur dans lesquels, la logique applicative est enfouie soit dans l'interface utilisateur chez le client, soit dans la base de données chez le serveur (ou dans les deux à la fois).Dans cette architecture, le serveur exécute la requête du client et fournit directement le service, sans faire appel à d'autres intermédiaires. L'architecture client/serveur à deux niveaux est schématisée comme suit :

Figure I.9 : Architecture client/serveur 2-tiers. **[12]**

I.5.2.2. Architecture client/serveur à 3-tiers

Cette architecture sépare l'application en trois niveaux de services distincts, conformes au principe précédent :

- **Premier niveaux:** constitué a l'affichage et les traitements locaux (contrôle de saisie, mise en forme de données…) sont pris en charge par le poste client
- **Deuxième niveaux :** constitué des traitements applicatifs globaux sont pris en charge par le serveur d'application
- **Troisième niveau :** contient les services de bases de données qui sont pris par un SGBD.

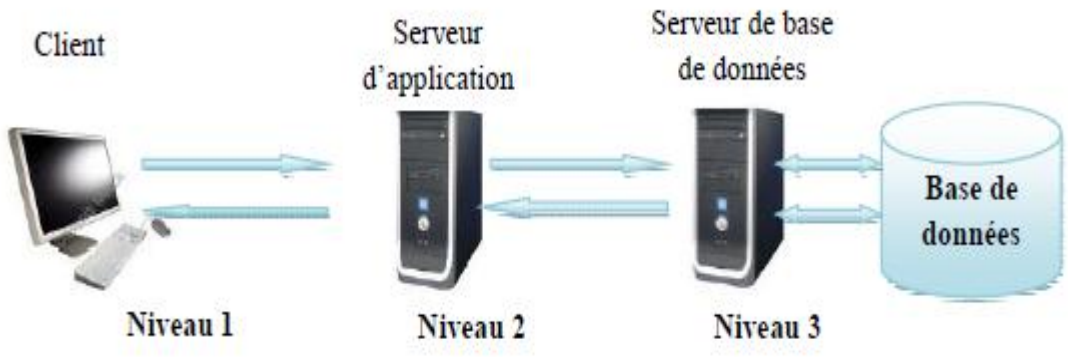

Figure I.10 : Architecture client/serveur à 3-tiers [12]

I.5.2.3. Architecture client/serveur à n-tiers

L'architecture à n-tiers a été pensée pour pallier aux limitations des architectures 3 tiers et concevoir des applications puissantes et simples à maintenir. Ce type d'architecture permet de distribuer plus librement la logique applicative, ce qui facilite la répartition de la charge entre tous les niveaux. Cette architecture est illustrée dans la figure suivante :

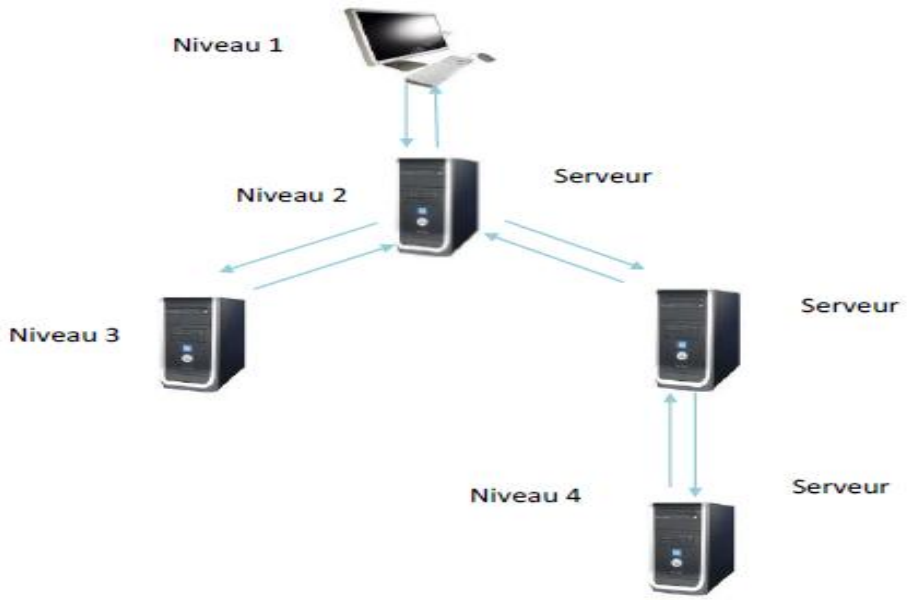

Figure I.11 : Architecture client/serveur à n-tiers [12]

I.5.3. Avantages de client / serveur [13]

Le modèle client/serveur est particulièrement recommandé pour les réseaux nécessitant un haut niveau de fiabilité. Ses principaux atouts sont :

- **Des ressources centralisées** : Etant donné que le serveur est au centre du réseau, il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisées, afin d'éviter les problèmes de redondance et de contradiction.
- **Une meilleur sécurité** : Car le nombre de points d'entrée permettant l'accès aux données est moins important.
- **Une administration au niveau serveur :** Les clients ayant peu d'importance dans ce modèle, ils ont moins besoin d'être administrés.
- **Un réseau évolutif** : Grâce à cette architecture on peut supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modifications majeures.

I.5.4. Inconvénients du modèle Client / Serveur [13]

- Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge (alors que les réseaux pair à pair fonctionnent mieux en ajoutant de nouveaux participants).
- Si le serveur n'est plus disponible, plus aucun des clients ne marche (le réseau pair à pair continue à fonctionner, même si plusieurs participants quittent le réseau).
- Les coûts de mise en place et de maintenance sont élevés.

I.6. Architecture d'égal à égal [11]

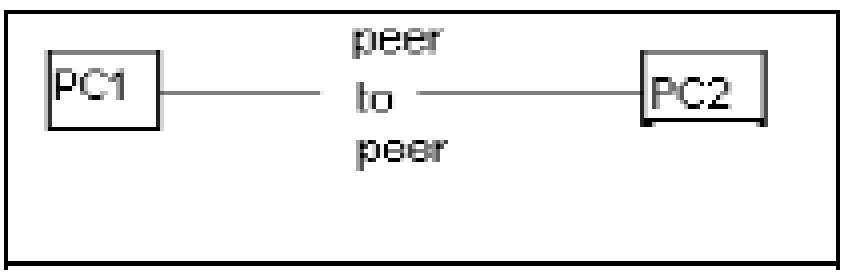

Figure I.12: Architecture peer to peer

I.6.1. Principe

Dans une architecture d'égal à égal (en anglais *Peer to Peer*), contrairement à une architecture de réseau de type client/serveur, il n'y a pas de serveur dédié. Ainsi chaque ordinateur dans un tel réseau est un peu serveur et un peu client. Cela signifie que chacun des ordinateurs du réseau est libre de partager ses ressources. Un ordinateur relié à une imprimante pourra donc éventuellement la partager afin que tous les autres ordinateurs puissent y accéder via le réseau.

I.6.2. Avantages

L'architecture d'égal à égal a tout de même quelques avantages parmi lesquels :

- Il est facile de mettre en réseau des postes qui étaient au départ isolés.
- Chaque utilisateur peut décider de partager l'une de ses ressources avec les autres postes.
- Dans un groupe de travail, l'imprimante peut être utilisée par tous.
- Cette méthode est pratique et peu coûteuse pour créer un réseau domestique.

I.6.3. Inconvénients

Les réseaux d'égal à égal ont énormément d'inconvénients :

- Chaque utilisateur a la responsabilité du fonctionnement du réseau.
- Les outils de sécurité sont très limités.
- Si un poste est éteint ou s'il se "plante", ses ressources ne sont plus accessibles.
- Le système devient ingérable lorsque le nombre de postes augmente.
- Lorsqu'une ressource est utilisée sur une machine, l'utilisateur de cette machine peut voir ses performances diminuer.

I.7. 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.

Partie 1 Problématique de la sécurité informatique

I.1. Introduction

Les problèmes techniques actuels de la sécurité informatique peuvent, au moins provisoirement, être classés en deux grandes catégories :

- ceux qui concernent la sécurité de l'ordinateur proprement dit, serveur ou poste de travail, de son système d'exploitation et des données qu'il abrite ;
- ceux qui découlent directement ou indirectement de l'essor des réseaux, qui multiplie la quantité et la gravité des menaces.

Si les problèmes de la première catégorie citée ici existent depuis la naissance de l'informatique, il est clair que l'essor des réseaux, puis de l'Internet, en a démultiplié l'impact potentiel en permettant leur combinaison avec ceux de la seconde catégorie.

La résorption des vulnérabilités repose sur un certain nombre de principes et de méthodes que nous allons énumérer dans la suite de cette partie.

I.2. Environnement de la sécurité

Plusieurs facteurs engendrent l'insécurité des systèmes informatiques et parmi eux on peut citer :

I.2.1. Vulnérabilité des systèmes

La vulnérabilité est une faille dans les actifs (les équipements, les matériels, les logiciels, les processus, etc.), les contrôles techniques de sécurité ou les procédures d'exploitation ou d'administration utilisés dans un réseau. Elle consiste en une faiblesse dans la protection du système, sous la forme d'une menace qui peut être exploitée pour intervenir sur l'ensemble du système, ou d'un intrus qui s'attaque aux actifs.

I.2.1.1. Vulnérabilités liées aux domaines physiques

- Manque de ressources au niveau équipement.
- Accès aux salles informatiques non sécurisé.
- Absence ou mauvaise stratégie de sauvegarde de données.

I.2.1.2. Vulnérabilités liées aux domaines technologiques

- Failles nombreuses dans les services et applicatifs Web et les bases de données.
- Pas de mises à jour des systèmes d'exploitation et des correctifs.
- Pas de contrôles suffisants sur les logiciels malveillants.
- Récurrence des failles et absence de supervision des événements.
- Réseaux complexes, non protégés.
- Mauvaise utilisation de la messagerie.

I.2.2. Les menaces [2]

Une menace est quelqu'un ou quelque chose qui peut exploiter une vulnérabilité pour obtenir, modifier ou empêcher l'accès à un actif ou encore le compromettre.

Elle existe en corrélation avec des vulnérabilités il peut y avoir aussi plusieurs menaces pour chaque vulnérabilité. La connaissance des différents types de menaces peut aider dans la détermination de leurs dangerosités et des contrôles adaptés permettant de réduire leur impact potentiel.

La menace est une source effective d'incident pouvant entrainer des effets indésirables et graves sur un actif ou un ensemble d'actifs.

Les menaces peuvent être classées par leurs Origine ou source, Type, Motivation, action.

Les principales menaces effectives auxquelles un système d'information peut être confronté sont :

- **Un utilisateur du système :** l'énorme majorité des problèmes liés à la sécurité d'un système d'information est l'utilisateur, généralement insouciant.
- **Une personne malveillante (Hackers et crackers) :** une personne parvient à s'introduire sur le système, légitimement ou non, et à accéder ensuite à des données ou à des programmes auxquels elle n'est pas censée avoir accès en utilisant par exemple des failles connues et non corrigées dans les logiciels.
- **Un programme malveillant :** un logiciel destiné à nuire ou à abuser des ressources du système est installé par mégarde ou par malveillance sur le système, ouvrant la porte à des intrusions ou modifiant les données. Des

données personnelles peuvent être collectées à l'insu de l'utilisateur et être réutilisées à des fins malveillantes ou commerciales.

I.2.3. Les attaques [3]

Les attaques se divisent, selon leurs types sur quatre catégories et leurs buts sont :

- **Interruption :** Vise la disponibilité des informations
- **Interception :** Vise la confidentialité des informations
- **Modification :** Vise l'intégrité des informations
- **Fabrication :** Vise l'authenticité des informations

I.2.3.1. Les attaques par programmes malveillants [3]

- **Les virus :** Un virus est un logiciel capable de s'installer sur un ordinateur à l'insu de son utilisateur légitime. Le terme virus est réservé aux logiciels qui se comportent ainsi avec un but malveillant, parce qu'il existe des usages légitimes de cette technique dite de *code mobile* : les appliquettes Java et les procédures JavaScript sont des programmes qui viennent de s'exécuter sur un ordinateur en se chargeant à distance depuis un serveur Web, et en principe avec un motif légitime. Les concepteurs de Java et de JavaScript nous assurent qu'ils ont pris toutes les précautions nécessaires pour que ces programmes ne puissent pas avoir d'effet indésirable sur un ordinateur, bien que ces précautions, comme toutes précautions, soient faillibles. Les appliquettes Java s'exécutent dans un bac à sable *(sandbox)* qui en principe les isole totalement du système de fichiers qui contient des documents ainsi que du reste de la mémoire de l'ordinateur.
  - **Cheval de Troie :** Un cheval de Troie *(Trojan horse)* est un logiciel qui se présente sous un jour honnête, utile ou agréable, et qui une fois installé sur un ordinateur y effectue des actions cachées et pernicieuses.
  - **Porte dérobée :** Une porte dérobée *(backdoor)* est un logiciel de communication caché, installé par exemple par un virus ou par un cheval de Troie, qui donne à un agresseur extérieur accès à l'ordinateur victime, par le réseau.

- **Bombe logique :** Une bombe logique est une fonction, cachée dans un programme en apparence honnête, utile ou agréable, qui se déclenchera à retardement, lorsque sera atteinte une certaine date, ou lorsque surviendra un certain événement. Cette fonction produira alors des actions indésirées, voire nuisibles.
- **Logiciel espion :** Un logiciel espion, comme son nom l'indique, collecte à l'insu de l'utilisateur légitime des informations au sein du système où il est installé, et les communique à un agent extérieur, par exemple au moyen d'une porte dérobée. Une variété particulièrement toxique de logiciel espion est le *keylogger* (espion dactylographique ?), qui enregistre fidèlement tout ce que l'utilisateur tape sur son clavier et le transmet à son honorable correspondant ; il capte ainsi notamment identifiants, mots de passe et codes secrets.

I.2.3.2. Les attaques par messagerie

- **Courrier électronique non sollicité (spam) :** Le courrier électronique non sollicité *(spam)* consiste en « communications électroniques massives, notamment de courrier électronique, sans sollicitation des destinataires, à des fins publicitaires ou malhonnêtes », selon Wikipédia. Ce n'est pas à proprement parler du logiciel, mais les moyens de le combattre sont voisins de ceux qui permettent de lutter contre les virus et autres malfaisances, parce que dans tous les cas il s'agit finalement d'analyser un flux de données en provenance du réseau pour rejeter des éléments indésirables.
- **L'Hameçonnage (phishing en anglais) :** un courrier électronique dont l'expéditeur se fait généralement passer pour un organisme financier et demandant au destinataire de fournir des informations confidentielles.
- **Le Canular informatique (hoax en anglais) :** un courrier électronique incitant généralement le destinataire à retransmettre le message à ses contacts sous divers prétextes. Ils encombrent le réseau, et font perdre du temps à leurs destinataires. Dans certains cas, ils incitent l'utilisateur à effectuer des manipulations dangereuses sur son poste (suppression d'un fichier prétendument lié à un virus par exemple).

I.2.3.3. Les attaques sur le réseau

- **Intrusion :** L'intrusion dans un système informatique a généralement pour but la réalisation d'une menace et est donc une attaque. Les conséquences peuvent être catastrophiques : vol, fraude, incident diplomatique, chantage…
  Le principal moyen pour prévenir les intrusions est le pare-feu ("firewall"). Il est efficace contre les fréquentes attaques de pirates amateurs, mais d'une efficacité toute relative contre des pirates expérimentés et bien informés. Une politique de gestion efficace des accès, des mots de passe et l'étude des fichiers « log » (traces) est complémentaire.
- **Écoute du réseau (sniffing) :** Il existe des logiciels qui, permettent d'intercepter certaines informations qui transitent sur un réseau local, en retranscrivant les trames dans un format plus lisible (Network packet sniffing). C'est l'une des raisons qui font que la topologie en étoile autour d'un hub n'est pas la plus sécurisée, puisque les trames qui sont émises en « broadcast » sur le réseau local peuvent être interceptées. De plus, l'utilisateur n'a aucun moyen de savoir qu'un pirate a mis son réseau en écoute. L'utilisation de switchs (commutateurs) réduit les possibilités d'écoute mais en inondant le commutateur, celui-ci peut se mettre en mode « HUB » par sécurité.
- **Le déni de service (Denial of service) :** L'attaquant n'obtient pas un accès au système informatique sur le réseau mais il parvient à mettre en panne certains composants stratégiques (le serveur de messagerie, le site web, etc). Le but d'une telle attaque n'est pas de dérober des informations sur une machine distante, mais de paralyser un service ou un réseau complet. Les utilisateurs ne peuvent plus alors accéder aux ressources. Les deux exemples principaux, sont le « Ping flood » ou l'envoi massif de courriers électroniques pour saturer une boîte aux lettres (mailbombing). La meilleure parade est le firewall ou la répartition des serveurs sur un réseau sécurisé.
- **IP Spoofing :** Usurpation d'adresse IP, on fait croire que la requête provient d'une machine autorisée. Une bonne configuration du routeur d'entrée permet d'éviter qu'une machine extérieure puisse se faire passer pour une machine interne.

- **DNS Spoofing :** Pousse un serveur DNS à accepter l'intrus. Pour l'éviter, il est intéressant de séparer le DNS du LAN de celui de l'espace public.

I.2.3.4. Les attaques sur les mots de passe

- **L'attaque par dictionnaire :** le mot testé est pris dans une liste prédéfinie contenant les mots de passe les plus courants et aussi des variantes de ceux-ci (à l'envers, avec un chiffre à la fin, etc.). Ces listes sont généralement dans toutes les langues les plus utilisées, contiennent des mots existants, ou des diminutifs (comme par exemple "powa" pour "power", ou "G0d" pour "god").
- **L'attaque par force brute :** toutes les possibilités sont faites dans l'ordre jusqu'à trouver la bonne solution (par exemple de "aaaaaa" jusqu'à "ZZZZZZ" pour un mot de passe composé strictement de six caractères alphabétiques).

I.3. Conclusion

Dans cette partie nous avons présenté la problématique de la sécurité informatique ainsi les différentes attaques. Dans la partie qui suit on va vous présenté quelques mécanismes de sécurisation.

Partie 2 ***Mécanismes de sécurisation***

II.1. Introduction

La sécurité du transport de l'information est une préoccupation primordiale dans le domaine des réseaux. Cette partie de ce chapitre présente une vue général sur les mécanismes fondamentaux de la sécurité mis en œuvre dans les réseaux, et quelques notions de base dans la cryptographie.

II.2. Les services de sécurité

En informatique, le terme sécurité recouvre tout ce qui concerne la protection des informations. Trois grands concepts de sécurité ont été définis :

- Les fonctions de sécurité, qui sont déterminées par les actions pouvant compromettre la sécurité d'un établissement.
- Les mécanismes de sécurité, qui définissent les algorithmes à mettre en œuvre.
- Les services de sécurité, qui représentent les logiciels et les matériels mettant en œuvre des mécanismes dans le but de mettre à la disposition des utilisateurs les fonctions de sécurité dont ils ont besoin.

  Cinq types de service de sécurité ont été définis sont:

  - **La confidentialité**, qui doit assurer la protection des données contre les attaques non autorisées.
  - **L'authentification**, qui doit permettre de s'assurer que celui qui se connecte est bien celui qui correspond au nom indiqué.
  - **L'intégrité**, qui garantit que les données reçues sont exactement celles qui ont été émises par l'émetteur autorisé.
  - **La non-répudiation**, qui assure qu'un message a bien été envoyé par une source spécifiée et reçu par un récepteur spécifié.
  - **Le contrôle d'accès**, qui a pour fonction de prévenir l'accès à des ressources sous des conditions définies et par des utilisateurs spécifiés.

