Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

MEMOIRE

Présenté

A L'UNIVERSITE DE TLEMCEN
FACULTE DE TECHNOLOGIE
DEPARTEMENT DE TELECOMMUNICATIONS

Pour l'obtention du diplôme de

MASTER
EN
TELECOMMUNICATIONS

Option
Réseaux Mobiles et Services de Télécommunications

Par

ALI Kheirr-dinne Mouhamed

ETUDE ET IMPLEMENTATION D'UNE SOLUTIONS DE SECURISATION DES COMMUNICATIONS PAR SSL/TLS

Soutenu le 14 juin 2015 devant le Jury

Dr.S. KAMECHE	Maître de Conférences à l'Université de Tlemcen	Président
Dr.EL. ZERROUKI	Maître de Conférences à l'Université de Tlemcen	Examinateur
Dr. A. ABDELMALEK	Maître de Conférences à l'Université de Tlemcen	Encadreur

Année Universitaire 2014/2015

Remerciement

Avant tout je voudrais remercie ALLAH le tout puissant de m'avoir donné le courage,
la volonté, la force, la patience et la chance de suivre le chemin de la science.
C'est avec mon enthousiasme le plus vif et le plus sincère que je voudrais rendre mérite à tous ceux qui à leur manière m'ont aidé à mener à bien ce travail.
Mes premiers remerciements sont adressés tout d'abord à mon encadreur Monsieur Abdelmalek Abdelhafid pour son aide consistante, ses conseils judicieux, et pour ses remarque objectifs. Il fut pour moi un directeur de travail attentif et disponible malgré ses charges nombreuses.
Je souhaite exprimer ma gratitude envers Monsieur Kameche Samir qui m'a honoré en acceptant d'être président de mon jury, ainsi qu'à Monsieur Zerrouki Hadj qui m'a honoré en acceptant d'être examinateur de mon travail.
Je profité de cette opportunité pour exprimer ma gratitude à tous les enseignants qui ont contribué par leur collaboration, disponibilité et sympathie, à notre formation.
Un grand merci pour ma familles de m'avoir soutenu, je serais jamais assez reconnaissant en vers mes parents qui ont toujours tout mis en œuvre pour qu'on s'épanouisse dans tout ce qu'on je suis aujourd'hui.
Mes remerciements à mes collègues de promotion RMST, à mes amis d'enfance et aussi à mes amis de l'association de l'A.S.P.E.W.I.T.

MERCI

Dédicace

Je dédie ce modeste travail

A mes parents

Aucun hommage ne pourrait être à la hauteur de l'amour Dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie.

A Mes frères

A Toute ma famille

A mes amis

Résume

SSL (Secure Socket Layer) et TLS (Transport Layer Secure) sont des normes qui définissent une extension de sécurité pour l'échange des données.

Les services de sécurité fournis sont la confidentialité, l'authentification et l'intégrité des données. La sécurisation se faisant au niveau des couches transport et application du modèle OSI. SSL et TLS sont déployé dans tout les systèmes de réseaux et fournir un moyen de protection unique pour tout l'échange des données. Dans ce travail, nous nous intéressons à une implémentation de cette solution afin de sécuriser les échanges entre un serveur Web et ses clients. L'implémentation est faite grâce à la boîte à outils cryptographiques Openssl.

Mot clé : SSL, TLS, Openssl, RSA, DES, AES, RC4, SHA, MD5, Client Hello, Serveur Hello, Certificat, CA.

Abstract

SSL (Secure Socket Layer) and TLS (Transport Layer Secure) are standards that define a security extension for the exchange of data.

Provided security services are confidentiality, authentication and identification, data integrity. Protections against replay and access control, these services are based on cryptographic mechanisms that give them security level student when used with strong algorithms. Securing being made in transport and application layers of the OSI model. SSL and TLS are deployed throughout the network systems and provide a single means of protection for all data exchange.

Keyword : SSL, TLS, Openssl, RSA, DES, AES, RC4, SHA, MD5, Client Hello, Serveur Hello, Certificat, CA.

Table de matiéres

Chapitre II : Mécanismes de sécurité

Chapitre III : Protocole SSL (Secure Socket Layer)/TLS (Transport Layer Secure)

Chapitre IV : Implémentation du protocole SSL \TLS avec Openssl

Liste des figures

Introduction générale

De nos jours, le réseau d'un système d'information prend une place important dans le domaine de télécommunication. Qui connais une rapide évolution. Il existe actuellement une multitude de réseau qui différent entre eux par leurs topologies, leurs protocole de communication qu'il utilisé, la nature des donnée à transmettre, les services offerts, comme il est indispensable de garantir la protection des ressource d'information.

La protection de système d'information, à l'heure actuelle connait un essor sans précédent. Il s'agit du déploiement de plusieurs solutions de sécurité successives qui permet de protéger l'information contre les attaques et les pertes des données. Ces dernières années, beaucoup d'efforts ont été fournis pour aboutir à des solutions destinées à différèrent environnement, ces solutions de sécurité qui répondent au besoins spécifiques des applications de communication, la liste des solutions de sécurité est très longue mais nous nous limitons dans ce travail à l'étude d'une de ces solutions ainsi que ses mécanismes d'authentification et l'échange des clé .

Cette solution s'applique au niveau de la sécurisation des échanges, qui est relativement plus simple à déployer et qui offre les mêmes services à toutes les applications de communication, en l'occurrence la sécurisation par le protocole de sécurisation d'échange SSL/TLS. Le protocole SSL/TLS (Secure Socket Layer)/(Transport Layer Secure) est une norme de technologie de sécurité qui permet d'établir une liaison cryptée entre deux entités communicantes et qui empêche la falsification des informations envoyées sur le réseau. Notre objectif dans ce travail est le déploiement de cette solution afin de sécuriser les échanges entre un serveur Web et ses clients.

La suite de ce mémoire est organisé en quatre partie : la première parie (chapitre 1) passe en revue les notions de base d'un système de sécurité informatique.

La deuxième parie (chapitre 2) : présent les techniques existantes pour les solutions de sécurité.

La troisième partie (chapitre 3) : décrit l'étude on détaillé du fonctionnement du protocole SSL/TLS.

Enfin, dans le chapitre 4 nous présentons l'implémentation du protocole SSL/TLS avec boîte à outils cryptographiques Openssl.

Chapitre I

Sécurité Informatique – Etat de l'art

I.1. Introduction

Aujourd'hui l'univers des systèmes d'information composé de réseaux et de système informatique prend un rôle et une place important, Cependant, L'actualité présent montre nous que le système d'information est vulnérable et qu'il peut subir des piratages, des attaques, des pertes de données, des sinistres.

Donc il est indispensable de savoir comment définir et de garantir la sécurité de ses ressources informatique. Pourtant que la sécurité des systèmes et définie comme un moyen de protéger dans un sens large le système d'information ou de minimiser les risques, par des services de sécurité qui assure la disponibilité, l'intégrité et la confidentialité.

Dans ce chapitre nous allons présenter les actions d'exposition sur un réseau et un système d'information et quelle sont les services et les fonctions dont ils besoin à mettre an ouvert à la disposition des utilisateurs pour assure ça protection.

I.2. Terminologie de la sécurité informatique

- ✓ **la sécurité informatique :** consiste à s'assurer que celui qui modifie ou consulte des données du système en a l'autorisation et qu'il peut le faire correctement car le service est disponible.

- ✓ **Les vulnérabilités** : ce sont les failles de sécurité dans un ou plusieurs systèmes. Tout système vu dans sa globalité présente des vulnérabilités, qui peuvent être exploitables ou non. [1]

- ✓ **Les menaces :** ce sont des adversaires déterminés capables de monter une attaque exploitant une vulnérabilité. [1]

- ✓ **Les attaques (exploits):** elles représentent les moyens d'exploiter une vulnérabilité. Il peut y avoir plusieurs attaques pour une même vulnérabilité mais toutes les vulnérabilités ne sont pas exploitables. [1]

- ✓ **Le risque** : est la combinaison d'une menace et des pertes qu'elle peut engendrer: c'est-à-dire de la potentialité de l'exploitation de vulnérabilité par un élément menaçant et de l'impact sur l'organisme. On remarquera que la notion de risque dépend de l'impact: une menace ayant une grande probabilité de se concrétiser, mais ayant un impact nul ne constitue pas un risque nul. Dans le langage courant, l'acceptation du mot "risque" peut être un peu différente et ne pas intégrer la notion d'impact. [2]

- ✓ **Impact:** conséquence sur l'organisme de la réalisation d'une menace. L'impact peut être exprimé financièrement, ou dans une échelle qui dépend du contexte. [2]

- ✓ **Les contre-mesures :** ce sont les procédures ou techniques permettant de résoudre une vulnérabilité ou de contrer une attaque spécifique (auquel cas il peut exister d'autres attaques sur la même vulnérabilité). [1]

- ✓ **Mécanismes de Sécurité** : un mécanisme qui est conçu pour détecter, prévenir et lutter contre une attaque de sécurité. [3]

- ✓ **Service de Sécurité :** un service qui augmente la sécurité des traitements et des échanges de données d'un système. Un service de sécurité utilise un ou plusieurs mécanismes de sécurité. [3]

I.3. Vulnérabilité, Menace, Attaque

I.3.1. Vulnérabilité

La vulnérabilité « en anglais « vulnerability », appelée parfois faille ou brèche représente le niveau d'exposition face à la menace dans un contexte particulier. Une vulnérabilité informatique est spécifiquement une faille ou faiblesse dans un composant matériel ou logiciel qui permet d'atteint l'intégrité et la confidentialité des données personnel. C'est-à-dire un attaquant se permet d'avoir accès non autorisé ou vole et

contourner l'information, mais généralement un attaquant cherche l'exploitation des bugs dans un logiciel.