II.3. La sécurité logicielle

II.3.1. Les Firewalls [2]

Un firewall est un système ou un groupe de système qui gère les contrôles d'accès entre deux réseaux. Plusieurs méthodes basées sur le filtrage de données sont utilisées à l'heure actuelle. Deux mécanismes sont utilisés : le premier consiste à interdire le trafic, et le deuxième à l'autoriser.

La méthode de filtrage de données Permet d'analyser, de sécuriser et de gérer le trafic réseau, et ainsi d'utiliser le réseau de la façon pour laquelle il a été prévu et sans l'encombrer avec des activités inutiles, et d'empêcher une personne sans autorisation d'accéder à ce réseau de données.

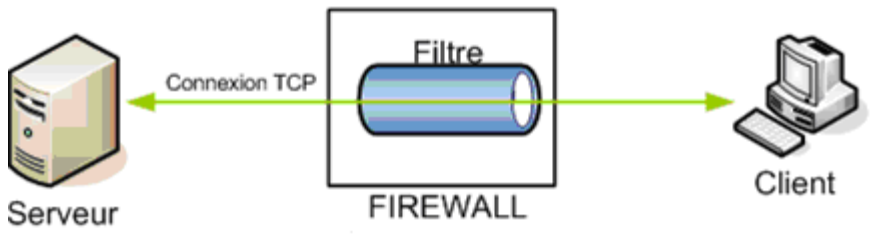

Figure II.1 : Filtrage de données.

Avantages :

- Sécurité en bout de chaîne (le poste client).
- Personnalisable assez facilement.

Inconvénients :

- Facilement contournable

II.3.2. Le serveur Proxy [18]

Un serveur proxy (traduction française de «proxy server», appelé aussi «serveur mandataire») est à l'origine une machine faisant fonction d'intermédiaire entre les ordinateurs d'un réseau local (utilisant parfois des protocoles autres que le protocole TCP/IP) et internet.

La plupart du temps le serveur proxy est utilisé pour le web, il s'agit alors d'un proxy HTTP. Toutefois il peut exister des serveurs proxy pour chaque protocole applicatif (FTP, ...).

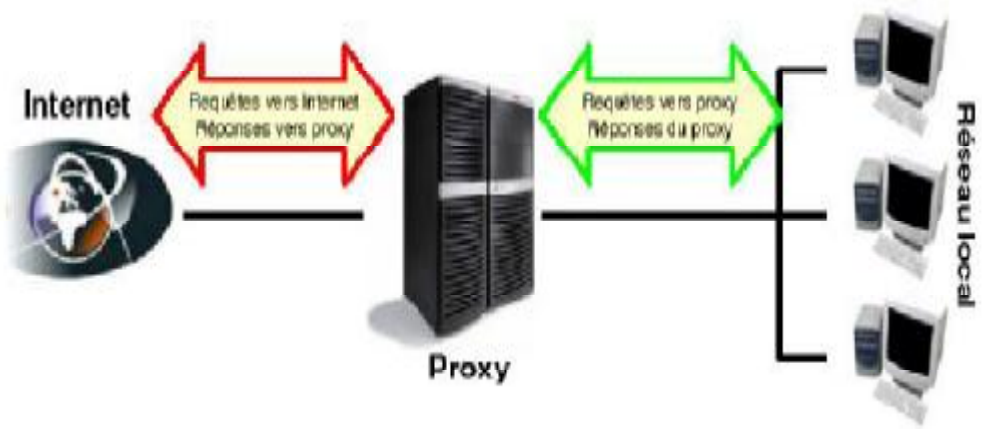

Figure II-2 : Serveur Proxy.

Le principe de fonctionnement basique d'un serveur proxy est assez simple : il s'agit d'un serveur "mandaté" par une application pour effectuer une requête sur Internet à sa place. Ainsi, lorsqu'un utilisateur se connecte à internet à l'aide d'une application cliente configurée pour utiliser un serveur proxy, celle-ci va se connecter en premier lieu au serveur proxy et lui donner sa requête. Le serveur proxy va alors se connecter au serveur que l'application cliente cherche à joindre et lui transmettre la requête. Le serveur va ensuite donner sa réponse au proxy, qui va à son tour la transmettre à l'application cliente.

La plupart des proxys assurent ainsi une fonction de cache (en anglais caching), c'est-à-dire la capacité à garder en mémoire (en "cache") les pages les plus souvent visitées par les utilisateurs du réseau local afin de pouvoir les fournir le plus rapidement possible.

Grâce à l'utilisation d'un proxy, il est possible d'assurer un suivi des connexions (en anglais logging ou tracking) via la constitution de journaux d'activité (logs) en enregistrant systématiquement les requêtes des utilisateurs lors de leurs demandes de connexion à Internet. Il est parfois possible de l'utiliser pour authentifier les utilisateurs, c'est-à-dire de leur demander de s'identifier à l'aide d'un nom d'utilisateur et d'un mot de passe par exemple.

II.3.3. Le Parfeu

La plupart des réseaux privés sont munis d'un pare-feu, est un équipement de réseau qui filtre les communications, un peu comme un routeur d'ailleurs il est possible de configurer un routeur pour lui faire jouer le rôle d'un pare-feu simple. Un routeur doit décider au coup par coup du sort de chaque paquet, avec seulement une

faible possibilité d'analyse historique, alors qu'un pare-feu efficace contre les attaques subtiles doit pouvoir faire des choses plus compliquées. Voila un exemple ci après d'un pare-feu, placé à l'entrée d'une entreprise afin d'empêcher l'entrée ou la sortie de paquets non autorisés par l'entreprise.

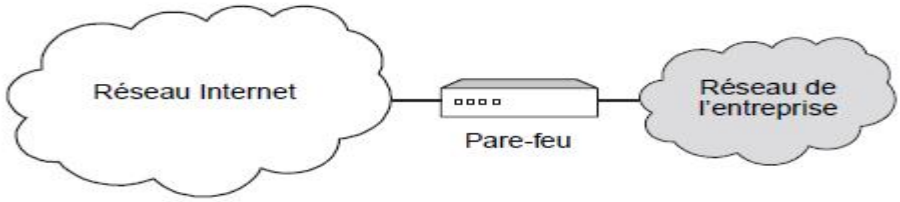

Figure II.3 : Situation d'un pare-feu dans l'entreprise [1]

II.3.4. Les VPNs [2]

Les réseaux privés virtuels (VPN : Virtual Private Network) permettent à l'utilisateur de créer un chemin virtuel sécurisé entre une source et une destination. Avec le développement d'Internet, il est intéressant de permettre ce processus de transfert de données sécurisé et fiable. Grâce à un principe de tunnel[1] (tunnelling) dont chaque extrémité est identifiée, les données transitent après avoir été chiffrées.

Un des grands intérêts des VPN est de réaliser des réseaux privés à moindre coût. En chiffrant les données, tout se passe exactement comme si la connexion se faisait en dehors d'Internet. Il faut par contre tenir compte de la toile (l'Internet), dans le sens où aucune qualité de service n'est garantie.

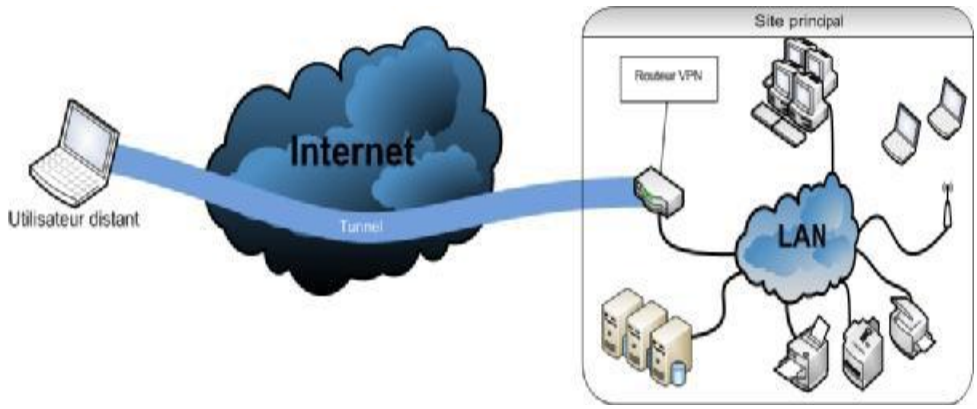

Figure II-4 : Exemple d'un VPN.

v Fonctionnement du VPN :

Un VPN repose sur un protocole, appelé protocole de tunnelisation, c'est-à-dire un protocole permettant aux données passant d'une extrémité à l'autre du VPN d'être sécurisées par des algorithmes de cryptographie. Le terme tunnel est utilisé pour symboliser le fait qu'entre l'entrée et la sortie du VPN les données sont chiffrées et donc normalement incompréhensibles pour toute personne située entre les deux extrémités du VPN, comme si les données passaient dans un tunnel. De plus, créer un tunnel signifie aussi encapsuler un protocole dans un protocole de même niveau du modèle OSI (IP dans IP Sec par exemple). Dans le cas d'un VPN établi entre deux machines, on appelle client VPN l'élément permettant de chiffrer les données à l'entrée et serveur VPN (ou plus généralement serveur d'accès distant) l'élément déchiffrant les données en sortie. Ainsi, lorsqu'un système extérieur à un réseau privé (client nomade, agence ou travailleur à domicile) souhaite se connecter au réseau de son entreprise :

- Les paquets (qui contiennent les données) sont chiffrés par le client VPN (selon l'algorithme décidé par les deux interlocuteurs lors de l'établissement du tunnel VPN) et éventuellement signés.
- Ils sont transmis par le biais du réseau transporteur (internet en général).
- Ils sont reçus par le serveur VPN qui les déchiffre et les traite si les vérifications requises sont correctes.

II.3.5. Les antivirus

La plupart des ordinateurs sont dotés d'un logiciel antivirus pré-intégré capable de détecter et effacer les principales menaces virales s'il est régulièrement mis à jour et correctement entretenu.

II.3.6. Les systèmes de détection d'intrusions

Un système de détection d'intrusion (ou IDS : Intrusion Detection System) est un mécanisme destiné à repérer des activités anormales ou suspectes sur la cible analysée (un réseau ou un hôte). Il permet ainsi d'avoir une connaissance sur les tentatives réussies comme échouées des intrusions, et cela est dû grâce à la consultation du journal d'Audit.

Il existe trois grandes familles distinctes d'IDS :

- Les NIDS (Network Based Intrusion Detection System), qui surveillent l'état de la sécurité au niveau du réseau.
- Les HIDS (HostBased Intrusion Detection System), qui surveillent l'état de la sécurité au niveau des hôtes.
- Les IDS hybrides, qui utilisent les NIDS et HIDS pour avoir des alertes plus pertinentes.

Les HIDS sont particulièrement efficaces pour déterminer si un hôte est contaminé et les NIDS permettent de surveiller l'ensemble d'un réseau contrairement à un HIDS qui est restreint à un hôte.

II.4. La cryptographie

Les récents développements de la cryptographie permettent de résoudre les nombreux problèmes menaçant la vie privée ou la sécurité sur internet, la cryptographie est l'étude des principes, méthodes et techniques mathématiques reliés aux aspects de sécurité de l'information.

La cryptographie nous permet de stoker des informations sensibles ou de les transmettre a travers des réseaux non surs (comme internet) de telle sorte qu'elles ne peuvent être lues par personne à l'exception du destinataire convenu.

II.4.1. Signature électronique [17]

Le paradigme de signature électronique (appelé aussi *signature numérique*) est un procédé permettant de garantir l'authenticité de l'expéditeur (fonction d'*authentification*) et de vérifier l'intégrité du message reçu.

La signature électronique assure également une fonction de non-répudiation, c'est-à-dire qu'elle permet d'assurer que l'expéditeur a bien envoyé le message (autrement dit elle empêche l'expéditeur de nier avoir expédié le message). Le SHA, MD4 et MD5 sont des exemples de l'algorithme de signature.

II.4.2. Les certificats [1]

Une difficulté qui s'impose à la station d'un réseau qui communique avec beaucoup d'interlocuteurs consiste à se rappeler de toutes les clés publiques dont elle a besoin pour récupérer les clés secrètes de session. Pour cela, il faut utiliser un service sécurisé et fiable, qui délivre des certificats. Un organisme offrant un service de gestion de clés publiques est une autorité de certification, appelée tiers de

confiance. Cet organisme émet des certificats au sujet de clés permettant à une entreprise de les utiliser avec confiance.

Un certificat est constitué d'une suite de symboles (document M) et d'une signature. Le format de certificat le plus courant provient du standard X.509 v2 ou v3.

II.5. Quelques définitions

II.5.1. La cryptologie

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

II.5.1.1. La cryptographie

La cryptographie est l'étude des méthodes donnant la possibilité d'envoyer des données de manière confidentielle sur un support donné.

II.5.1.2. La cryptanalyse

Opposée à la cryptographie, elle a pour but de retrouver le texte clair à partir de textes chiffrés en déterminant les failles des algorithmes utilisés.

I.5.2. La stéganographie

Sert à cacher des messages secrets dans d'autres messages, généralement l'expéditeur écrit un message inoffensif et dissimile un message secret dedans.

I.5.3. La cryptosystème

Il est défini comme l'ensemble des clés possibles (espace de clés), des textes clairs et chiffrés possibles associés à un algorithme donné.

I.5.4. Texte chiffré

Appelé également cryptogramme, le texte chiffré est le résultat de l'application d'un chiffrement à un texte clair.

II.5.5. Clef

Il s'agit du paramètre impliqué et autorisant des opérations de chiffrement et/ou déchiffrement. Dans le cas d'un algorithme symétrique, la clef est identique lors des deux opérations. Dans le cas d'algorithmes asymétriques, elle diffère pour les deux opérations.

I.5.6. Le chiffrement

Le chiffrement consiste à transformer une donnée (texte, message, ...) afin de la rendre incompréhensible par une personne autre que celui qui a créé le message et celui qui en est le destinataire. La fonction permettant de retrouver le texte clair à partir du texte chiffré porte le nom de déchiffrement.

II.6. Conclusion

Nous avons consacré la première partie de ce chapitre à la présentation de la Problématique de la sécurité informatique, les menaces, les attaques… et dans la deuxième partie nous avons décrire quelques mécanismes de sécurisation.

Dans le chapitre suivant nous détaillerons beaucoup plus cette notion de cryptographie en donnons quelques algorithmes utilisés ainsi que la sécurisation en utilisant les fonctions de hachages.

Partie 1 **La cryptographie**

I.1. Introduction

Ce chapitre est divisé en trois parties, dans la première partie on va vous présenté les différents types de la cryptographie et leurs algorithmes de chiffrement, puis dans la deuxième partie on vous donne le fonctionnement des différentes fonctions de hachage et enfin la troisième partie consiste a étudié un cas pratique en utilisant la fonction de hachage SHA-1.

I.2. La cryptographie classique [19]

I.2.1. Substitution monoalphabétique

Chaque lettre est remplacée par une autre lettre ou symbole. Parmi les plus connus, on citera le chiffre de César, le chiffre affine, ou encore les chiffres désordonnés. Tous ces chiffres sont sensibles à l'analyse de fréquence d'apparition des lettres (nombre de fois qu'apparait une même lettre dans un texte). De nos jours, ces chiffres sont utilisés pour le grand public, pour les énigmes de revues ou de journaux.

I.2.1. 1. Chiffre affine

On dit qu'une fonction est affine lorsqu'elle est de la forme x -> a * x + b, c'est-à-dire un polynôme de degré 1. Une fonction linéaire est une fonction affine particulière.

L'idée est d'utiliser comme fonction de chiffrement une fonction affine du type

y = (ax + b) mod 26,

Où a et b sont des constantes, et où x et y sont des nombres correspondant aux lettres de l'alphabet

(A=0,B=1,...). On peut remarquer que si a = 1, alors on retrouve le chiffre de César où b est le décalage (le k du chiffre de César).

I.2.2. Chiffrement polygraphique

Il s'agit ici de chiffrer un groupe de n lettres par un autre groupe de n symboles. On citera notamment le chiffre de Playfair et le chiffre de Hill. Ce type de chiffrement porte également le nom de substitutions polygrammiques.

I.2.2.1. Chiffre de Playfair (1854)

On chiffre 2 lettres par 2 autres. On procède donc par digramme. On dispose les 25 lettres de l'alphabet (W exclu car inutile à l'époque, on utilise V à la place) dans une grille de 5x5, ce qui donne la clef. La variante anglaise consiste à garder le W et à fusionner I et J.

Il y a 4 règles à appliquer selon les deux lettres à chiffrer lors de l'étape de substitution. Pour le déchiffrement, on procède dans l'ordre inverse.

1. Si les lettres sont sur des "coins", les lettres chiffrées sont les 2 autres coins.
Exemple : OK devient VA, RE devient XI ...
2. Si les lettres sont sur la même ligne, il faut prendre les deux lettres qui les suivent immédiatement à leur droite.
3. Si les lettres sont sur la même colonne, il faut prendre les deux lettres qui les suivent immédiatement en dessous.
4. Si elles sont identiques, il faut insérer une nulle (habituellement le X) entre les deux pour éliminer ce doublon.

Exemple : "balloon" devient "ba" "lx" "lo" "on".

I.2.3. cryptographie par transpositions

Elles consistent, par définition, à changer l'ordre des lettres. C'est un système simple, mais peu sûre pour de très brefs messages car il y a peu de variantes. Ainsi, un mot de trois lettres ne pourra être transposé que dans 6 (=3!) positions différentes. Par exemple, "col" ne peut se transformer qu'en "col", "clo", "ocl", "olc", "lco" et "loc".

Lorsque le nombre de lettres croît, il devient de plus en plus difficile de retrouver le texte original sans connaître le procédé de brouillage. Ainsi, une phrase de 35 lettres peut être disposée de 35! = 10^{40} manières différentes. Ce chiffrement nécessite un procédé rigoureux convenu auparavant entre les parties.

Une transposition rectangulaire consiste à écrire le message dans une grille rectangulaire, puis à arranger les colonnes de cette grille selon un mot de passe donné (le rang des lettres dans l'alphabet donne l'agencement des colonnes).

Exemple :

Dans la figure, on a choisi comme clef GRAIN pour chiffrer le message SALUT LES PETITS POTS.En remplissant la grille, on constate qu'il reste deux cases vides, que l'on peut remplir avec des nulles (ou pas, selon les désirs des correspondants).

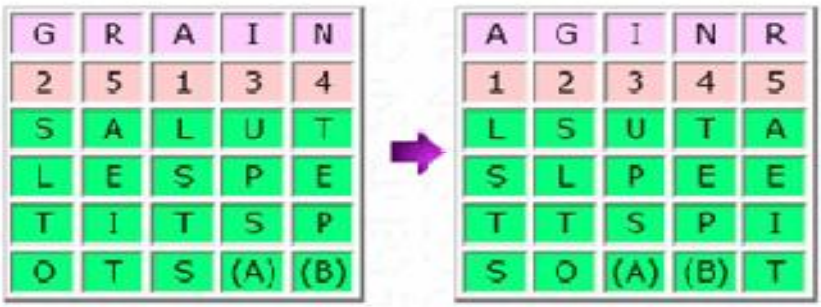

Figure III.1 : Application d'une transposition[19]

I.3. La cryptographie moderne

I.3.1. Le cryptage symétrique

Dans la cryptographie symétrique, aussi appelée chiffrement à clé secrète, la clé, est identique au chiffrement qu'au déchiffrement. Cette clé, unique, doit donc être gardée secrète par son propriétaire. L'inconvénient évident de ce système, est que l'émetteur et le destinataire du message doivent avoir convenu de la clé avant l'envoi du message. Ils doivent donc disposer d'un canal sûr, pour s'échanger la clé. Ceci pose problème sur Internet où il n'y a pas de canaux sûrs. Un exemple d'un tel algorithme est le chiffrement de CESAR.**[2]**

I.3.1.1. Le principe de base

Un expéditeur et un destinataire souhaitant communiquer de manière sécurisée à l'aide du cryptage conventionnel doivent convenir d'une clé et ne pas la divulguer. Dans la majorité des systèmes de cryptage symétrique la clé de chiffrement et la clé de déchiffrement sont identiques.

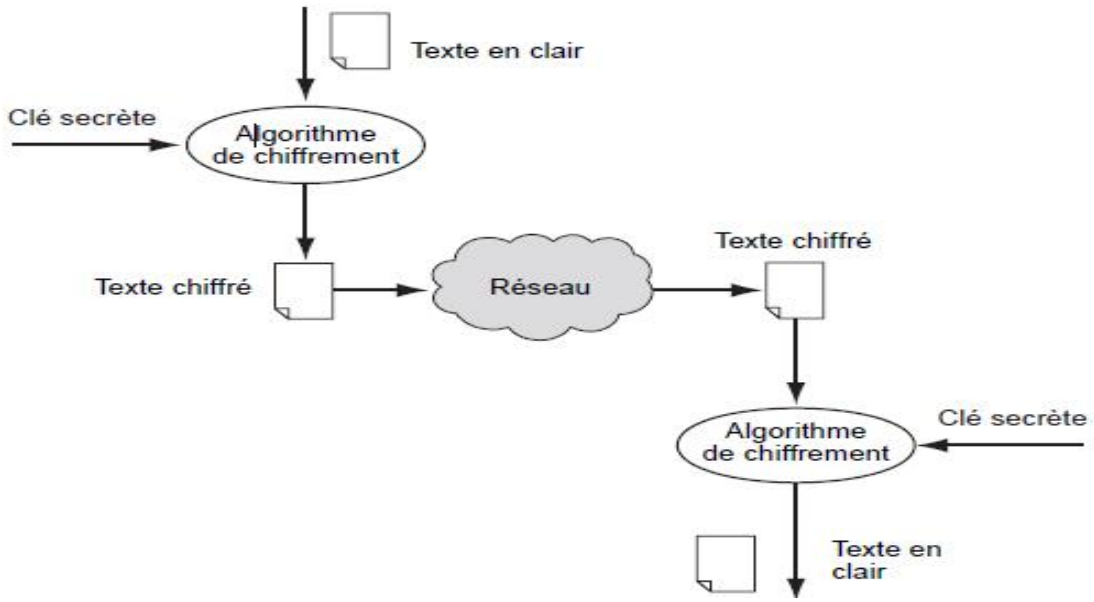

Figure III.2. Algorithme de chiffrement symétrique[1]

I.3.1.2. Exemples de cryptage symétrique

v Le chiffrement de CESAR

Un exemple extrêmement simple de chiffrement conventionnel est un chiffre à substitution. Un chiffre à substitution remplace un morceau d'information par un autre. Le plus souvent, cela est effectué en décalant des lettres de l'alphabet. Dans le cas du chiffrement de Jules César, l'algorithme consiste en un décalage de l'alphabet, et la clé est le nombre de caractères à décaler. Par exemple, si nous codons le mot "INFORMATIQUE" en utilisant une valeur de 3 pour la "clé de César", nous décalons l'alphabet de telle sorte que la quatrième lettre en descendant (D) commence l'alphabet.

Donc en commençant avec :ABCDEFGHIJKLMNOPQRSTUVWXYZ

En faisant pour le to0ut 3 décalages de suite, on obtient :

DEFGHIJKLMNOPQRSTUVWXYZABC

Où D=A, E=B, F=C, et ainsi de suite.

En utilisant ce schéma, le texte clair, " INFORMATIQUE " se chiffre comme "LQIRPDWLTXH ".

Il existe plusieurs algorithmes de chiffrement symétrique. Les algorithmes les plus connus sont : Kerberos, DES (Data Encryption Standard).

I.3.1.3. Avantages et inconvénients de cryptage symétrique

v Avantages

- rapidité du chiffrement/déchiffrement

v Inconvénients

- échange de la clé ***k*** par un autre canal,
- Pour ***n*** participants on a besoin de ***n(n-1)/2*** clés secrètes enregistrées

I.3.2. Le cryptage asymétrique [20]

Les problèmes de distribution de clé sont résolus par la cryptographie à clé publique,dont le concept fut inventé par WhitfieldDiffie et Martin Hellman en 1975.

La cryptographie à clé publique repose sur un schéma asymétrique qui utilise une pairede clés pour le chiffrement, une clé publique qui chiffre les données, et une clé privée(secrète) correspondante, aussi appelée clé secrète, qui sera utilisée pour le déchiffrement.Vous pouvez publier largement votre clé publique, tout en gardant votre clé privée secrète.

Toute personne en possession d'une copie de votre clé publique peut ensuite chiffrerdes informations que vous seul pourrez lire. Même des gens que vous n'avez jamaisrencontrés.

Il est mathématiquement impossible de déduire la clé privée de la clé publique. Toutepersonne ayant une clé publique peut chiffrer des informations mais ne peut pas les déchiffrer.Seule la personne qui a la clé privée correspondante peut déchiffrer les informations.

Le principal avantage de la cryptographie à clé publique est qu'elle permet à des gensqui n'ont pas d'accord de sécurité préalable d'échanger des messages de manière sûre. Lanécessité pour l'expéditeur et le destinataire de partager des clés secrètes via un canal sûr estéliminée; toutes les communications impliquent uniquement des clés publiques, et aucune cléprivée n'est jamais transmise ou partagée.

I.3.2.1. Le principe de base

Le principe est donc de distribuer la clé publique tout en conservant la clé privée secrète. Tout utilisateur possédant une copie de la clé publique pourra ensuite crypter des informations que seul le propriétaire de la clé privée pourra déchiffrer.

Il faut également noter que si le cryptage est bien possible à l'aide de la clé publique, l'opération inverse (le décryptage) ne sera pas possible au moyen de la clé publique mais exigera l'utilisation de la clé privée correspondante.

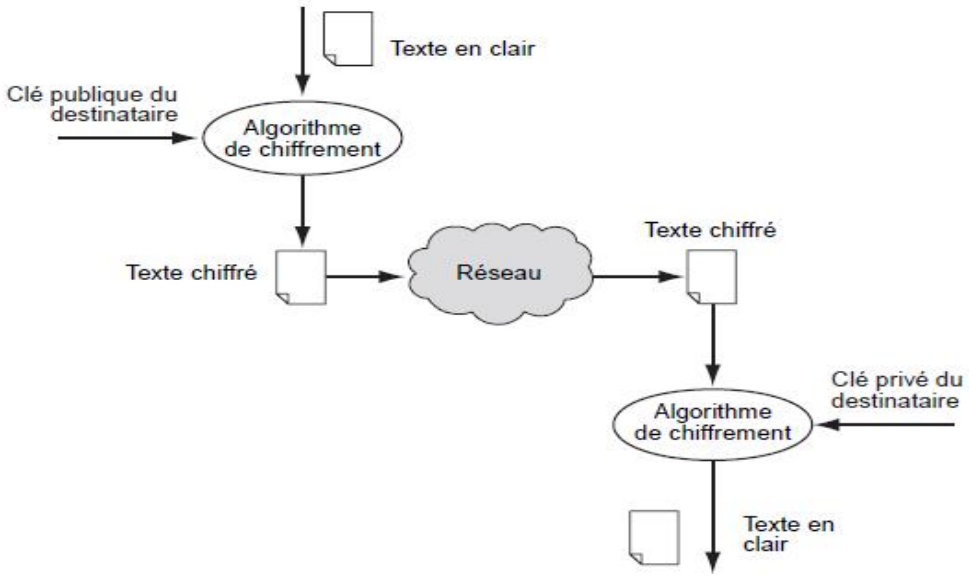

Figure III.3 : Algorithme de chiffrement asymétrique[1]

I.3.2.2. Exemples de cryptage asymétrique

v Le cryptosystème RSA[3]

Inventé par **R**ivest, **S**hamir et **A**dleman en 1978, RSA permet le chiffrement de façon très sure. Il est aujourd'hui encore très largement utilisé. Cet algorithme repose sur la difficulté de factoriser des grands nombres.

Voici comment se fait la génération des paires de clés :

- On commence par choisir deux grands nombres premiers, p et q, et on calcule n = p*q.

n est rendu public ; p et q doivent rester secrets et sont donc détruits une fois les clés générées.

- On choisit ensuite aléatoirement une clé publique e telle que e et (p-1)*(q-1) soient premiers entre eux.

- La clé privée d est obtenue grâce à l'algorithme d'Euclide : e*d ≈1 mod (p-1)*(q-1).

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* = 3 et *q* = 11.

-Calculer *n* = *p***q*, soit dans notre exemple *n* = 33.

-Calculer *z* = (*p* − 1)*(*q* − 1). (Ce nombre est la valeur de la fonction *Ø*(*n*), dite fonction phi d'Euler, et on notera qu'elle donne la taille du groupe multiplicatif modulo *n*,$\mathbf{Z_n^*}$Dans notre exemple *z* = 20.

- Prendre un petit entier *e*, impair et premier avec *z*, soit *e* = 7. Dans la pratique, *e* sera toujours petit devant n^2.

- Calculer l'inverse de *e* (mod*z*), c'est-à-dire *d* tel que *e.d*= 1 mod*z*. Les théorèmes de l'arithmétique modulaire nous assurent que, dans notre cas, *d* existe et est unique. Dans notre exemple *d* = 3.

- La paire *P* = (*e, n*) est la clé publique.

- Le triplet *S* = (*d, p, q*) est la clé privée.

Voyons ce que donne notre exemple. La clé publique de Boris est donc (7*,* 33). Aïcha veut lui envoyer un message *M*, disons le nombre 19. Elle se procure la clé publique de Boris sur son site Web et elle procède au chiffrement de son message *M* pour obtenir le chiffré *C* comme ceci :

$$C = P(M) = M^e \bmod n$$ *(la fonction de l'algorithme* chiffrement*)*

$$C = P(19) = 19^7 \bmod 33 = 13$$

Pour obtenir le texte clair *T*, Boris décode avec sa clé secrète ainsi :

$$T = S(C) = C^d \bmod n$$ *(la fonction de l'algorithme* déchiffrement*)*

$$T = S(13) = 13^3 \bmod 33 = 19$$

En fait c'est très logique :

$$S(C) = C^d \bmod n$$

$$= (M^e)^d \bmod n$$

$$= M^{e,d} \bmod n$$

$= M \bmod n$

Le dernier résultat, $M^{e,d} = M \pmod n$ découle du fait que *e* et *d* sont inversesmodulo *n*.

I.3.2.3. Avantages et inconvénients de cryptage asymétrique

v Avantages

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

v Inconvénients

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

I.3.3. La signature numérique

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 $M^e \bmod n$, 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.

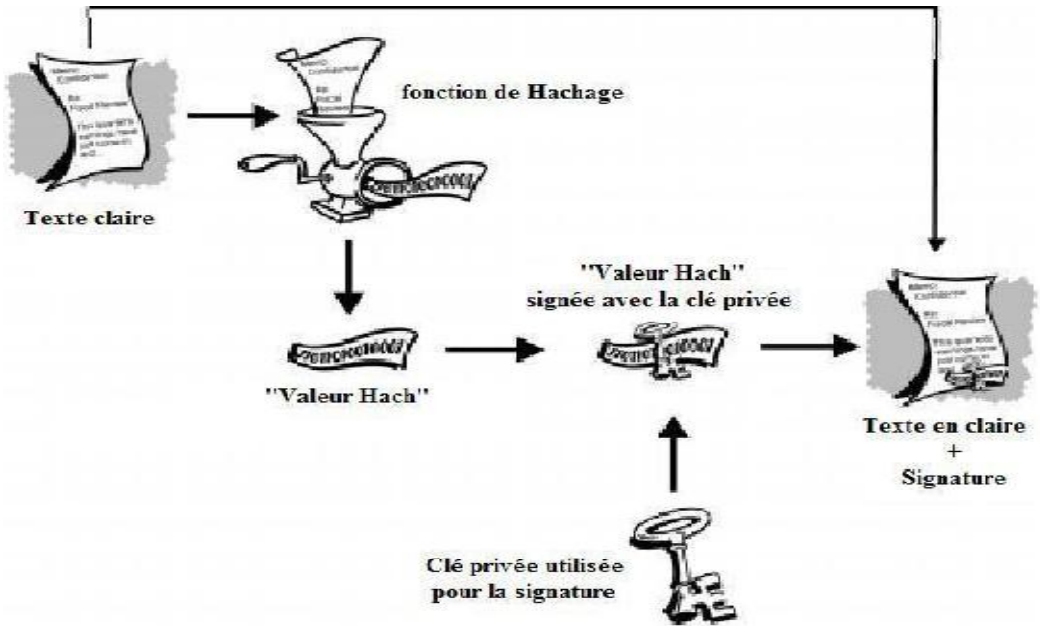

Figure III.4 : Le cryptage avec signature.[2]

v 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

Le système décrit précédemment comporte certains inconvénients. Il est lent et produit un volume important de données (au moins le double de la taille des informations d'origine).

Même ci la cryptographie est un moyen de sécurisation, mais elle reste toujours faible de résoudre certains problèmes comme Convertir des entrées de taille variable vers unetaille fixe et le calcule d'empreintes uniques.

Dans la partie suivante on va vous présenté les fonctions de hachage qui résoudre le problème définit précédemment.

Partie 2 Les fonctions de hachage cryptographiques

II.1. Introduction

Les fonctions de hachage servent à rendre plus rapide l'identification des données grâce à son fonctionnement qui permet de calculer l'empreinte d'une donnée et qui ne doit coûter qu'un temps négligeable. Une fonction de hachage doit par ailleurs éviter autant que possible les collisions (états dans lesquels des données différentes ont une empreinte identique) : dans le cas des tables de hachage, ou de traitements automatisés, les collisions empêchent la différenciation des données ou, au mieux, ralentissent le processus.

II.2. Principe des fonctions de hachage

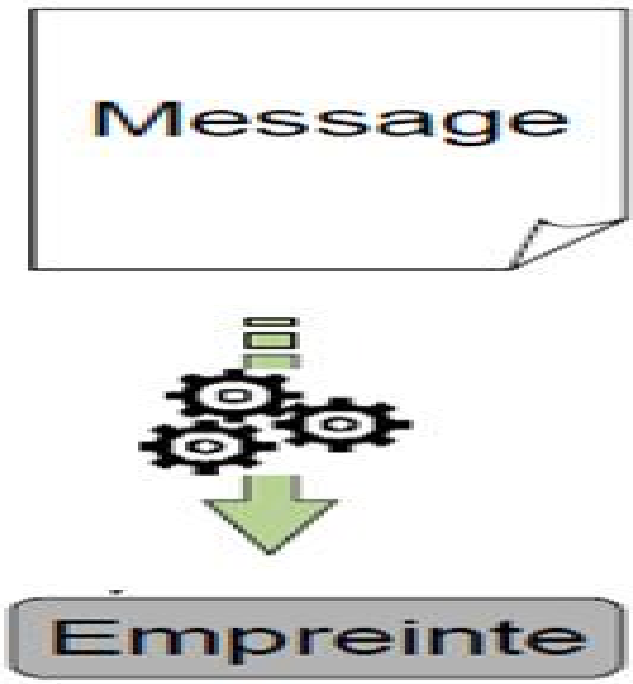

Figure III.5. Principe fonction de hachage.[2]

Cette fonction traite une entrée de longueur variable**m** (dans ce cas, un message pouvant contenir indifféremment des milliers ou des millions de bits), afin d'obtenir en sortie un élément de longueur fixe**n**, par exemple 128bits, appelée « valeur hash » En cas de modification des données (même d'un seul bit), la fonction

de hachage garantit la production d'une « valeur hash » différente. On utilise le résumé (hash) et la clé privée pour générer la « signature ».

H : $\{0,1\}^*$ **à** $\{0,1\}^n$. m **à** H(m)

Le logiciel de cryptographie transmet en même temps la signature et le texte en clair. A la réception du message par le destinataire, son logiciel traite à nouveau le message, vérifiant ainsi la signature.

Si une fonction de hachage sécurisée est utilisée, il est impossible de récupérer la signature d'un document pour la joindre à un autre document ou d'altérer un message signé.

En effet, la moindre modification apportée à un document signé entraîne l'échec du processus de vérification de la signature numérique.

II.3. Propriétés de la fonction de hachage [22]

Les fonctions de hachage doivent posséder plusieurs propriétés utiles en cryptographie.Les principales sont la résistance aux attaques recherchant des collisions, des préimages ou des secondes préimages.

- **Résistance à la recherche decollision :**l'attaque générique est moins triviale. Elle utilise le paradoxe des anniversaires : pour trouver deux valeurs identiques parmi a valeurs possibles, il suffit d'en tirer aléatoirement$\sqrt{a}$ pour avoir une bonne probabilité de succès. Cela s'explique par l'observation qu'en tirant$\sqrt{a}$ éléments, on forme approximativement$(\sqrt{a})^2/2 = a/2$ paires possibles, ce qui est suffisant pour obtenir une bonne probabilité de succès étant donné la taille de l'ensemble de tirage. Ainsi, nous devons utiliser $2^{n/2}$ messages pour trouver une collision pour une fonction de hachage idéale.trouver deux messages distincts M1 et M2, tels que H(M1) = H(M2) **à Une attaque naïve coûte $2^{n/2}$**.
- **Résistance à la recherche depréimage :** On définit donc l'impossibilité d'un attaquant relativement à une certaine quantité d'opérations, déterminée par la meilleure attaque générique contre une fonction de hachage idéale. Ainsi, un attaquant ne doit pas être en mesure de trouver une préimage en moins de $O(2^n)$ opérations, puisque la meilleure attaque générique consiste à essayer

$O(2^n)$ messages distincts pour avoir une bonne probabilité de succès d'obtenir une solution.

Étant donné un haché H1 choisi aléatoirement, trouver un message M1 tel que H(M1) = H1 **à Une attaque naïve coûte 2^n**.

- **Résistance à la recherche deseconde préimage :**Le raisonnement est identique pour la seconde préimage, $O(2^n)$ opérations sont nécessaires dans le cas d'une fonction de hachage idéale.

Etant donné un message M1 choisi aléatoirement, trouver un message distinct M2 tel que H(M1) = H(M2) **à Une attaque naïve coûte $2^{n/2} < ? < 2^n$**.

II.4.Domaines d'utilisation des fonctions de hachage cryptographiques[21]

Les fonctions de hachage cryptographiques possèdent de nombreux domaines d'utilisation, on les qualifie parfois de la cryptographie. Chacun de ces domaines requiert des propriétés particulières de sécurité, nous dressons dans cette section une liste non exhaustive de ces domaines.

II.4.1. L'intégrité des données

Il s'agitdela fonctionnalité principaledemandéeà une fonctiondehachage cryptographique.Ellepermetdevérifierquedesdonnées n'ont pasétéaltéréesdepuis leur créationoulorsdeleurtransmission.Le moindre changementdanslesdonnéesdoit,avec unetrèsgrandeprobabilité,aboutirà l'obtention d'empreintes différentes.Historiquement,lespremières fonctionsproposéespour assurer cette fonctionnalitéétaientfondéessurlathéorie des codesetétaientnommées codesde détectiondemanipulations(MDCpourManipulation DétectionCodes).