Exemples de vulnérabilités :

- ✓ Utilisation des mots de passe non robustes
- ✓ Présence de comptes non protégés par mot de passe
- ✓ La sécurité est chère et difficile: Les organisations n'ont pas de budget pour ça
- ✓ La sécurité ne peut être sûre à 100%, elle est même souvent inefficace
- ✓ La politique de sécurité est complexe et basée sur des jugements humains
- ✓ Les organisations acceptent de courir le risque, la sécurité n'est pas une priorité
- ✓ De nouvelles technologies (et donc vulnérabilités) émergent en permanence
- ✓ Les systèmes de sécurité sont faits, gérés et configurés par des hommes

I.3.2. Menace

La menace (en anglais « threat ») représente le type d'action susceptible de nuire dans l'absolu. Ceux sont les résultantes d'actions et d'opération du fait d'autrui. Il existe deux catégories :

Passives : atteinte à la confidentialité (prélèvement par copie, écoute de l'information sur les voies de communication), Cette catégorie est souvent plus difficile à détecter car elle ne cause aucune altération des données. Le but de l'adversaire est de collecter les informations qui ont été transmises (adresse IP, port, services, environnement, etc.) afin d'analyser leur contenus et de mener par la suite des attaques actives.

Actives : nuisent à l'intégrité des données (brouillage, déguisement (se faire passer pour quelqu'un d'autre), interposition (vol de session). On remarque principalement dans cette catégorie les attaques suivantes:

- L'homme du milieu : cette attaque a lieu lorsqu'une entité prétend être une autre entité. Dans la plupart du temps, l'attaquant utilise les techniques de détournement de flux pour rediriger les flux des deux bouts de la communication vers lui. Ceci, afin de surveiller tout leur trafic réseau et de le modifier à sa guise pour l'obtention de tout type d'information.

- Le *rejeu des paquets* : le rejeu implique la capture passive de données et leur retransmission ultérieure en vue de produire un effet non autorisé. Pour empêcher tel type d'attaques, on utilise souvent les nombres aléatoires dans les messages envoyés.
- La *modification des messages* : ceci signifie que certaines portions d'un message légitime sont altérés ou que les messages sont retardés ou réorganisés. Ce type d'attaques est souvent utilisé avec les protocoles qui sont basés sur le protocole de transport UDP (comme le cas du protocole ISAKMP).

- Le *déni de service* : cette attaque porte bien son nom puisque qu'elle aboutira à l'indisponibilité du service (saturation des ressources allouées à une application spécifique), de la machine visée ou même la saturation d'un réseau complet. Cette attaque vise surtout l'exploitation d'une mauvaise implémentation d'un protocole ou à des faiblesses de celui-ci. Pour les protocoles de sécurité, le déni de service peut prendre deux formes :

  - La première forme vise la vulnérabilité des protocoles de transport qu'ils utilisent tels que les attaques sur le protocole IP (IP Spoofing), le protocole TCP (SYN Flooding) et le protocole UDP (UDP Flooding, Packet Fragment).
  - La deuxième forme porte surtout sur la phase d'initialisation de ces protocoles (Le protocole IKE, le Handshake de SSL/TLS, etc.). [4]

I.3.3. Programmes malveillants

I.3.3.1. Les virus

Un virus est un petit programme informatique situé dans le corps d'un autre, qui, lorsqu'on l'exécute, se charge en mémoire et exécute les instructions que son auteur a programmé. La définition d'un virus pourrait être la suivante :

« Tout programme d'ordinateur capable d'infecter un autre programme d'ordinateur en le modifiant de façon à ce qu'il puisse à son tour se reproduire. » Le véritable nom donné aux virus est *CPA* soit *Code Auto-Propageable*, mais par analogie avec le domaine médical, le nom de "virus" leur a été donné. Les virus résidents (appelés TSR en anglais pour *Terminate and stay resident*) se chargent dans la mémoire vive de l'ordinateur afin d'infecter les fichiers exécutables lancés par l'utilisateur. Les virus non résidants infectent les programmes présents sur le disque dur dès leur exécution.

I. 3.1.2. Les vers

Un ver informatique (en anglais *worm*) est un programme qui peut s'auto-reproduire et se déplacer à travers un réseau en utilisant les mécanismes réseau, sans avoir réellement besoin d'un support physique ou logique (disque dur, programme hôte, fichier, etc.) pour se propager; un ver est donc un virus réseau. La plus célèbre des vers date de 1988. Un étudiant (Robert T. Morris, de Cornell University) avait fabriqué un programme capable de se propager sur un réseau, il le lança et, 8 heures après l'avoir lâché, celui-ci avait déjà infecté plusieurs milliers d'ordinateurs.