II.4.2. Authentification de messages

Lapropriétéd'intégriténepermetpasdese prémunir contreunadversaireactifquiessaieraitd'altérer malicieusementlesdonnées.Unmoyendepalier ce problème consisteà procéderà l'authentificationdela sourcedesdonnéesenutilisant des codes d'authentificationdemessage(MACpour Message AuthentificationCodes). L'objectif d'un code d'authentificationdemessageestdouble:ildoitpermettre d'authentifier la sourced'unmessageetde vérifier l'intégritédesdonnéessansl'aided'un

mécanismeadditionnel. L'authentificationdemessagesrelèvedudomainede la cryptographiesymétrique,l'utilisation d'une clé secrèteétant nécessaire.

II.4.3. Signature électronique

Les schémasdesignature électroniquesontsansauundoute l'applicationlaplusimportantedes fonctionsde hachage cryptographiques.Unesignature électroniqueestunéquivalent électroniqued'unesignature écrite.

Elle permetdeplus,de détectersil'informationsignéeaétéaltéréeaprèssasignature.Lesalgorithmesutiliséspour signerdesdonnées nécessitent des calculsimportants et sont donc relativement lents comparativementauxvitesses d'exécutiondes fonctions de hachage.Aussi,affin d'accélérerles procéduresdesignatureetde vérificationde signature,onutiliseune fonctionde hachage cryptographique pour calculer l'empreinte desdonnéesàsigneretappliquerl'algorithmedesignature à cetteempreinte.

II.4.4. Protection de mots de passe

Uneautre application courantedes fonctionsde hachage cryptographiquesestla protectiondemotsdepasse.Unmotde passe est une chaîne de caractèresutiliséepour authentifierl'identitéd'unutilisateur ou autoriser l'accèsaux ressourcesd'unsystèmeinformatique.Ilest nécessairede protégerlesmotsdepasse afin deles stocker.Une solution courante consisteànestocker queleur empreinte calculéeenappliquantune fonctionde hachage cryptographique à une combinaisondumotdepasseetd'unsel(Salt).

II.4.5. Dérivation de clé

Dans le cadrede la cryptographiesymétrique,lesparties partagent une clé secrète commune.Ilestalorsfréquentque différentes clés supplémentairessoient nécessairespour différentesapplications.Ladérivation de clé(ou diversificationde clé) consisteàgénéreruneou plusieurs clésàpartird'une même valeur secrète.

Cetteutilisationdes fonctionsde hachage cryptographiquesa pour but d'empêcherunadversaireayantobtenu une clédérivéed'obtenirdes informations surlavaleur secrèteoules autres clésdérivées.

II.5. Lesfonctions de compression [21]

La fonction de compression est la principale composante d'une fonction de hachage itérée. En pratique, c'est aussi souvent la partie la plus vulnérable et donc la plus difficile à construire. Historiquement, on peut distinguer trois manières de procéder pour créer une telle primitive : à partir de rien, en utilisant un algorithme de chiffrement par blocs, ou en se fondant sur un problème difficile.

II.5.1. Les fonctions de compression ad hoc

Les fonctions de compression ad hoc sont en pratique les plus rapides puisqu'elles utilisent des opérations très peu coûteuses. Le désavantage est que la sécurité de ces fonctions est en général conjecturée et n'est pas fondée sur des preuves réductionnistes, car le principe de construction vise avant tout la performance. Les plus connues font partie de la famille MD-SHA.

On peut tout de même remarquer que même si le fonctionnement interne de ces fonctions ne repose sur aucune primitive cryptographique déjà connue, elles définissent en fait le plus souvent un algorithme de chiffrement par blocs ad hoc E qui est imbriqué dans un schéma de type Davies-Meyer.

$$H_i = h\,(H_{i-1},\ M_i) = E_{M_i}(H_{i-1}) \oplus H_{i-1}$$

Où $E_x(y)$ représente le chiffrement du message y par la clé x. D'ailleurs, des algorithmes de chiffrement par blocs déduits d'une fonction de compression ad hoc ont parfois été proposés. Par exemple, SHACAL est extrait de la fonction de compression de SHA-1.

II.5.2. Les fonctions de compression fondées sur un algorithme de chiffrement par blocs

Les fonctions de compression fondées sur un algorithme de chiffrement par blocs paraissent intéressantes pour les concepteurs. En effet, nous connaissons de nombreux candidats efficaces et sûrs, et cette approche débouche parfois sur des preuves de sécurité. Plus précisément, certaines propriétés de la fonction de hachage (résistance à la recherche de collisions, de préimages ou de secondes préimages) peuvent être démontrées en supposant la primitive interne idéale.

Cela fut le cas pour certains des premiers schémas considérés par Preneel et al. Ils étudièrent tous les candidats simples pour construire une fonction de compression à partir d'un algorithme de chiffrement par blocs. Plus précisément, ils analysèrent toutes les constructions de la forme :

$$H_i = h\,(H_{i-1},\, M_i) = E_{v1}\,(V2) \oplus V3$$

Où V1, V2 et V3 sont des combinaisons linéaires des deux entrées $\mathbf{H_{i-1}}$, $\mathbf{M_i}$. Il y a donc $\mathbf{(2^2)^3 = 64}$ candidats au total, dont seulement 12 furent conjecturés sûrs (une vulnérabilité fut décrite pour chacun des autres candidats). Ce n'est que plusieurs années plus tard que Black et al. Ils démontrèrent la validité de cette conjecture, dans le modèle du chiffrement idéal ou modèle de la boîte noire où l'on considère la primitive interne comme une permutation parfaitement aléatoire. Les plus connus de ces schémas sont ceux de Matyas-Meyer-Oseas, de Davies-Meyer, et de Miyaguchi-Preneel. Ils sont explicités dans la figure après. Chacun d'eux possède un taux d'efficacité égal à 1, défini par le rapport entre le nombre de blocs de message traités et le nombre d'appels à un algorithme de chiffrement par blocs. Néanmoins, il faut noter que pour le schéma de Davies-Meyer il est facile de trouver des points fixes, i.e. des valeurs de la variable de chaînage interne restant identiques après application de la fonction de compression : $\mathbf{h\,(H_{i-1},M_i) = H_{i-1}}$. Il suffit de déchiffrer le chiffré nul pour E avec une clé égale à un message $\mathbf{M_i}$ tiré aléatoirement. Cela fournit une variable de chaînage d'entrée vérifiant :

$$H_i = E_{Mi}\,(H_{i-1}) \oplus H_{i-1} = H_{i-1}$$

Cette vulnérabilité n'est pas très pénalisante puisque la fonction de compression est en principe résistante à la recherche de préimages. Ainsi, il sera très difficile en pratique pour un attaquant d'atteindre un tel point fixe durant l'exécution.

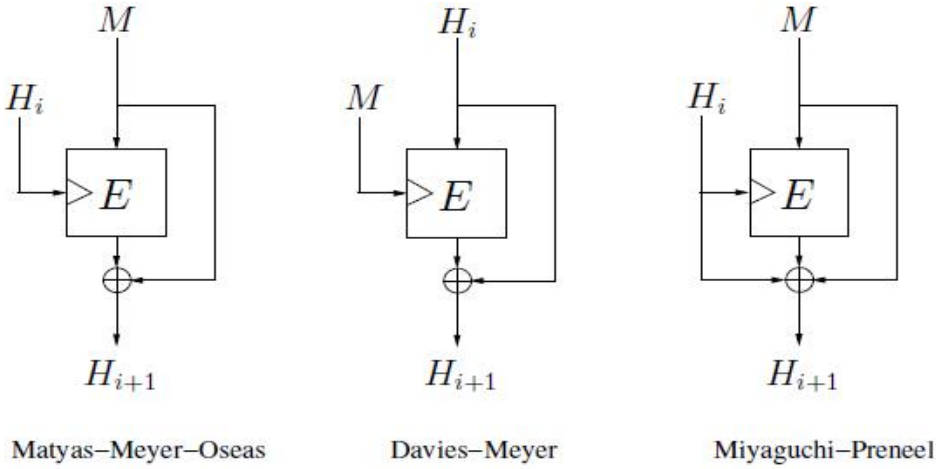

Figure III.6. Les trois schémas les plus connus de fonctions de compression

Les trois schémas les plus connus de fonctions de compression de taille simple fondées sur des algorithmes de chiffrement par blocs. M représente le bloc de message à hacher, $\mathbf{H_i}$ et $\mathbf{H_{i+1}}$ représentent respectivement l'ancienne et la nouvelle variable de chaînage. Enfin, E est un algorithme de chiffrement par blocs dont l'entrée relative à la clé est marqué par une encoche.

II.5.3. Lesfonctions de compression fondées sur une structure algébrique

Le dernier groupe de fonctions de compression, celles fondées sur une structure algébrique,est situé à l'opposé de celui des fonctions ad hoc. En effet, certaines de ces constructions fournissentdes preuves permettant de faire reposer leur sécurité sur un problème supposé difficile,pour l'une ou plusieurs des notions de sécurité habituelles (résistance à la recherche de collisions,de préimages, de secondes préimages). Depuis les récentes attaques dévastatrices contreles membres de la famille MD-SHA, ces fonctions ont connu un regain d'intérêt certain. Cependant,les principaux inconvénients de cette approche sont la nécessité de posséder une trèsgrande quantité de mémoire ou encore une vitesse d'exécution souvent lente à cause d'opérationsalgébriques très coûteuses. De plus, ces fonctions possèdent une forte structure algébriqueet cela peut poser problème puisque les fonctions de hachage sont aussi utilisées pour casserde telles structures dans certaines primitives cryptographiques.

II.6. L'extension de domaine [22]

L'extension de domaine est l'une des deux composantes d'une fonction de hachage itérée. En raison de sa simplicité, l'algorithme de Merkle-Damgård est demeuré longtemps seul et incontesté, mais récemment, des recherches ont montré les limites de cette technique. À présent, de nombreux chercheurs tentent de proposer une nouvelle méthode, à la fois simple et sûre, qui pourrait s'imposer comme nouveau standard.

II.6. L'algorithme de Merkle-Damgård

En 1989, Ralph Merkle et Ivan Damgård proposèrent indépendamment un algorithme d'extension de domaine très simple, dont certaines propriétés de sécurité peuvent être démontrées en supposant certaines propriétés de la fonction de compression interne.

Ø Fonctionnement de L'algorithme de Merkle-Damgård

- On prépare tout d'abord le message à hacher M en y ajoutant un rembourrage.
- La fonction de compression prenant en entrée des blocs de message de taille fixe m, le rembourrage permet de ramener la taille du message à hacher à un multiple de m.
- On rajoute tout d'abord à M un bit à 1 puis u bits à 0, où u est le plus petit nombre positif ou nul tel que la longueur finale du message soit égale à m−v (mod m) (typiquement, v = 64 bits).
- Ensuite, un deuxième rembourrage intervient : on ajoute un bloc de v bits contenant la représentation en base binaire de la taille de M. La taille finale du message rembourré sera bien un multiple de m. On remarque que la taille maximale de message pouvant être hachée est limitée à 2^v bits, mais en pratique ce nombre est assez grand pour ne jamais être atteint.
- Le message rembourré est ensuite divisé en k blocs de message $\mathbf{M_i}$ de m bits chacun, qui serviront à mettre à jour la variable de chaînage $\mathbf{H_{i-1}}$ pour donner H_i à l'aide de la fonction de compression h :

$$\mathbf{H_i = h(H_{i-1}, M_i)}$$

La variable de chaînage initiale H_0 est fixée à la valeur d'**IV** et la dernière variable de chaînage H_k permet de déduire le haché final. En général, l'état interne de la fonction de hachage est de même taille n que le haché final. Dans ce cas, le haché est directement égal à H_k. Si l'état interne est plus grand que la taille de haché, on doit effectuer une certaine troncature pour obtenir la bonne taille de sortie. Il est aussi possible d'appliquer une fonction de sortie sur $H_k$avant d'obtenir le haché, mais cette étape est souvent omise en pratique. Le processus entier est décrit en **figure III.7**. Dans la suite, sauf mention contraire, nous considérons que la taille de l'état interne de la fonction de hachage est égale à celle du haché : n = N.

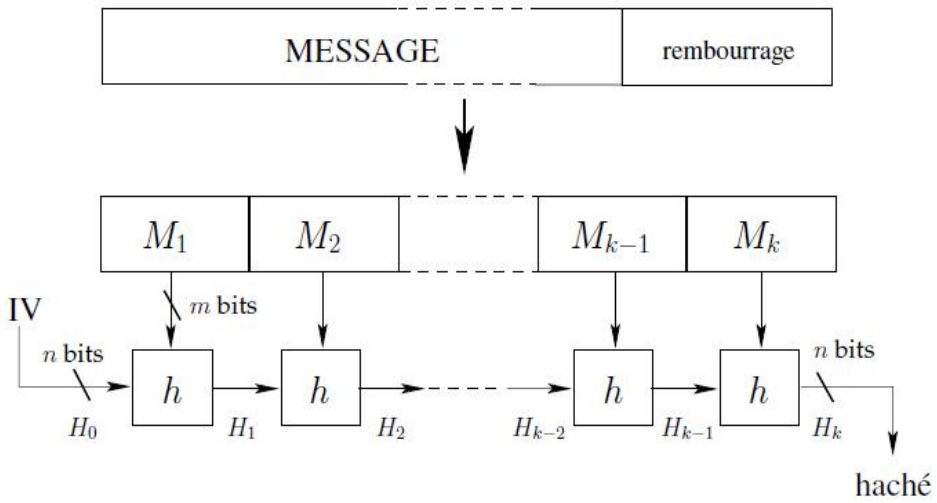

Figure III.7 : L'algorithme d'extension de domaine Merkle-Damgård[22]

Résumé de l'algorithme d'extension de domaine Merkle-Damgård

ENTRÉE : une fonction de compression $f : \{0,1\}^n \times \{0,1\}^r \longrightarrow \{0,1\}^n$ résistante aux collisions.
SORTIE : une fonction de hachage $h : \{0,1\}^* \longrightarrow \{0,1\}^n$ résistante aux collisions.

Découper un message x de taille b bits en blocs $x_1, x_2, \ldots, x_t$ de taille r bits en ajoutant des bits 0 au bloc x_t si nécessaire, et définir un bloc supplémentaire x_{t+1}, destiné à contenir le codage de la longueur du message.

$H_0 = 0^n$ $\{n$ *bits à* $0\}$
$i = 1$
while $i \leq t+1$ **do**
  $H_i = f(H_{i-1} \parallel x_i)$
end while
L'empreinte du message x est alors $h(x) = H_{t+1} = f(H_t \parallel x_{t+1})$.

Les variables H_i sont appelées variables de chaînage, et la variable H_0 est qualifiée de valeur initiale (*Initial Value*).

II.7. Conclusion

Dans cette partie nous avons présenté le principe de fonctionnement des fonctions de hachage et puis dans la partie qui suit nous nous intéressons à l'algorithme de la fonction de hachage SHA-1.

Partie 3 La fonction de hachage SHA-1

III.1. Introduction

Nous présentons dans cette partie la fonction de hachage de la famille Secure Hash Algorithm(SHA-1), la plus implantée en pratique et la plus étudiée car notre projet est basé sur cette fonction qui permet de calculer une représentation condensée d'un message ou d'un fichier de données.

III.2. Historique

Le SHA-1 est le successeur du SHA-0 qui a été rapidement mis de côté par le NIST (national institute standards and technology) pour des raisons de sécurité insuffisante. Le SHA-0 était légitimement soupçonné de contenir des failles qui permettraient d'aboutir rapidement à des collisions (deux documents différents qui génèrent le même *condensat*). Face à la controverse soulevée par le fonctionnement du SHA-0 et certains constats que l'on attribue à la NSA (National Security Agency), le SHA-0 s'est vu modifié peu après sa sortie (1993) et complexifié pour devenir le SHA-1 (1995). Une collision complète sur le SHA-0 a été découverte par Antoine Joux *et al.*En août 2004 et laisse penser que le SHA-1 pourrait lui aussi subir une attaque.**[23]**

La modification de message et lapartie non linéaire aboutirent en fait à la première attaque théorique contre SHA-1. Les méthodes sont identiques à celles utilisées pour SHA-0, excepté la recherchede vecteurs de perturbation. En effet, la rotation dans l'expansion de message changecomplètement la situation et il a fallu créer une nouvelle heuristique pour trouver de bonscandidats.**[22]**

III.3. Fonctionnement de la fonction SHA-1 [23]

Le SHA-1 prend un message d'un maximum de 2^{64} bits en entrée et produit une empreinte de 160 bits.

Le SHA-1 commence par ajouter à la fin du message un bit à 1 suivi d'une série de bits à 0, puis la longueur du message initial (en bits) codée sur 64 bits. La série de 0 a une longueur telle que la séquence ainsi prolongée a une longueur multiple de 512 bits. L'algorithme travaille ensuite successivement sur des blocs de

512 bits.Pour chaque bloc l'algorithme calcule 80 tours (ou rondes, « *rounds* » en anglais) successifs et applique une série de transformations sur l'entrée.

La première étape consiste à calculer 80 valeurs sur 32 bits. Les 16 premières valeurs sont obtenues directement à partir du bloc « message » en entrée. Les 64 autres sont calculées successivement.Le SHA-1 les obtient grâce à une rotation (absente dans SHA-0) qui est appliquée sur le résultat d'un XOR, il utilise pour cela 4 mots obtenus dans les itérations précédentes.

On définit ensuite 5 variables qui sont initialisées avec des constantes (spécifiées par le standard), le SHA-1 utilise encore 4 autres constantes dans ses calculs. Si un bloc de 512 bits a déjà été calculé auparavant, les variables sont initialisées avec les valeurs obtenues à la fin du calcul sur le bloc précédent.Il s'ensuit 80 tours qui alternent des rotations, des additions entre les variables et les constantes. Selon le numéro du tour, le SHA-1 utilise une des quatre fonctions booléennes. L'une de ces fonctions est appliquée sur 3 des 5 variables disponibles. Les variables sont mises à jour pour le tour suivant grâce à des permutations et une rotation. En résumé, le SHA-1 change sa méthode de calcul tous les 20 tours et utilise les sorties des tours précédents.

À la fin des 80 tours, on additionne le résultat avec le vecteur initial. Lorsque tous les blocs ont été traités, les cinq variables concaténées (5 × 32 = 160 bits) représentent la signature.

III.3. Fonction de compression

Chaque ronde comprend 20 étapes qui manipulent ainsi les 5 registres : (A,B,C,D,E) ß (E + f(t,B,C,D) + (A << 5) +W_t + K_t), A, (B << 30),C,D)

où A,B,C,D,E se rapportent aux 5 registres, t est le numéro de l'itération, f(t, B, C, D) est une fonctionprimitive non-linéaire de la ronde pour l'itération t, W_t est dérivé du bloc de message(32 bits) et K_t estune valeur additive.