C'est ainsi que de nombreux ordinateurs sont tombés en pannes en quelques heures car le « ver » (car c'est bien d'un ver dont il s'agissait) se reproduisait trop vite pour qu'il puisse être effacé sur le réseau. De plus, tous ces vers ont créé une saturation au niveau de la bande passante, ce qui a obligé la NSA à arrêter les connexions pendant une journée.

Les vers actuels se propagent principalement grâce à la messagerie (et notamment par le client de messagerie Outlook) grâce à des fichiers attachés contenant des instructions permettant de récupérer l'ensemble des adresses de courrier contenues dans le carnet d'adresse et en envoyant des copies d'eux-mêmes à tous ces destinataires.

Ces vers sont la plupart du temps des scripts (généralement VBScript) ou des fichiers exécutables envoyés en pièce jointe et se déclenchant lorsque l'utilisateur destinataire clique sur le fichier attaché.

I.3.1.3. Cheval de Troie

On appelle « Cheval de Troie » (en anglais trojan horse) un programme informatique effectuant des opérations malicieuses à l'insu de l'utilisateur. Le nom « Cheval de Troie » provient d'une légende narrée dans l'Iliade (de l'écrivain Homère) à propos du siège de la ville de Troie par les Grecs. La légende veut que les Grecs, n'arrivant pas à pénétrer dans les fortifications de la ville, aient l'idée de donner en cadeau un énorme cheval de bois en offrande à la ville en abandonnant le siège. Les troyens (peuple de la ville de Troie), apprécièrent cette offrande à priori inoffensive et la ramenèrent dans les murs de la ville. Cependant le cheval était rempli de soldats cachés qui s'empressèrent d'en sortir à la tombée de la nuit, alors que la ville entière était endormie, pour ouvrir les portes de la cité et en donner l'accès au reste de l'armée ...

Un cheval de Troie (informatique) est donc un programme caché dans un autre qui exécute des commandes sournoises, et qui généralement donne un accès à l'ordinateur sur

lequel il est exécuté en ouvrant une porte dérobée (en anglais backdoor), par extension il est parfois nommé troyen par analogie avec les habitants de la ville de Troie. , A la façon du virus, le cheval de Troie est un code (programme) nuisible placé dans un programme sain (imaginez une fausse commande de listage des fichiers, qui détruit les fichiers au-lieu d'en afficher la liste).

Un cheval de Troie peut par exemple

- voler des mots de passe ;
- copier des données sensibles ;
- exécuter tout autre action nuisible ;

Un tel programme peut créer, de l'intérieur du réseau, une brèche volontaire dans la sécurité pour autoriser des accès à des parties protégées du réseau à des personnes se connectant de l'extérieur.

Les principaux chevaux de Troie sont des programmes ouvrant des ports de la machine, c'est-à-dire permettant à son concepteur de s'introduire sur votre machine par le réseau en ouvrant ou une porte dérobée.

C'est la raison pour laquelle on parle généralement de backdoor (littéralement porte de derrière) ou de back orifice (terme imagé vulgaire signifiant "orifice de derrière").

Un cheval de Troie n'est pas nécessairement un virus, dans la mesure où son but n'est pas de se reproduire pour infecter d'autres machines. Par contre certains virus peuvent également être des chevaux de Troie, c'est-à-dire se propager comme un virus et ouvrir un port sur les machines infectées !

Détecter un tel programme est difficile car il faut arriver à détecter si l'action du programme (le cheval de Troie) est voulue ou non par l'utilisateur.

I.3.1.4. Les bombes logiques

Sont appelés bombes logiques les dispositifs programmés dont le déclenchement s'effectue à un moment déterminé en exploitant la date du système, le lancement d'une commande, ou n'importe quel appel au système.

Ainsi ce type de virus est capable de s'activer à un moment précis sur un grand nombre de machines (on parle alors de bombe à retardement ou de bombe temporelle), la bombe

logique Tchernobyl s'est activée le 26 avril 1999, jour du 13ème anniversaire de la catastrophe nucléaire ...

Les bombes logiques sont généralement utilisées dans le but de créer un déni de service en saturant les connexions réseau d'un site, d'un service en ligne ou d'une entreprise.

I.3.1.5. Espiogiciels

Un espiogiciel (en anglais spyware) est un programme chargé de recueillir des informations sur l'utilisateur de l'ordinateur sur lequel il est installé (on l'appelle donc parfois mouchard) afin de les envoyer à la société qui le diffuse pour lui permettre de dresser le profil des internautes (on parle de profilage).

Les récoltes d'informations peuvent ainsi être :

- la traçabilité des URL des sites visités,
- le traquage des mots-clés saisis dans les moteurs de recherche,
- l'analyse des achats réalisés via internet,
- voire les informations de paiement bancaire (numéro de carte bleu / VISA)
- ou bien des informations personnelles.

Les spywares s'installent généralement en même temps que d'autres logiciels (la plupart du temps des freewares ou sharewares). En effet, cela permet aux auteurs des dits logiciels de rentabiliser leur programme, par de la vente d'informations statistiques, et ainsi permettre de distribuer leur logiciel gratuitement. Il s'agit donc d'un modèle économique dans lequel la gratuité est obtenue contre la cession de données à caractère personnel.

Les spywares ne sont pas forcément illégaux car la licence d'utilisation du logiciel qu'ils accompagnent précise que ce programme tiers va être installé ! En revanche étant donné que la longue licence d'utilisation est rarement lue en entier par les utilisateurs, ceux-ci savent très rarement qu'un tel logiciel effectue ce profilage dans leur dos.

Par ailleurs, outre le préjudice causé par la divulgation d'informations à caractère personnel, les spywares peuvent également être une source de nuisances diverses :

- consommation de mémoire vive,
- utilisation d'espace disque,
- mobilisation des ressources du processeur,
- plantages d'autres applications,

- gêne ergonomique (par exemple l'ouverture d'écrans publicitaires ciblés en fonction des données collectées).

Types de spywares :

On distingue généralement deux types de spywares :

- ✓ Les spywares internes (ou spywares internes ou spywares intégrés) comportant directement des lignes de codes dédiées aux fonctions de collecte de données.
- ✓ Les spywares externes, programmes de collectes autonomes installés

Voici une liste non exhaustive de spywares non intégrés :

Alexa, Aureate/Radiate, BargainBuddy, ClickTillUWin, Conducent Timesink, Cydoor, Comet Cursor, Doubleclick, DSSAgent, EverAd, eZula/KaZaa Toptext, Flashpoint/Flashtrack, Flyswat, Gator / Claria, GoHip, Hotbar, ISTbar, Lop, NewDotNet, Realplayer, SaveNow, Songspy, Xupiter, Web3000 et WebHancer

I.3.1.6. Le keylogger

Un keylogger (littéralement enregistreur de touches) est un dispositif chargé d'enregistrer les frappes de touches du clavier et de les enregistrer, à l'insu de l'utilisateur. Il s'agit donc d'un dispositif d'espionnage.

Certains keyloggers sont capables d'enregistrer les URL visitées, les courriers électroniques consultés ou envoyés, les fichiers ouverts, voire de créer une vidéo retraçant toute l'activité de l'ordinateur !

Dans la mesure où les keyloggers enregistrent toutes les frappes de clavier, ils peuvent servir à des personnes malintentionnées pour récupérer les mots de passe des utilisateurs du poste de travail ! Cela signifie donc qu'il faut être particulièrement vigilant lorsque vous utilisez un ordinateur en lequel vous ne pouvez pas avoir confiance (poste en libre accès dans une entreprise, une école ou un lieu public tel qu'un cybercafé).

I.3.1.7. Les spams

On appelle spam (le terme de pourriel est parfois également utilisé) l'envoi massif de courrier électronique à des destinataires ne l'ayant pas sollicité. Le spam consiste à envoyer plusieurs e-mails identiques (souvent de type publicitaire) à un grand nombre de personnes sur internet.

I.4. Les attaques

Le réseau d'entreprise reste vulnérable à de nombreuses attaques potentielles, ces attaques peuvent compromettre la fiabilité des données de l'entreprise et donc réduire sa crédibilité. Une « attaque » est l'exploitation d'une faille d'un système informatique (système d'exploitation, logiciel ou bien même de l'utilisateur) à des fins non connues par l'exploitant des systèmes et généralement préjudiciables.

Sur internet des attaques ont lieu en permanence, à raison de plusieurs attaques par minute sur chaque machine connectée. Ces attaques sont pour la plupart lancées automatiquement à partir de machines infectées (par des virus, chevaux de Troie, vers, etc.), à l'insu de leur propriétaire. Plus rarement il s'agit de l'action de pirates informatiques.

Attaque = cible + méthode + Vulnérabilités

- Attaque par TCP :

Le protocole TCP utilise des numéros de port qui permettent de détermine une adresse de socket, c'est-à-dire un point d'accès au réseau. Cette adresse de socket est obtenue par la concaténation de l'adresse IP et de l'adresse de port. Une attaque par TCP revient à utiliser un point d'accès pour faire autre chose que ce dont le point d'accès a été défini. En particulier, un pirate peut utiliser un port classique pour entrer dans un ordinateur ou dans le réseau d'une entreprise.

- Attaque par dictionnaire :

C'est une méthode utilisée pour trouver un mot de passe ou une clé. Elle consiste à tester une série de mot de passe potentiel, les une à la suite des autres en espérant que le mot de passe utilisé pour le chiffrement soit contenu dans le dictionnaire. Si tel n'est pas le cas le cas l'attaque échouera.