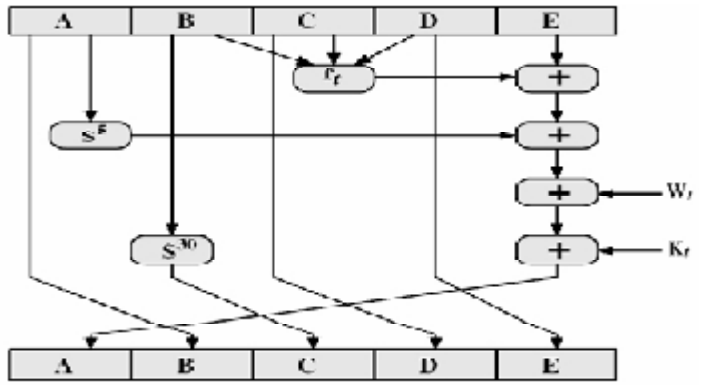

Figure III.8 : Fonction de ronde du SHA-1[19]

Le calcul des W_t, dérivés des blocs de messages est donné à la **Figure III.9**. Pour ce calcul, il faut remarquer qu'il introduit beaucoup de redondance, ce qui introduit une forte interdépendance des blocs.

En conséquence, il est donc relativement difficile de trouver deux messages avec le même haché. La formule résumant le calcul de $W_t$est donnée comme suit.

- Fonctions primitives
  - Travaille sur 3*32 bits et fournit un résultat sur 32 bits
  - $0 \leq t \leq 19$ $f_1=f(t,B,C,D)$ $(B \wedge C) \vee (\neg B \wedge D)$
  - $20 \leq t \leq 39$ $f_2=f(t,B,C,D)$ $B \oplus C \oplus D$
  - $40 \leq t \leq 59$ $f_3=f(t,B,C,D)$ $(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
  - $60 \leq t \leq 79$ $f_4=f(t,B,C,D)$ $B \oplus C \oplus D$
- W_t
  - $0 \leq t \leq 15$: les 16 premières valeurs du bloc
  - $16 \leq t$: $W_t = ((W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \ominus W_{t-3}) <<< 1)$

v Operations de mots logique

- **^**= opération binaire AND
- **v**= opération binaire OR
- $\oplus$= opération binaire XOR
- ┐= complément binaire
- **+**= addition modulo 2^w
- **<<**= décalage binaire à gauche, où**x << n** s'obtient en supprimant les n bits de gauche de x et ajoutant n zéros à droite.
- **>>**= décalage binaire à droite, où **x >> n**s'obtient en supprimant les n bits de droite de x et ajoutant n zéros à gauche.

- **Constantes**

SHA-1 utilise quatre valeurs réparties dans les 80 constantes $K_0, K_1, \ldots, K_{79}$ d'après la règle

$$K_t = \begin{cases} 0x5a827999, & \text{si } 0 \leq t \leq 19 \\ 0x6ed9eba1, & \text{si } 20 \leq t \leq 39 \\ 0x8f1bbcdc, & \text{si } 40 \leq t \leq 59 \\ 0xca62c1d6, & \text{si } 60 \leq t \leq 79 \end{cases}$$

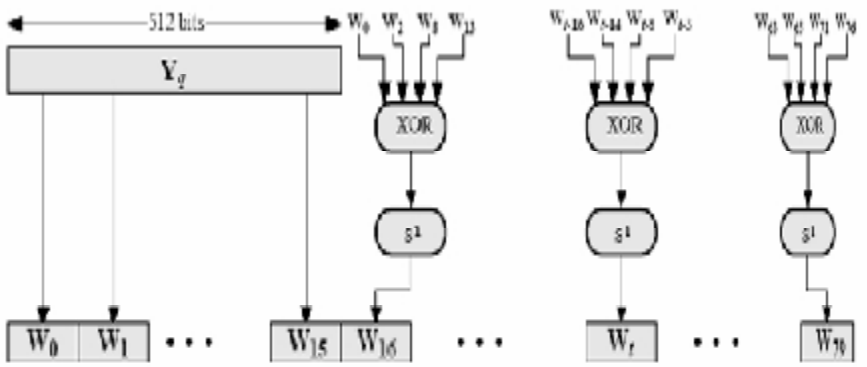

Figure III.9 : Calcul du W_t dérivé [19]

III.4. Comparatif MD5 - SHA-1

L'attaque par force brute est plus difficile (160 contre 128 bits pour MD5). Mais il est un peu plus lent que MD5 (80 contre 64 itérations). Il fut conçu comme simple et compact, et optimisé pour les processeurs big-endian.

La comparaison entre MD5 et SHA-1 se résume dans le tableau donné à la **figure III.10** suivante **:**

	MD5	SHA-1
Haché	128	160
Taille bloc de base	512	512
Nombres d'étapes	64 (4*16)	80 (4*20)
Taille du msg max.	inf.	$2^{64}-1$
Nb fcts primitives	4	4
Nb const. Add.	64	4
Type d'architecture	Little-endian	Big-endian

Figure III.10 : Comparatif MD5 - SHA-1

III.5. Exemple

Considérons le message «abc» qui correspond à la séquence de bits suivante : 01100001011000100110001 1. La longueur du message est donc 24.

1. On ajoute des bits de façon à obtenir une séquence de 512 [24/512] — 64 = 448 bits. On ajoute donc la série de bits commençant par le bit 1 suivi de 423 bits 0. Ensuite, on ajoute à cette séquence la représentation binaire du nombre 24 (écrit 64 bits avec 64 bits), soit 00 • • • 0 11000, pour obtenir un message de 512 bits.
2. Il faut ensuite diviser la séquence de bits en blocs de 512 bits. Dans cet exemple, il y a un seul bloc : m = (mi).
3. On pose H_0 = 67452301, H_1= EFCDABS9, H_2 = 98BADCFE, H_3 = 10325476 et H_4 = C3D2E1F0.
4. (a) On écrit $m_1 = (W_0, W_1, W_2, ..., W_{15})$, où W_0 = 61626380, W_1= 00000000, W_2 = 00000000, W_3 = 00000000, W_4= 00000000, W_5 = 00000000, W_6 = 00000000, W_7 = 00000000, W_8 = 00000000, W_9 = 00000000, W_{10} = 00000000, W_{11} = 00000000, W_{12} = 00000000, W_{13} = 00000000, W_{14} = 00000000 et W_{15} = 00000018 sont écrits en hexadécimal.

(b) Il faut ensuite trouver W_t pour 15< t < 80 :

$W_{16} = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16})$ 1

$= (W_{13} \oplus W_8 \oplus W_2 \oplus W0) <<1$

$= 00000000 \oplus 00000000 \oplus 00000000 \oplus 61626380$ 1

$= 01100001011000100110001110000000$ 1

$= C2C4C700.$

$W_{17} = (w_{14} \oplus w_9 \oplus w_3 \oplus w_1)$ 1

$= 00000000 \oplus 00000000 \oplus 00000000 \oplus 00000000$ 1

$= 00000000$

$W_{18} = (W_{15} \oplus W_{10} \oplus W4 \oplus W_2)$ 1

$= 00000018 \oplus 00000000 \oplus 00000000 \oplus 00000000$ 1

$= 00000000000000000000000000011000$ 1

$= 00000030$

$W_{l9} = (W_{16} \oplus W_{11} \oplus W_5 \oplus W_3)$ 1

$= C2C4C700 \oplus 00000000 \oplus 00000000 \oplus 00000000$ 1

$= 11000010110001001100011100000000$ 1

$=85898E01$

$W_{20} = (W17 \oplus W12 \oplus W_6 \oplus W_4)$ 1

$= 00000000 \oplus 00000000 \oplus 00000000 \oplus 00000000$ 1

$= 00000000$

$W_{2l} = (W_{18} \oplus W_{13} \oplus W_7 \oplus W_5)$ 1

$= 00000030 \oplus 00000000 \oplus 00000000 \oplus 00000000$ 1

$= 00000000000000000000000000110000$ 1

$= 00000060$

$W_{22} = (W_{19} \oplus W_{14} \oplus W_6 \oplus W_6)$ 1

= 85898E01 $\oplus$ 00000000 $\oplus$ 00000000 $\oplus$ 00000000 1

= 10000101100010011000111000000001 1

= 0B131C03

(c) Après avoir trouvé ainsi W16,..., W79, on pose

a = H_0 = 67452301, b = H_1 = EFCDAB89, c = H_2 = 98BADCFE, d = H_3 = 10325476

et e = H_4 = C3D2E1F0.

(d) Pour t = 0 :

T = (67452301 << 5) + C3D2E1F0 + 61626380 + 5A827999
+f_0 (EFCDAB89,98BADCFE, 10325476),
= E8A4602C
+C3D2E1F0
+61626380
+5.4827999
+98BADCF8
= 0116FC2D,
e = 10325476,
d = 98BADCFE,
c = 7BF36AE2,
b = 67452301,
a = 0116FC2D.

Pour connaître les valeurs obtenues pour t = 1,..., 79 voir [24].

Le message haché obtenu, en hexadécimal, est

,49993E364706816AEA3£25717850C26C9CD0D89D.

III.6. Conclusion

Ce chapitre est partagé en trois parties dans la première partie nous avons présenté les différents types de la cryptographie ainsi que leurs inconvénients puis dans la deuxième partie de chapitre on a décrit les fonctions de hachage cryptographiques ou ce trouve la fonction de hachage SHA-1 qui nous intéresse beaucoup plus, et on a terminé ce chapitre par décrire le fonctionnement de cette fonction.

Le chapitre suivant sera consacrée à l'analyse et la conception de notre application.

I. Introduction

Apres avoir vu, dans les chapitres précédents les différents concepts nécessaires à l'accomplissement de notre travail, nous passons maintenant à la partie conception.

Aujourd'hui, le développement d'un logiciel s'appuie sur une prospective orientée objet. Tout simplement parce qu'elle a démontré son efficacité lors de la construction de systèmes dans les domaines les plus divers et qu'elle englobe toutes les dimensions de tous les degrés de complexité.

Dans ce qui suit on va vous présentée l'UML (**U**nified **M**odeling **L**anguage) pour la modélisation orienté objet car il est le reflet du futur système avant même sa réalisation, dans le but d'avoir une meilleure analyse et de rendre la conception de notre application plus complète et tous cela grâce au diagramme qu'il offre.

II. But et contexte de la plate-forme

Le système doit offrir une IHM (L'interface homme –machine) simple et facile à utiliser.

La fenêtre principale permet à l'utilisateur d'avoir l'empreintes de tous type de fichier (par exemple Word, PDF, image, …etc.) comme il peut aussi avoir l'empreinte de ce dernier à partir de sans adresse physique.

III. La modélisation UML

UML offre un moyen astucieux permettant de représenter diverses projections d'un même système informatique. Plusieurs diagrammes UML existent, et pour modéliser notre application nous allons présenter les diagrammes suivant :

- Diagramme de cas d'utilisation
- Diagramme de séquence
- Diagramme de classes
- Diagramme d'activité

IV. Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation est un graphe d'acteur, un ensemble de cas d'utilisation réunis par la limite de système, des associations de communication entre les acteurs et les cas d'utilisation, et des généralisations entre cas d'utilisation.

Dans notre cas on a un seul acteur qu'est l'utilisateur, la figure suivante représente son diagramme d'utilisation.

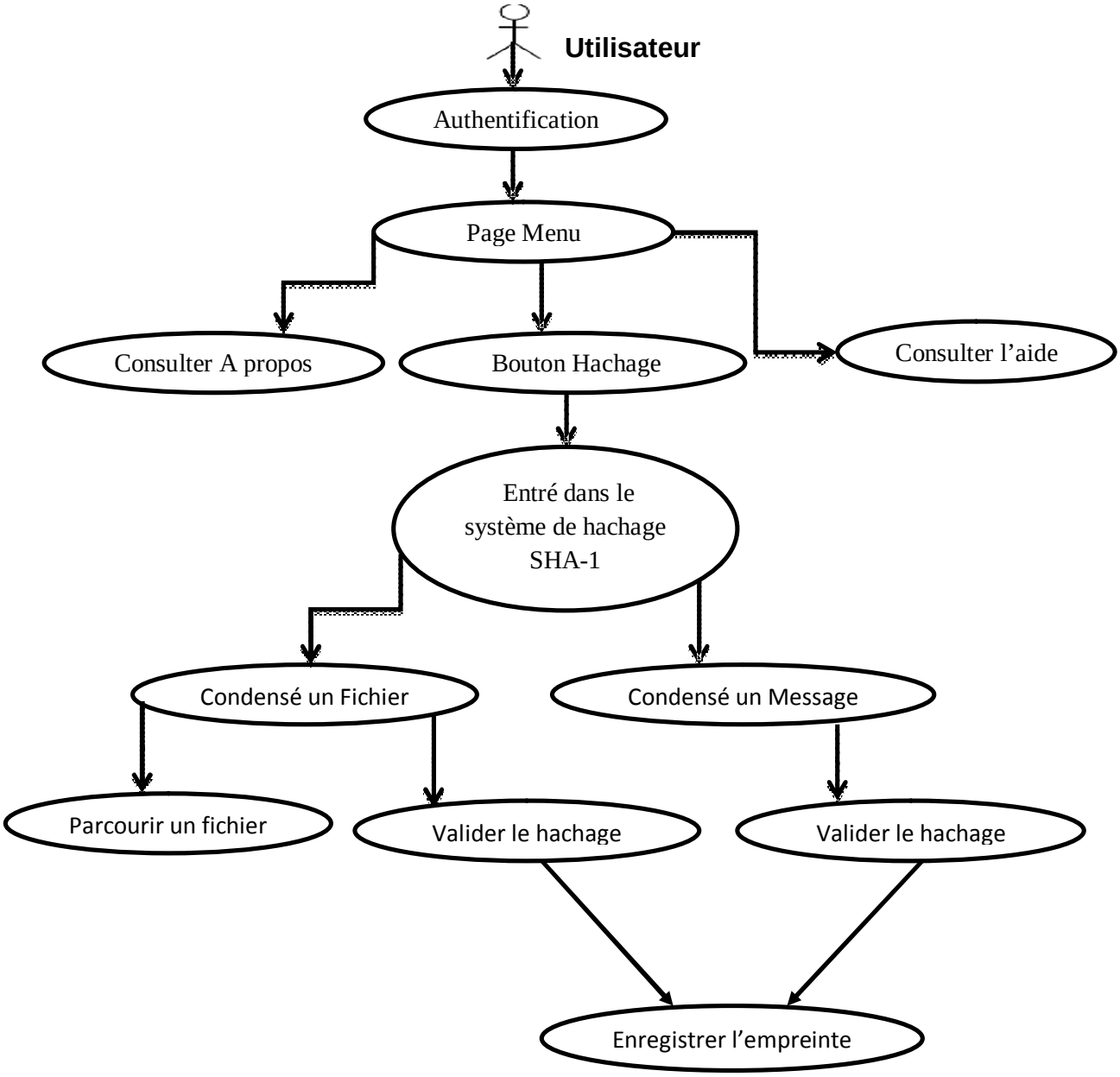

Figure IV.1 : Diagramme de cas d'utilisation

V. Elaboration des Diagrammes de séquences et diagrammes d'activités

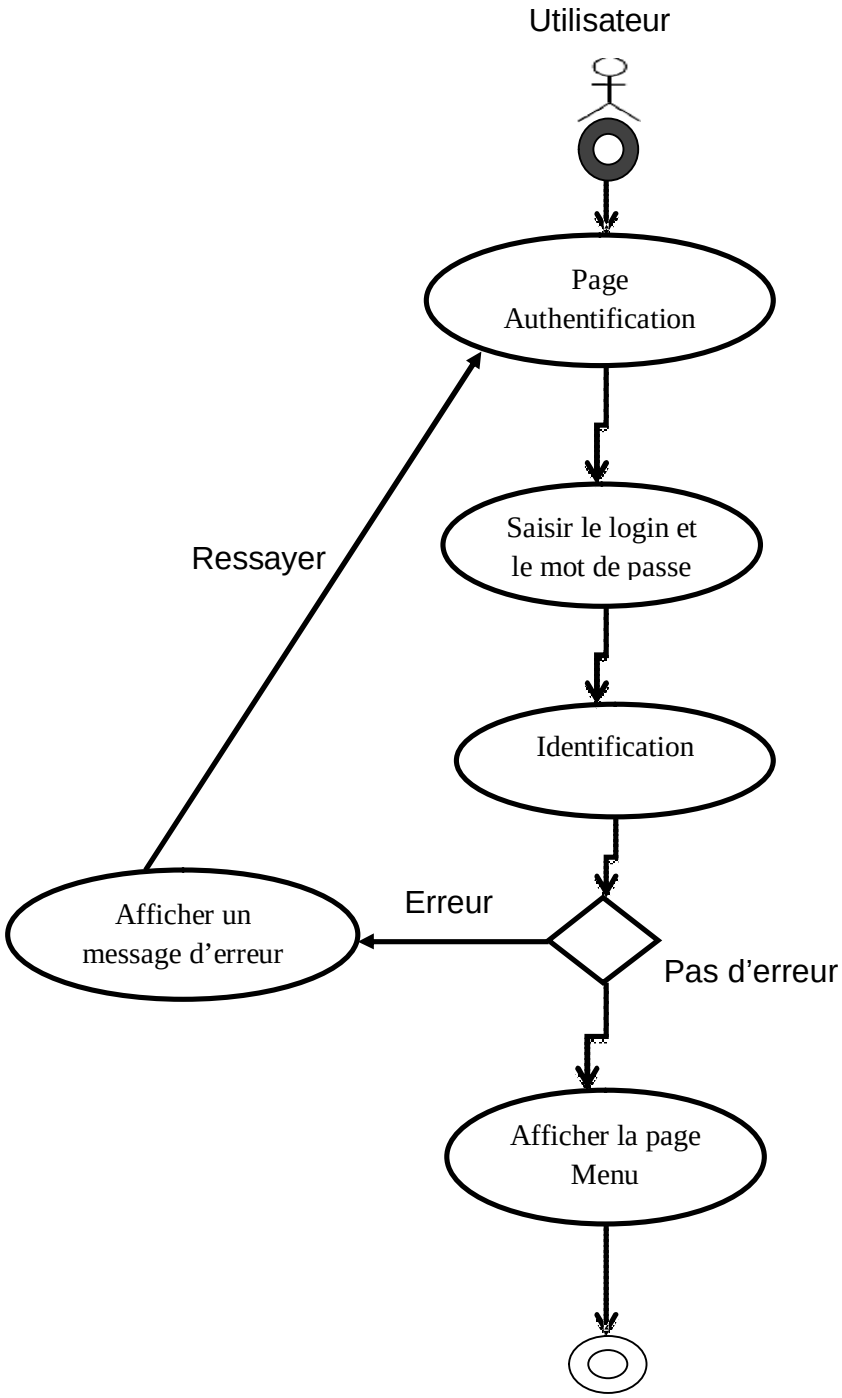

Figure IV.2 : Diagramme d'activité du cas d'utilisation << s'authentifier >>

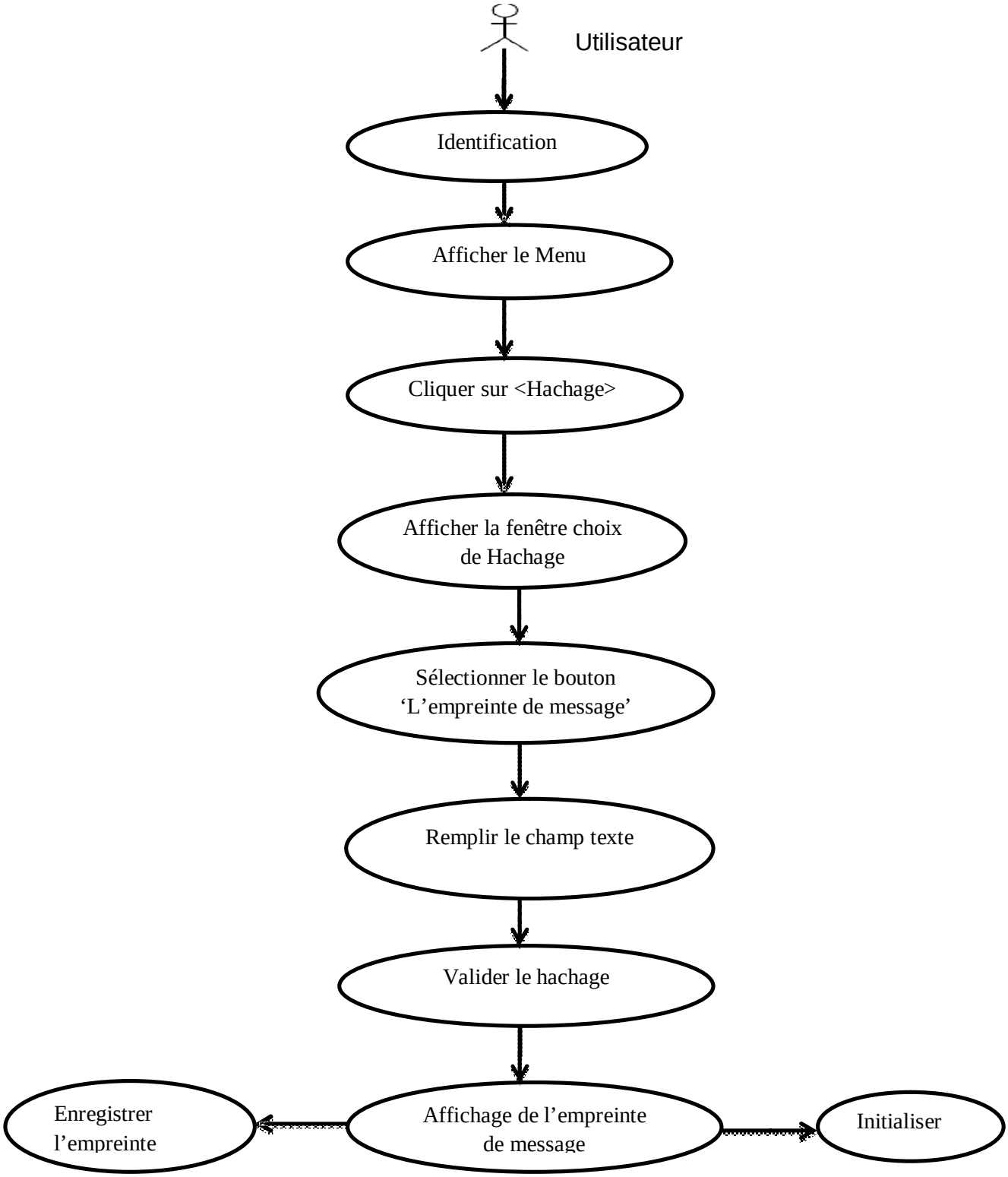

Figure IV.3 : Diagramme d'activité du cas d'utilisation << L'empreinte de message par saisi>>