- Attaque par force brute :

L'attaque par force brute est une méthode utilisée en cryptanalyse pour trouver un mot de passe ou une clé, Il s'agit de tester, une à une, toutes les combinaisons possibles. Cette méthode de recherche exhaustive ne réussite que les cas ou le mot de passe cherché est constitué de peu de caractères. Ces programmes tentent toutes les possibilités de mot de passe dans un ordre aléatoire afin de berner les logiciels de sécurité qui empêchent de tenter tous les mots de passe dans l'ordre.

Pour contrer cette méthode, il suffit simplement de choisir des mots de passe d'une grande longueur ou des clés suffisamment grandes. Ainsi, l'attaquant devra mettre beaucoup de temps pour trouver le bon mot passe. Cette méthode est très sensible aux capacités de calcul des machines effectuant l'algorithme.
Cette méthode est souvent combinée avec l'attaque par dictionnaire et table arc-en-ciel pour obtenir de meilleurs résultats.

Elle n'est pas une attaque à proprement parler car elle contente de définir le plus petit temps qu'il faudra pour trouver le secret. Cette méthode étant applicable à n'importe quel algorithme, lui donner le titre d'attaque signifierait que tous les protocoles sont attaqués et donc non fiables, Il s'agit donc d'un abuse de langage fréquent.

- Attaque radio :

Les faiblesses du WPA permettent aux pirates de contourner le chiffrement des données transmises sans fil. Ils peuvent espionner passivement le contenu des messages ou analyser le trafic d'un réseau WLAN en vue d'attaques futures. Ces intrus peuvent également lancer des attaques actives, par exemple en rediffusant ou modifiant des messages.
Le déversement massif d'onde parasites sur un point d'accès sans fil est autre attaque courante. Le pirate peut alors installer un point d'accès factice et détourner les systèmes de communication en se faisant passer pour une ressource légitime.

✓ Attaques visant le réseau d'entreprise :

Les réseaux WLAN représentent une double menace pour les ressources légitimes de l'entreprise. Tout d'abord, les faiblesses des mécanismes d'authentification permettent à tout intrus non autorisé de pénétrer aisément le réseau de l'entreprise et d'accéder à ses ressources.

Ensuite, la plupart des implémentations sans fil ne contrôlent que très peu l'accès entre le segment du réseau sans fil et réseau câble. Les pirates peuvent donc diriger tout le trafic voulu vers les ressources de l'entreprise. Comparés une passerelle Internet, les points d'accès sans fil ne cherchent quasiment pas, voir pas du tout, à détecter et à se protéger contre le trafic malveillant.

✓ Attaques visant les ordinateurs clients

Bien que les clients sans fil soient situés à l'extérieur de u pare-feu de l'entreprise, plusieurs d'architecture réseau les traitent comme des clients internes. Les pirates peuvent alors compromettre des clients individuels, en les utilisant pour collecter des informations ou comme des tunnels d'entrée vers le réseau de l'entreprise. En mode 802.11 « ah-doc », les ordinateurs clients sont autorisés à se connecter directement entre eux ouvrant ainsi un véritable boulevard aux pirates. [4]

I.5. Les risques

Les risques dépendent des paramètres que l'on peut maitriser. Il existe deux types de risques :

- Le risque structurel : dépend de l'organisation de l'entreprise.
- Le risque accidentel : indépendant de tous les facteurs de l'entreprise.

Et existe quatre niveaux de risque :

- Acceptables : pas de conséquences graves pour les utilisateurs de réseau.
  Exemple : panne électrique, perte de liaison.
- Courants : pas de préjudices graves au réseau, on répare facilement.
  Exemple : gestion de réseau, mauvaise configuration, erreur utilisateur.
- Majeurs : dus à des facteurs graves et qui causent de gros dégâts mais récupérable.
  Exemple : foudre qui tombe sur un routeur.
- Inacceptables : fatals pour l'entreprise, ils peuvent entrainer son dépôt de bilan.
  Exemple : perte ou corruption des informations importantes.
- Majeurs : dus à des facteurs graves et qui causent de gros dégâts mais récupérable.
  Exemple : foudre qui tombe sur un routeur. [4]

Risque = Menace × Vulnérabilité/contre-mesure

I.6. Les contre-mesures

Contre-mesures techniques : fonctions et mécanismes dédiés à la sécurité d'un système, la contre-mesure est l'ensemble des actions mises en œuvre en prévention de la menace. Les contre-mesures à mettre en œuvre ne sont pas uniquement des solutions techniques mais également des mesures de formation et de sensibilisation à l'intention des utilisateurs, ainsi qu'un ensemble de règles clairement définies.

I.7. Services de sécurité

Les services de sécurité représentent les logiciels et matériels mettant en œuvre des mécanismes dans le but de mettre à la disposition des utilisateurs les fonctions de sécurité dont ils ont besoin.

I.7.1. Disponibilité

La disponibilité est de pair avec son accessibilité : Une ressource doit être accessible, avec un temps de réponse acceptable.

La disponibilité des services, système et données est obtenue :

- Par un dimensionnement approprié
- Par une gestion opérationnelle des ressources et des services

Ce paramètre est mesuré par une montée en charge du système afin de s'assurer de la totale disponibilité du service. Un service doit aussi être assuré avec le minimum d'interruption en respect avec l'engagement établi.

De plus des pertes de données sont possibles si l'enregistrement et le stockage ne sont pas gérés correctement, d'où l'importance d'une haute disponibilité d'un système et de la mise en place d'une politique de sauvegarde. [5]

I.7.2. Authentification

L'authentification a pour but de garantir l'identité des correspondants. D'après le dictionnaire de l'informatique, l'authentification est l'opération par laquelle le destinataire et/ou l'émetteur d'un message s'assure de l'identité de son interlocuteur.

Le service d'authentification permet d'assurer qu'une communication est authentique. On peut distinguer deux types d'authentification: l'authentification d'un tiers et l'authentification de la source des données.

L'authentification d'un tiers consiste pour ce dernier à prouver son identité. L'authentification de la source des données sert à prouver que les données reçues viennent bien d'un tel émetteur déclaré. Les signatures numériques peuvent aussi servir à l'authentification, La signature numérique sera abordée dans la section Intégrité.

L'authentification nécessite de fournir une identification et de la prouver. Sur la plupart des réseaux, le mécanisme d'authentification utilise une paire code d'identification /mot de passe.

Cependant, en raison de la vulnérabilité constamment associée à l'utilisation des mots de passe, il est souvent recommandé de recourir à des mécanismes plus robustes tels que l'authentification par des certificats, des clés publiques ou à travers des centres de distribution des clés.

- Les protocoles d'authentification :

Un protocole d'authentification est un moyen de contrôle d'accès caractérisé par les trois A (AAA) : Authentification (authentification), Authorization (autorisation), Accounting (rapport).

Les protocoles d'authentification utilisent différentes manières d'authentifier un utilisateur ou une machine. Il existe différents algorithmes, différentes techniques, mais tous, dans un souci de sécurité utilisant le principe de chiffrement qui est à base de clés.

- Protocole PAP (Password Authentication Protocol) :

Le protocole PAP est, comme son nom l'indique, basé sur l'authentification par mot de passe. Les mots de passe sont envoyés en clair sur le réseau, ce qui représenté un danger important. PAP est un protocole d'autorisation d'accès pour l'ouverture d'une session sur le réseau. Il est de moins en moins utilisé au profit CHAP.

- Protocole CHAP(Challenge-Handshake Authentiction Protocol) :

Le protocole CHAP est basé sur le mode d'authentification « Défi-Réponse ». Le serveur d'authentification envoie un identifiant au hasard, c'est le défi. Le client transforme le défi avec sa clé et l'algorithme MD5 puis le renvoie au serveur : c'est la réponse. Le serveur applique le même algorithme avec la clé de client, compare les deux résultats puis accorde ou rejette la connexion, ce qui le rend relativement.

- Protocole TACACS (Terminal Acccess Controller Acess Control System) :

TATCACS est le plus ancien des protocoles d'authentification. Il a été récemment actualisé dans une nouvelle variante appelée TACACS supporte plusieurs types d'authentification : l'authentification dite classique avec nom d'utilisateur/mot de passe complétée par l'utilisation des challenges. Le mécanisme d'authentification donne la possibilité, après la transaction du login (nom d'utilisateur)

Et du mot de passe, de vérifier son identité en lui posant un certain nombre de questions.

- Protocole RADIUS(Remote Authentication Dial-User Service) :

Mis au point initialement par Livingston, est un protocole d'authentification standard, qui fonctionne selon le monde Client/serveur.

Un point d'accès (routeur,switch,serveur) fonctionne comme un client RADIUS qui effectue des requêtes sur le serveur. Le standard RADUIS est basé sur un ensemble d'attributs relatifs aux utilisateurs mais beaucoup d'implémentations spécifiques du protocole apportent leur propre jeu d'attributs. De plus, toutes les transactions RADUIS entre le client et le serveur sont protégées par un secret partagé qui n'est jamais transmis sur le réseau, ce qui représente une sécurité supplémentaire. [4]

I.7.3. Intégrité

Dans certains cas, Il peut être nécessaire d'assurer simplement que les données sont intègres, c'est à dire qu'elles n'ont pas été au passage falsifiées par un intrus. Ces données restent « claires », au sens ou elles ne sont pas secrètes.

L'intégrité se rapporte à la protection contre les changements et les altérations. Il y a une intégrité si les données émises sont identiques à celles reçues. Des différences peuvent apparaître si quelqu'un tente de modifier ces données ou tout simplement si un problème de transmission/réception intervient.