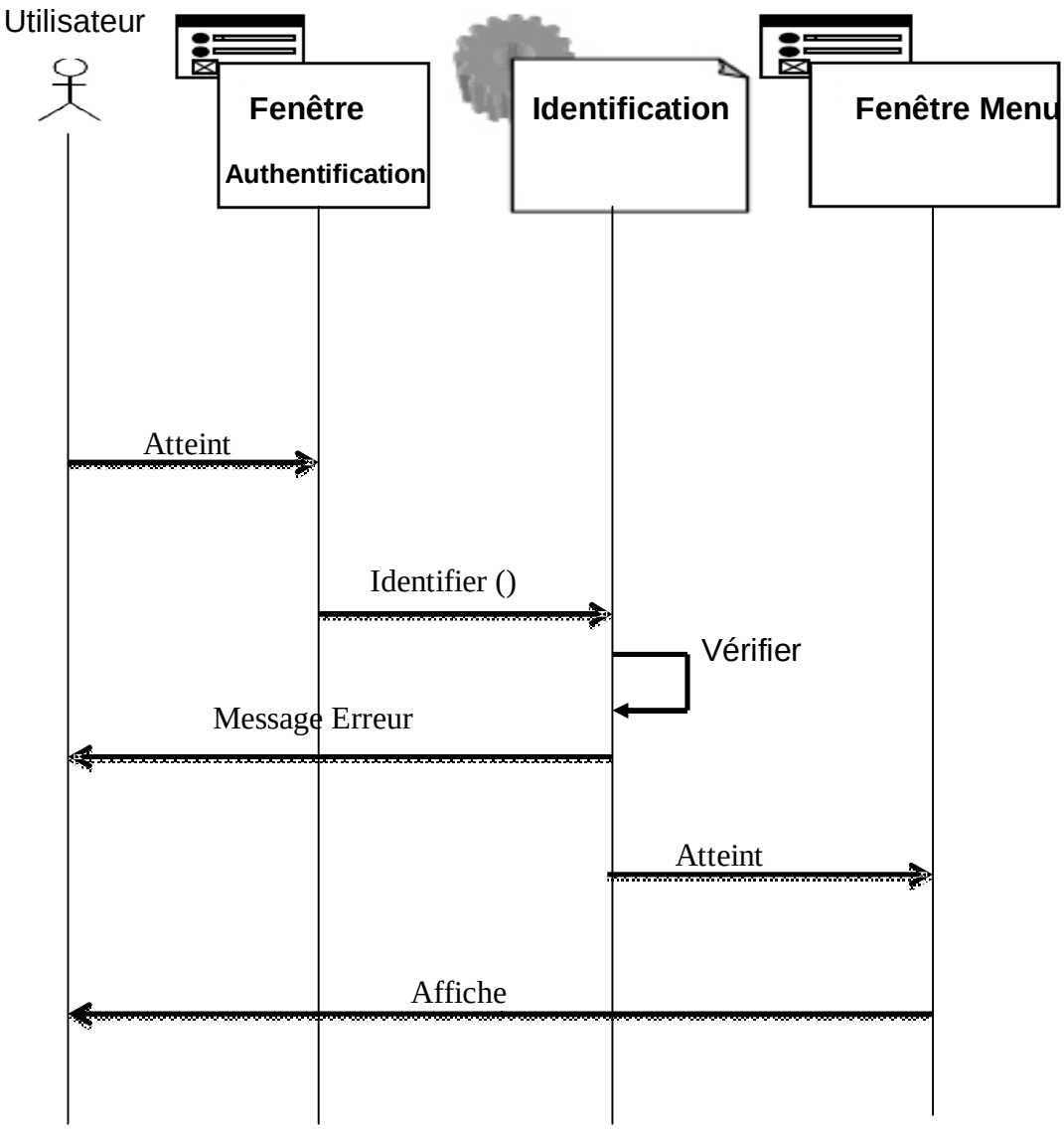

Figure IV.4 : Diagramme de séquence détaillé du cas d'utilisation < S'authentifier>

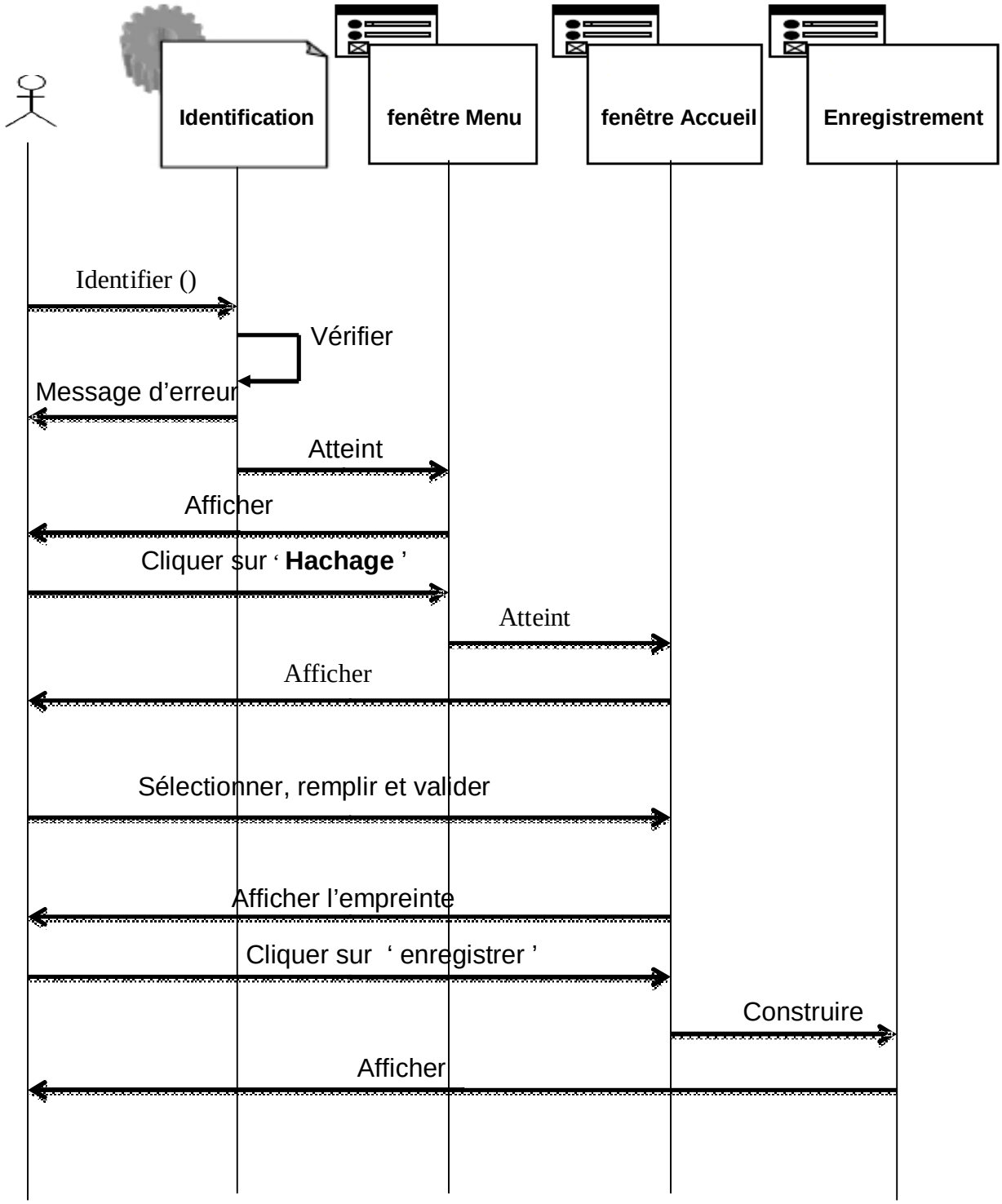

Figure IV.5 : Diagramme de séquence détaillé du cas d'utilisation << L'empreinte de texte par saisi >>

VI. Elaboration des Diagrammes de classes

Un diagramme de classe est une collection d'éléments de modèles (statiques), tels que des classes ou des interfaces et leurs relations, connectés entre eux comme un graphe.

En effet un diagramme de classe montre uniquement des aspects statiques du modèle et fait abstraction des aspects dynamiques ou temporels, mêmes si les éléments du diagramme de classe peuvent avoir un comportement dynamique important.

Parmi Les associations utilisées dans ces diagrammes on trouve :

- **Submit :** Une association de soumission se trouve toujours entre un formulaire et une page serveur. Les valeurs des champs du formulaire sont soumises au serveur qu'il les traite, par l'intermédiaire de pages serveur**.**
- **Build :** Est une association orientée entre les pages client et les pages serveur. Elle indique quelle page serveur est responsable de la création de la page client. Une page serveur peut construire plusieurs pages client, mais une page client n'est construite que par une et une seule page serveur.
- **Redirect :** Est une association unidirectionnelle qui relie deux pages client ou serveur.
- **Link :** C'est une association entre une page client et une autre page client ou serveur.

Dans ce qui suit nous allons présenter quelques diagrammes de classes de quelques cas d'utilisations :

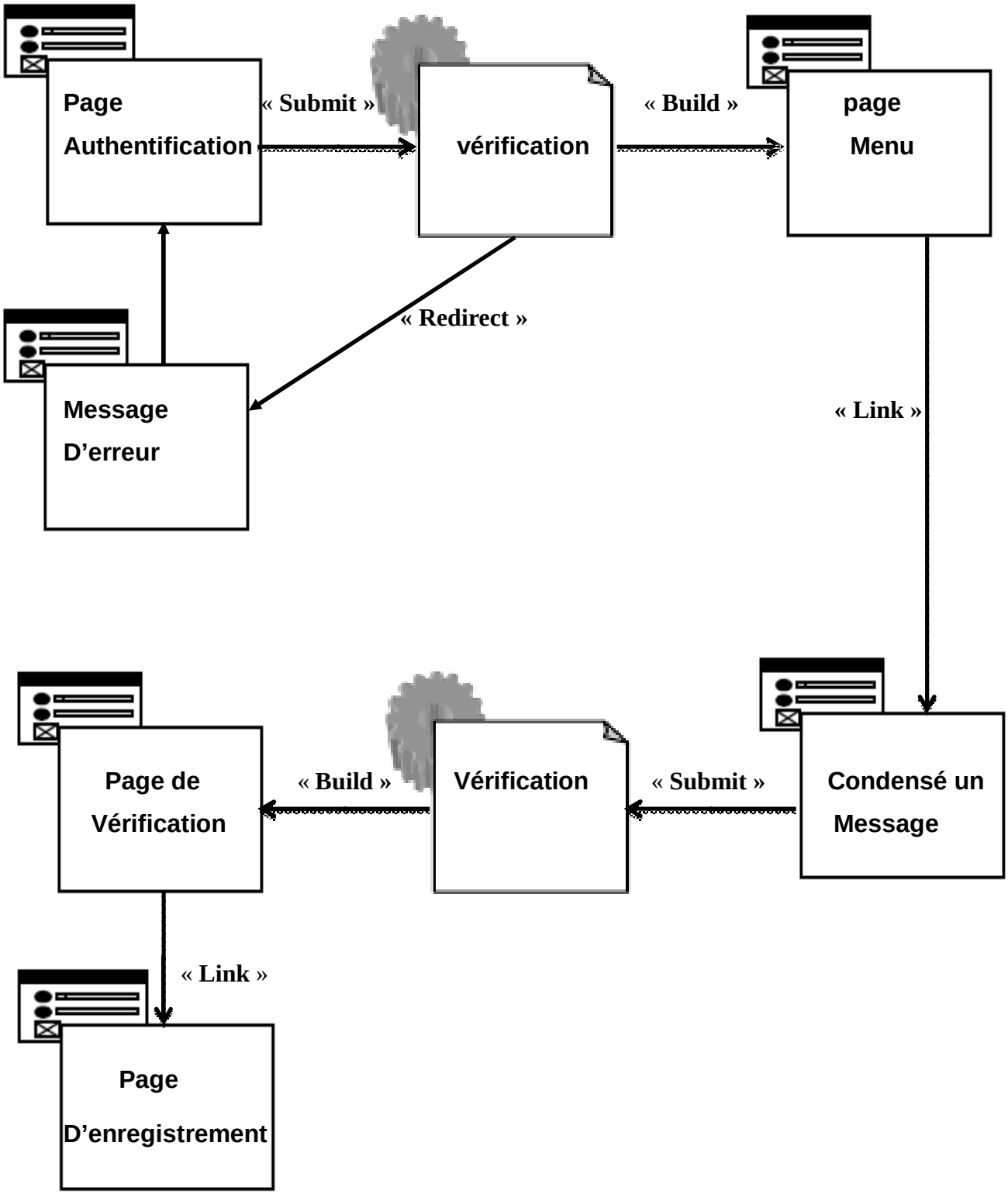

Figure IV.6 : Diagramme de classes « Condensé un message »

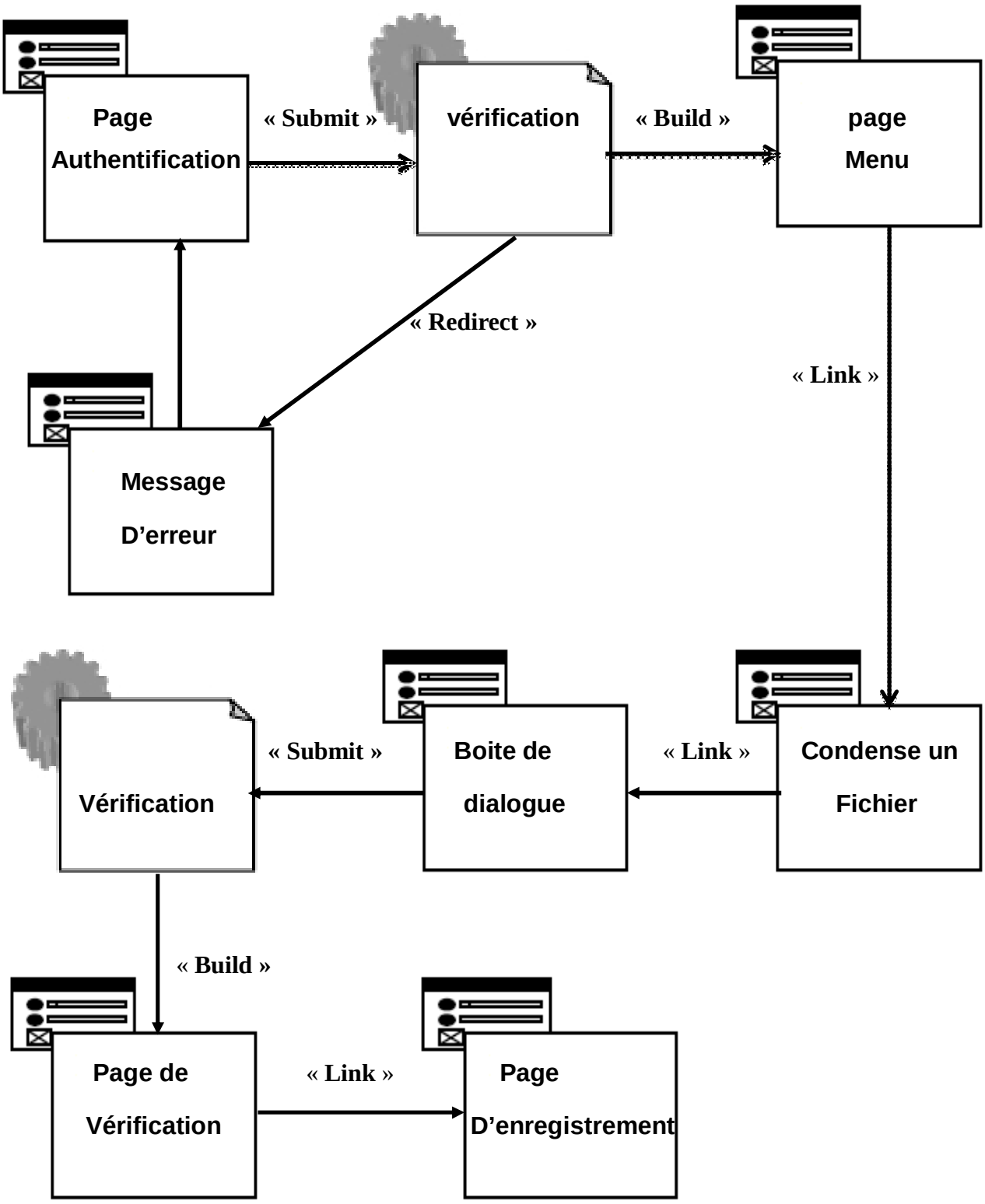

Figure IV.7 : Diagramme de classes « Condensé un fichier »

VII. Conclusion

Dans ce chapitre, nous avons présenté la conception de l'application en nous appuyant sur une démarche de modélisation basée sur UML en donnons quelques diagrammes de cas d'utilisation (Diagrammes de séquence, Diagrammes d'activités, Diagrammes de classes).

Dans le dernier chapitre, qui va suivre, seront définie les outils et les langages de programmation qui ont servies à mettre en œuvre l'application ainsi que la présentation de certaines interfaces.

I. Introduction

Après avoir présenté dans le chapitre précédent l'Analyse et la Conception de notre système, nous allons, tout d'abord, 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.

II. Environnement de développement

II.1. Matériel utilisé

Durant la réalisation de notre application, nous avons utilisé une machine ayant les caractéristiques suivantes :

- Un microprocesseur Intel(R) Core(TM)2 Duo.
- Fréquence d'horloge 2.2GHz.
- RAM DDR de 3 GO.
- Disque dure de 250 GO.

Nous avons développé notre application sous un système d'exploitation Microsoft Windows 7.

II.2. Présentation de l'environnement

II.2.1. 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 indépendant de toute plate-forme :** Une application en java fonctionne sur n'importe quel environnement (Unix, Windows, …) disposant d'une JVM (Java Virtual Machine), en français : la machine virtuelle java.

- **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 développé 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.

II.2.2. Présentation de NetBeans

III. Présentation du logiciel

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 7.0.1 (NetBeans 7.0.1).

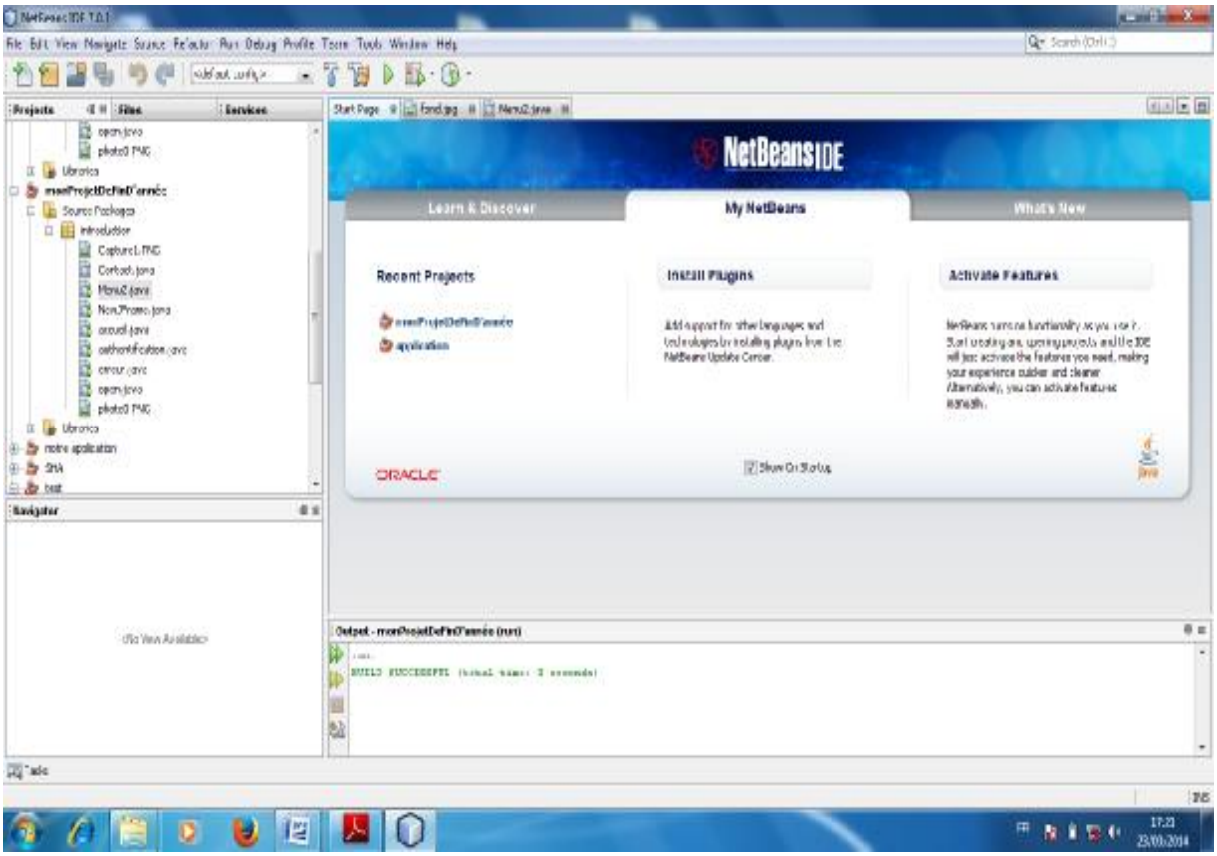

Figure V.1: Interface d'environnement de développement NetBeans

IV. Présentation de l'application

Ce logiciel permet de convertir des textes saisis ou des fichiers sélectionnés par l'utilisateur et avoir leurs empreintes grâce au système de hachage SHA-1 car il est très utile pour la sécurisation des données.

Ø L'empreinte d'un texte saisie

Permet à l'utilisateur de saisir un texte et avoir a la fin leur empreinte.

Ø L'empreinte d'un fichier sélectionné

Permet à l'utilisateur de sélectionner un fichier et avoir a la fin leur empreinte.

V. Présentation des interfaces de notre application

Nous allons maintenant présenter les différentes interfaces du logiciel ainsi que son fonctionnement.

V.1. Interface d'authentification

Le but de cette interface est d'authentifier l'utilisateur du logiciel pour offrir une certaine privatisation de l'utilisation de ce logiciel, et ainsi empêcher toute tentative d'utilisation illégale des personnes ne possédant pas le droit de l'utiliser.

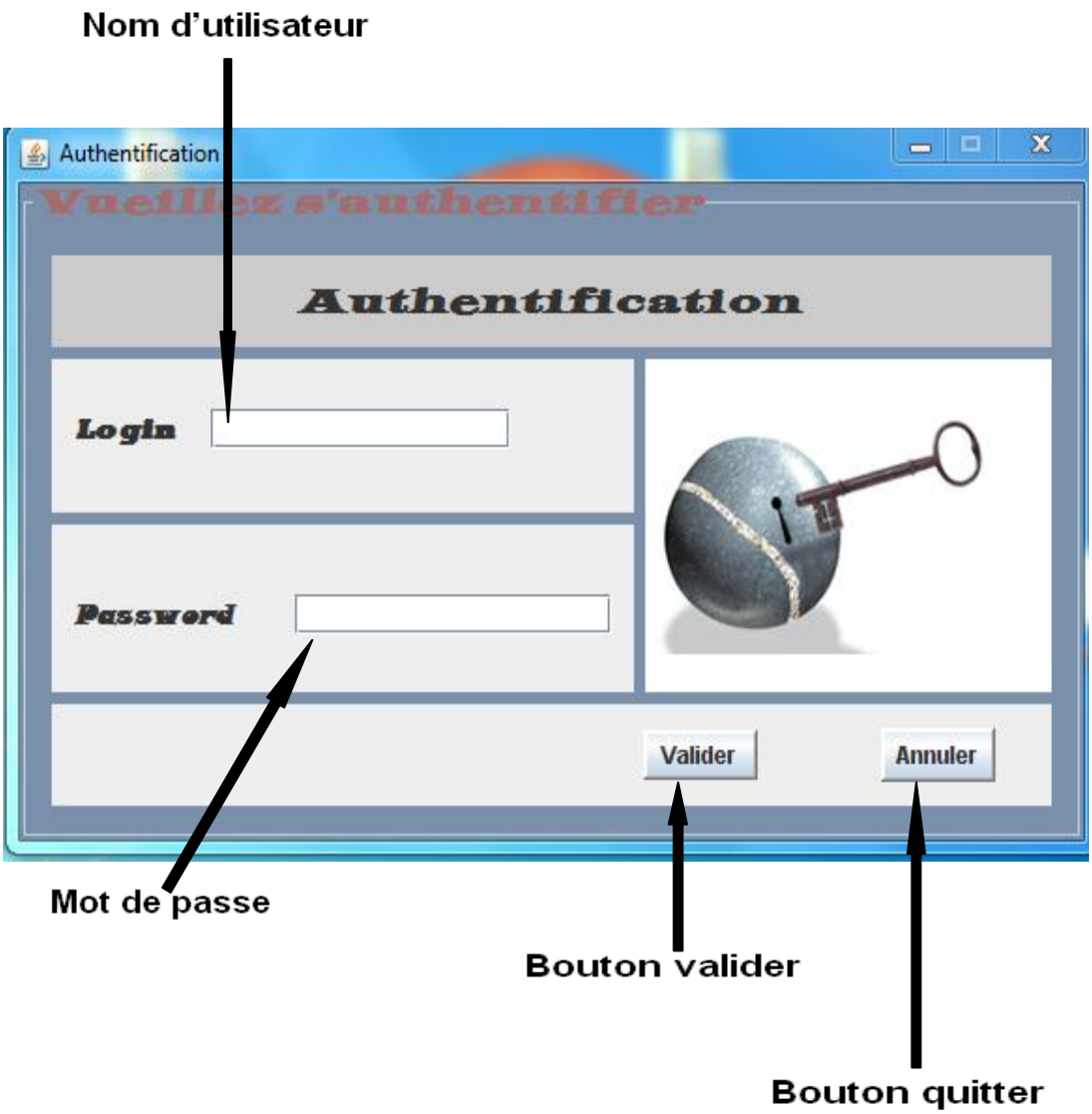

Figure V.2 : Interface d'authentification.

V.2. Interface Menu

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

- Le bouton «hachage» : permet à l'utilisateur de lancer une opération de hachage.
- Le bouton «À propos» : permet le lancement de la fenêtre à propos.
- Le bouton «Quitter» : pour quitter l'application.

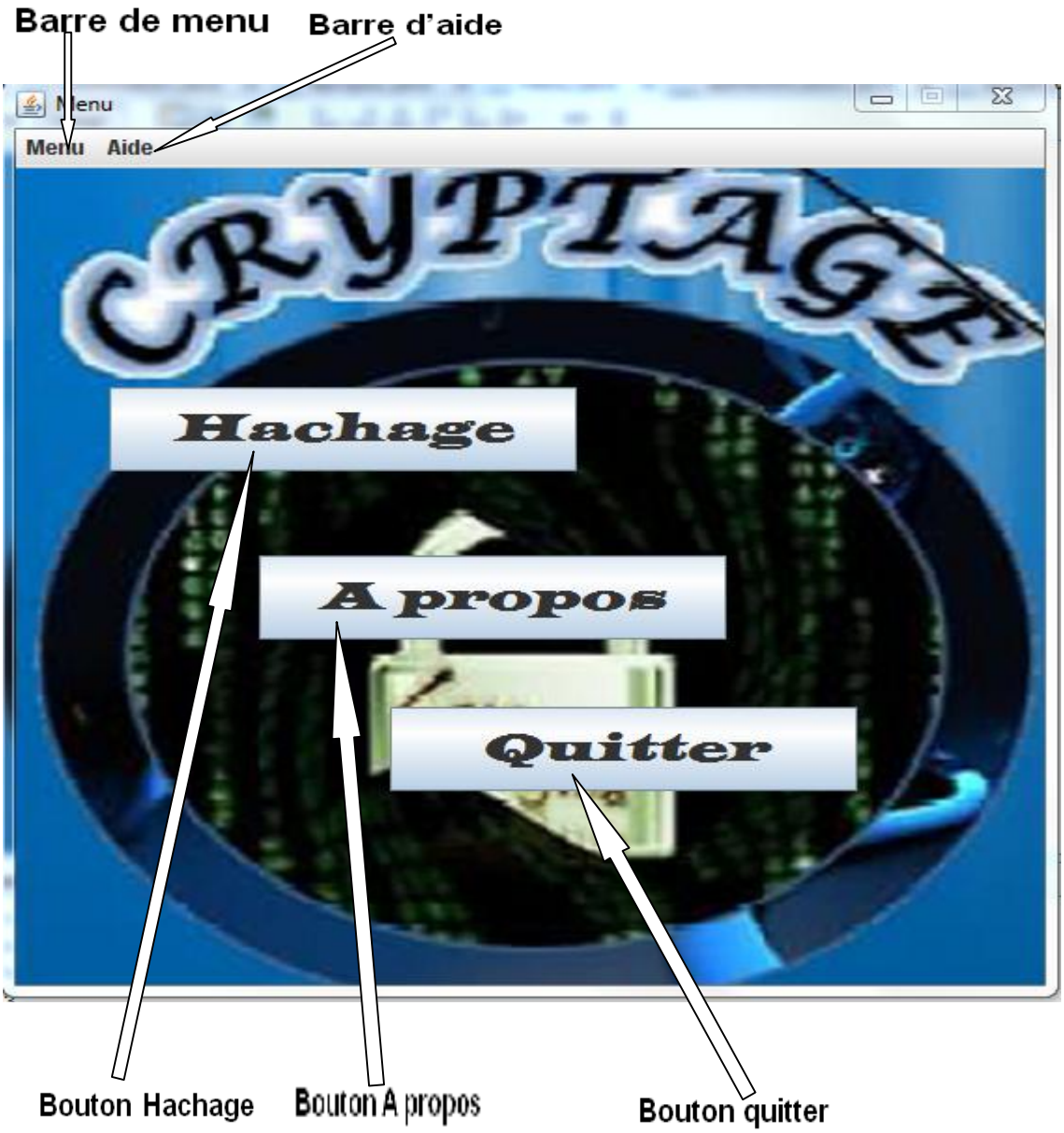

Figure V.3 : Interface Menu

V.3. Interface principale

Dans cette interface, l'utilisateur choisit le type de cryptage qui lui convient, notre application offre deux types qui sont :

- L'empreinte de fichiers par sélection.
- L'empreinte de message.
- Pour calculer l'empreinte d'un fichier, l'utilisateur doit sélectionner le bouton «L'empreinte de fichier par sélection» et l'autre partie de la page (L'empreinte de message) sera inaccessible, puis il clique sur le bouton «Valider».

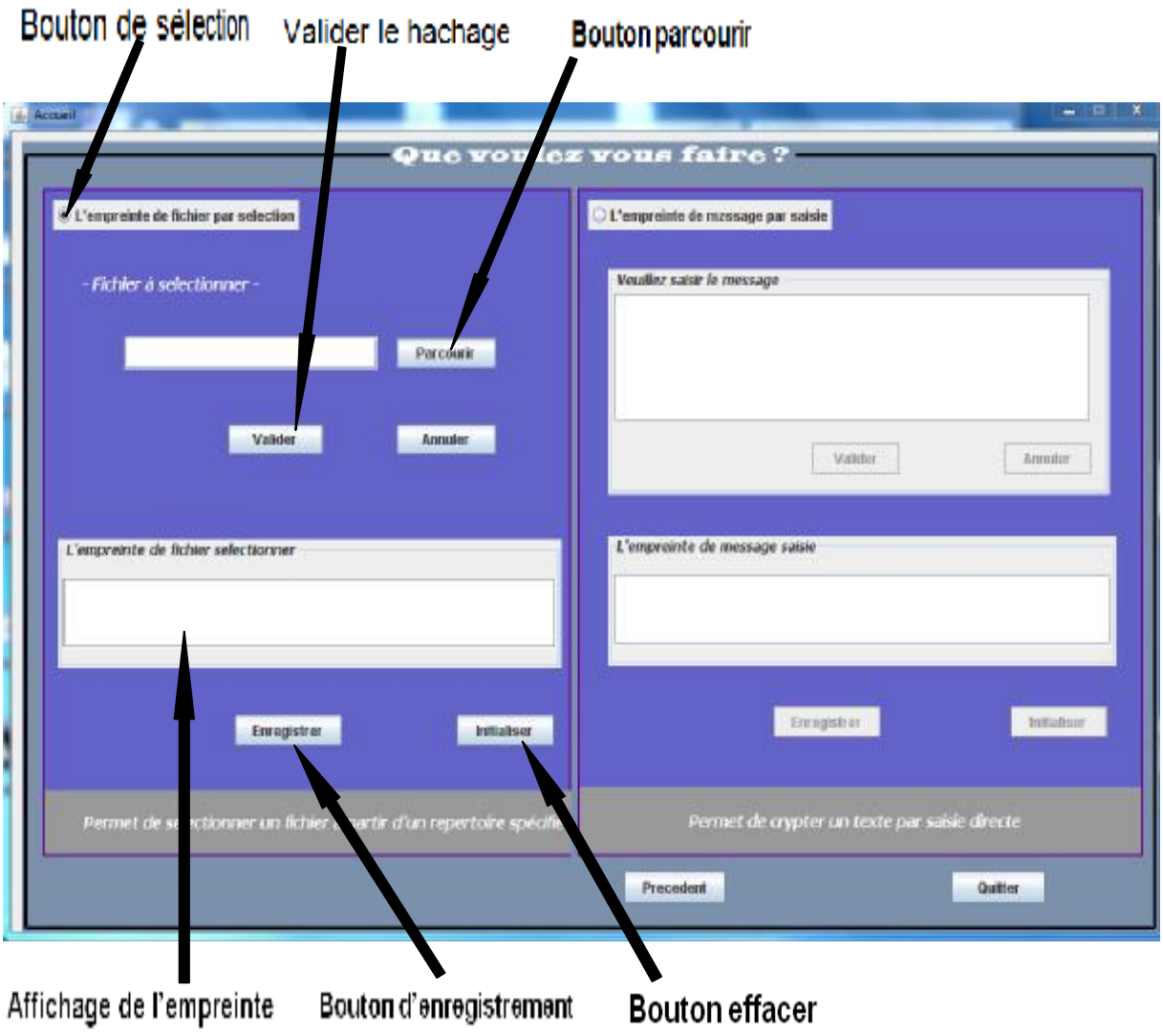

Figure V.4 : Interface Accueil (hachage de fichier par sélection).

Si l'utilisateur veut avoir l'empreinte d'un texte (message) en le saisissant, il n'a qu'à sélectionner le bouton «L'empreinte de texte par saisie», puis il clique sur «Valider».

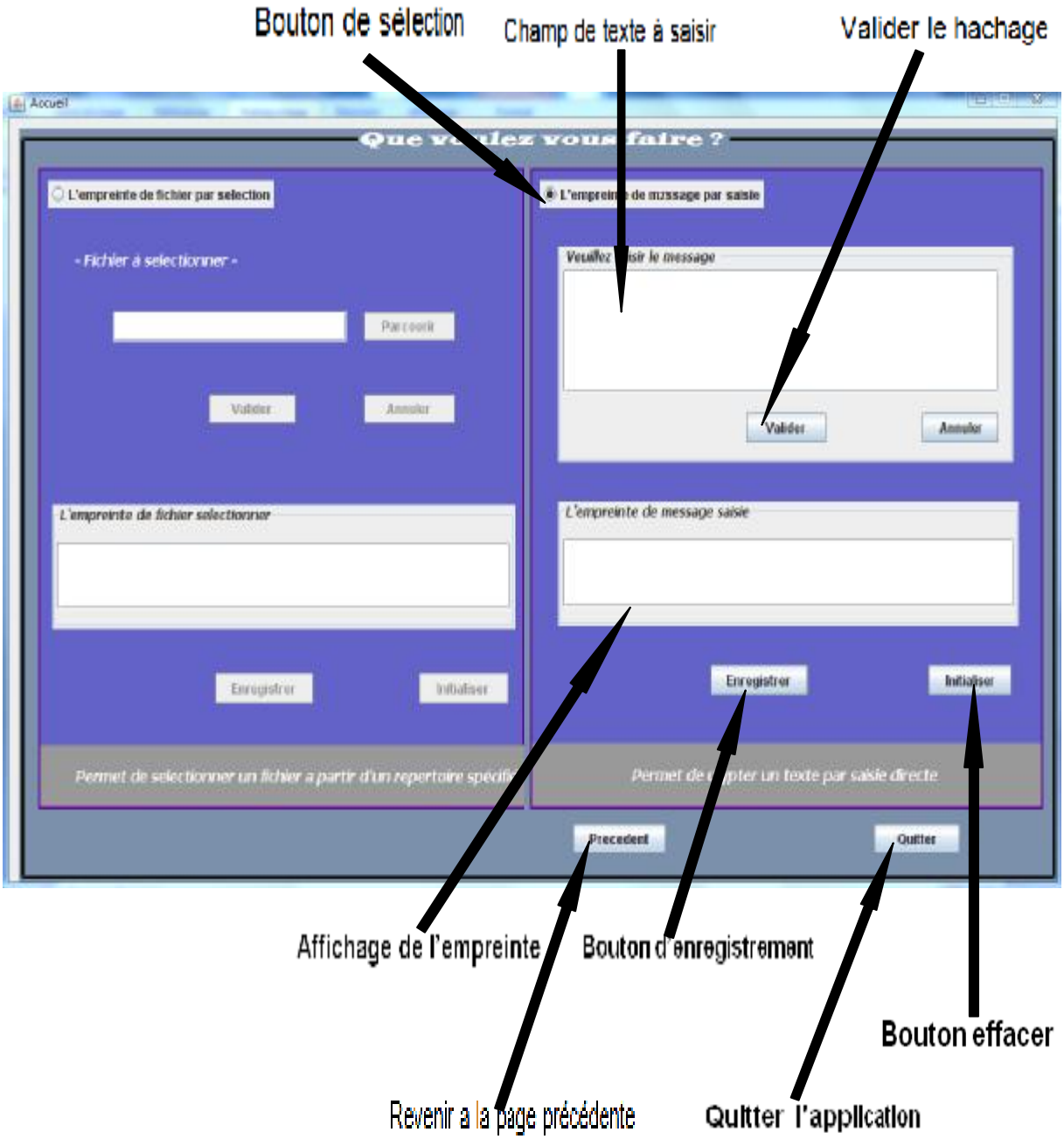

Figure V.5 : Interface Accueil (L'empreinte de message par saisi).

VI. Exemples d'utilisation (Cas de L'empreinte de fichier par sélection)

Afin d'évaluer la performance de notre logiciel, nous vous proposons un exemple d'utilisation :

- ü L'empreinte d'un fichier

v L'empreinte de fichier par sélection

Pour dérouler cet exemple, nous allons utiliser le fichier «analyse et coception.doc » dont le contenu est le suivant :

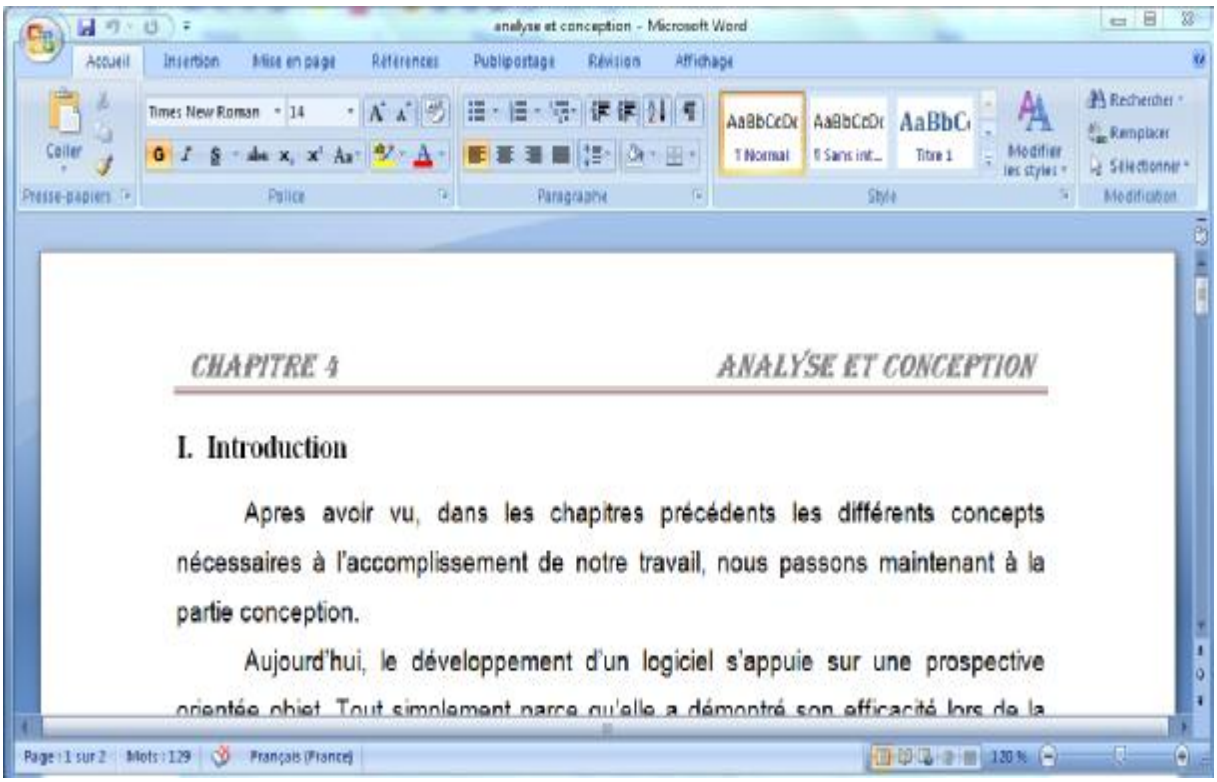

Figure V.6 : Contenu du fichier à hacher.

Pour commencer l'opération de cryptage de ce fichier, on effectue l'opération d'authentification puis on clique sur le bouton « Hachage», la page principale de hachage s'affiche et le bouton <L'empreinte de fichier par sélection> sélectionné par défaut.

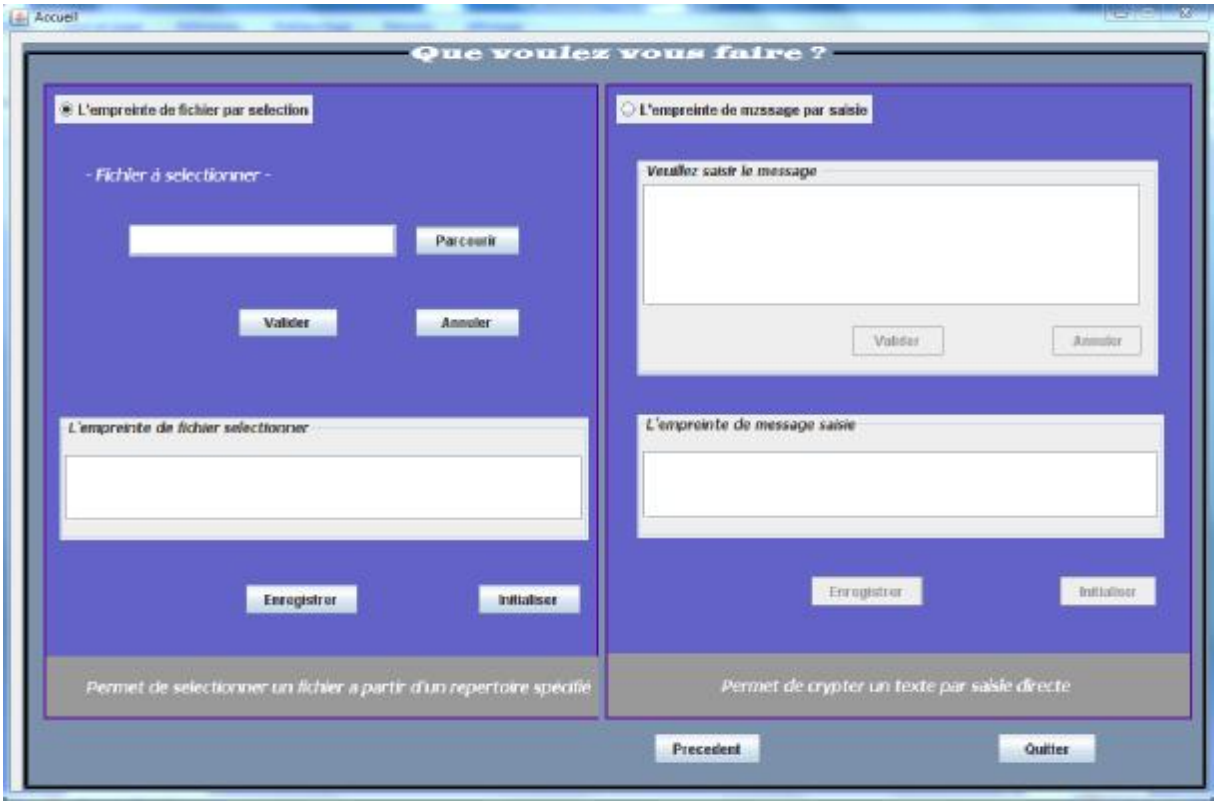

Figure V.7 : Choix de hachage < L'empreinte de fichier par sélection

On choisi dans la fenêtre principale « L'empreinte de fichier » puis en cliquant sur le bouton « Parcourir » et la fenêtre suivante s'affiche :

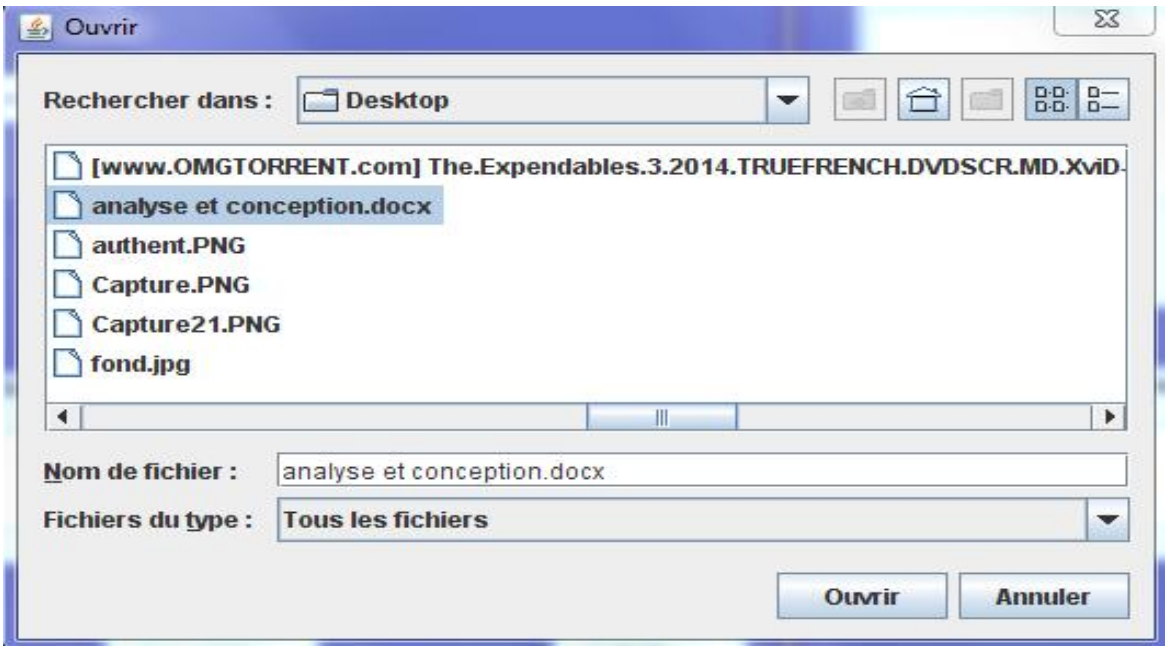

Figure V.8 : choix de fichier à sélectionner

Une fois sélectionné, le nom du fichier s'affiche dans le champ « Fichier à sélectionner », et le répertoire du fichier qui sera haché s'affiche dans le champ de sélection.

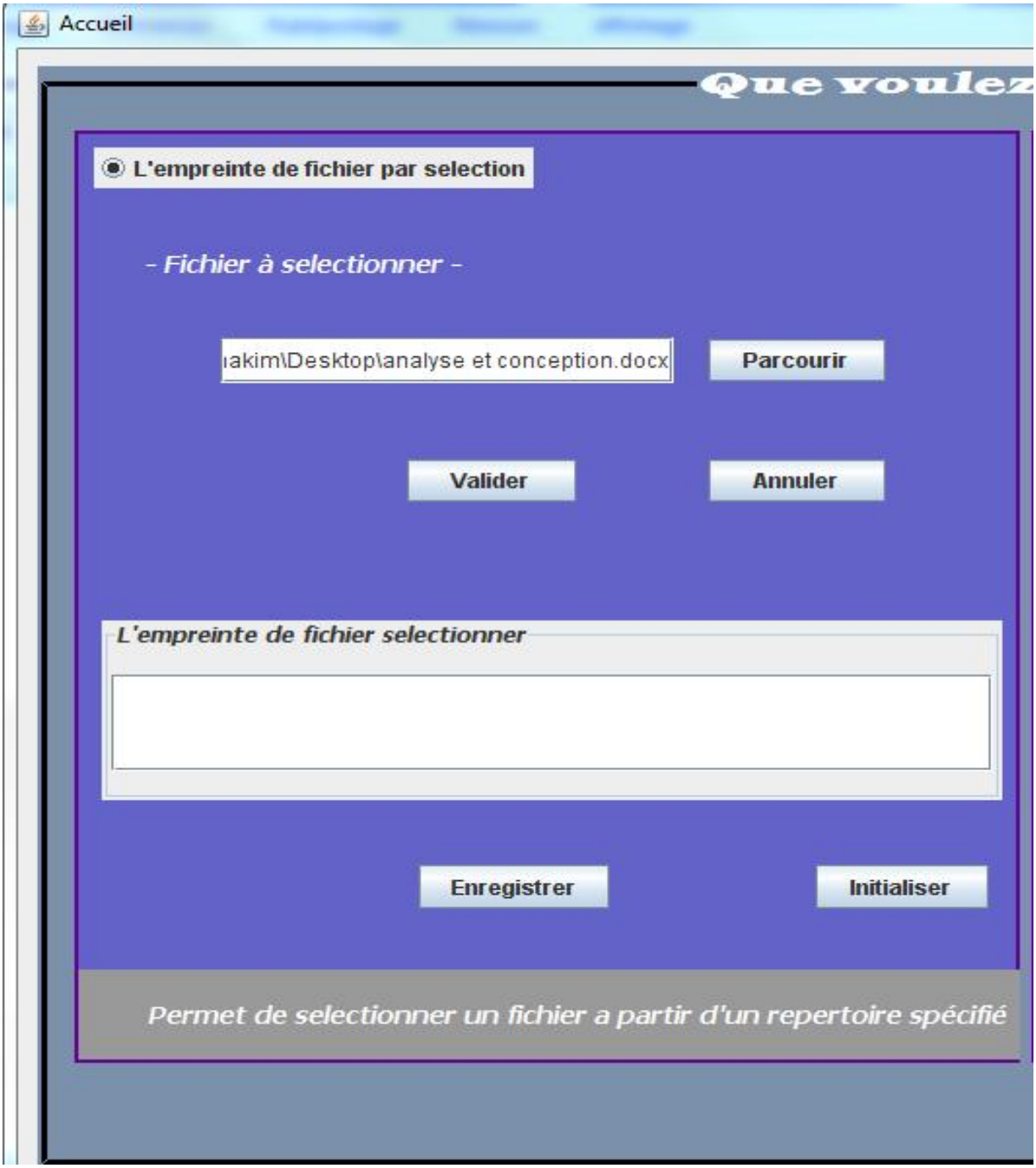

Figure V.9 : choix de fichier à hacher.

Pour trouver l'empreinte de fichier sélectionné << analyse et coception.doc >> on clique sur le bouton valider et le résultat s'affiche.

Figure V.10 : L'empreinte de fichier< analyse et coception.doc **>**

Et à la fin on a deux boutons : le premier permet d'effacer l'empreinte et le fichier sélectionné, et le second permet d'enregistrer l'empreinte puis un message de confirmation s'affiche :

Figure V.11 : message de confirmation d'enregistrement

VII. 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.

CONCLUSION GÉNÉRALE

Depuis toujours la sécurité réseaux est l'un des domaines les plus sensible que connaissent les entreprises dotées d'un réseau informatique, et malgré les efforts qui sont fourni par la communauté informatique pour mieux sécurisé la transmission des informations sur le réseau, il reste toujours des failles provoqués par des hackers qui cherchent la moindre erreur pour l'exploiter et pour l'utiliser à des fins négatives contre les entreprises. Mais il y a toujours des propositions ou des solutions qui peuvent empêcher tout cela et maitre un terme à ses failles.

Le travail que nous avons mené, nous a permis d'étudier une solution contrer ces menaces et également de nous initier à la sécurité informatique surtout dans le domaine de la cryptographie.

En effet, notre application s'est portée sur le chiffrement des informations.Le logiciel que nous avons réalisé permet de chiffrer (crypte) des fichiers, et des textes saisis,grâce à l'algorithme SHA-1 base sur les fonctions de hachage .

Bibliographie

[1]:Guy pujolle, Olivier Salvatori, Jacques Nozick, les réseaux 6éme édition, 2008.

[2] : mémoire licence, sécurisation des données 2009 /2010 réalisé par mestourlinda et tinkicht Nassim.

[3] : livre securité informatique 2009

[6]: Guy Pujolle, Les Réseaux, Edition EYROLLES, 1995-1997.

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

[10] : Andrew TANNENBAUM, LES RESEAUX, Prentice Hall, 1999.

[11] : Larry L. PETERSON, Bruce S. DAVIE, "Réseaux d'ordinateurs : une approche système". Vuibert, 1998.

[13] : «Le client / serveur » (George et Olivier Gardarin) Edition Eyrolles 2000.

[14] : mémoire licence, conception et réalisation d'une application client/serveur 2009/2010 réalisé par Chabakamel.

[15] : mémoire licence, Conception et Réalisation D'une Application Client/serveur sous ORACLE 2009/2010 réalisé par: DJOUADI Farid et BOUAMARA Tarik.

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

[18]: Jean François PillouetJean Philippe Bay, Tout sur la sécurité informatique (2eme

Édition), Edition DUNOD, 2009.

[19] : Cryptographie et Sécurité informatique, Renaud Dumont, 2009 – 2010, Notes de cours (pdf)

[20]: Phil Zimmermann, Une Introduction à la Cryptographie, Edition NAI, 1998.

[21]: Thèse, Analyse et conception de fonctions de hachage cryptographiques réalisé par M^{er}Stéphane Manuel, le 23 novembre 2010.

[22] : Thèse, Analyse de fonctions de hachage cryptographiques, réalisé par Thomas Peyrin, le 3 novembre 2008.

Sites

[4] : http://fr.wikipedia.org/wiki/ securité informatique.

[5] http://christian.calica.free.f/reseaux/

[7] http://perso.menara.13 ma/~elkharki/

[8] www.commentçamarche.net.

[12] : http://fr.wikipedia.org/wiki/Client-serveur.

[17] : http://fr.wikipedia.org/wiki/signature électronique.

[23] : http://fr.wikipedia.org/wiki/Origine_:_SHA-0_et_SHA-1.

[24] :www.nist.gov.