Les techniques utilisées pour faire face à cela sont, les bits de parité, les checksums ou encore les fonctions de hachage à sens unique. Ces mécanismes ne peuvent cependant pas garantir absolument l'intégrité. Il est possible en effet, que les données altérées aient la même somme de contrôle. Il est aussi possible qu'un attaquant modifie les données et recalcule le résultat de la fonction de hachage (empreinte). Pour que seul l'expéditeur soit capable de modifier l'empreinte, on utilise des fonctions de hachage à clés secrètes ou privées. Dans ce cas, on garantit à la fois l'intégrité et l'authentification Ces deux services de sécurité sont souvent fournis par les mêmes mécanismes pour la simple raison qu'ils n'ont de sens qu'accompagnés l'un de l'autre (dans le contexte d'un réseau peu sûr).

I.7.4. La non-répudiation

La non-répudiation est le fait de ne pouvoir nier ou rejeter qu'un événement a eu lieu A cette notion sont associées L'imputabilité: une action a eu lieu et automatiquement un enregistrement, preuve de l'action, est effectué La tracabilité: mémorisation de l'origine du

message L'auditabilité: capacité d'un système à garantir la présence d'informations nécessaires à une analyse ultérieure d'un événement. L'existence de fichiers journal permet de garantir l'imputation et l'auditabilité.

La non répudiation empêche tant l'expéditeur que le destinataire de nier avoir transmis un message. On dénombre deux types de service de non répudiation :

1. La non répudiation de l'origine qui protège un destinataire confronté à un expéditeur niant avoir envoyé le message.

2. La non répudiation de la réception qui joue le rôle inverse du précédent, à savoir démontré que le destinataire a bien reçu le message que l'expéditeur lui a envoyé.

Dans le cadre de la cryptographie à clé publique, chaque utilisateur est le seul et unique détenteur de la clé privée. Ainsi, tout message accompagné par la signature électronique d'un utilisateur ne pourra pas être répudié par celui-ci, à moins que tout le système de sécurité n'ait été pénétré.

A l'opposé, la non répudiation n'est pas directement acquise dans les systèmes utilisant des clés secrètes. La clé de chiffrement étant distribuée par le serveur de distribution de clés aux deux parties, un utilisateur peut nier avoir envoyé le message en question en alléguant que la clé secrète partagée a été divulguée soit par une compromission du destinataire, soit par une attaque réussie contre le serveur de distribution de clés.

La non répudiation de la réception peut se faire en obligeant le destinataire à envoyer un accusé de réception signé et horodaté.

I.7.5. Confidentialité

La confidentialité est un service de sécurité qui consiste à assurer que seules les personnes autorisées peuvent prendre connaissance des données. Pour obtenir ce service, on utilise généralement le chiffrement des données concernées à l'aide d'un algorithme cryptographique.

Si seules les données sont chiffrées, une oreille espionne peut tout de même écouter les informations de l'en-tête. Elle peut ainsi, à partir des adresses source et destination, identifier les tiers communicants et analyser leur communication : fréquence des envois, quantité de données échangée, etc. On parle de protection contre l'analyse de trafic lorsqu'en plus de la confidentialité, on garantit l'impossibilité de connaître ces informations.

L'authentification, l'intégrité et la confidentialité vont souvent ensemble et offrent la base des services de sécurité.

I.8. Conclusion

L'objectif de ce chapitre est de se familiariser avec les notions de base de la sécurité informatique, telles que l'exigence de la sécurité informatique, les attaques employées sur le réseau et sur les machines, ainsi que les solutions proposées pour contrecarrer ces attaques.

Les laboratoires de recherche étudient aujourd'hui les meilleures solutions de la protection des données en développant des outils visant à empêcher, la divulgation, la modification, et l'utilisation non-autorisées de ressources réseau ou informatiques de façon générale.

Dans le chapitre suivant, nous allons détailler quelques mécanismes utilisés dans les solutions de sécurité déployées dans les systèmes d'information.

Table des matières

Chapitre II

Mécanismes de sécurité

II.1. Introduction

Les mécanismes de sécurité ont une importance capitale dans toute solution de sécurité informatique. Dans ce chapitre, nous allons survoler les plus pertinents d'entre eux, notamment ceux basés sur la cryptographie symétrique et asymétrique, et ceux basés sur les fonctions à sens unique, les protocoles associés ainsi que les outils de protection des réseaux.

II.2. La cryptographie

Le mot cryptographie est un terme générique désignant l'ensemble des techniques permettant de chiffrer des messages, c'est-à-dire permettant de les rendre inintelligibles sans une action spécifique.

La cryptographie est essentiellement basée sur l'arithmétique, Il s'agit dans le cas d'un texte de transformer les lettres qui composent le message en une succession de chiffres (sous forme de bits dans le cas de l'informatique car le fonctionnement des ordinateurs est basé sur le binaire), puis ensuite de faire des calculs sur ces chiffres pour les modifier de telle façon à les rendre incompréhensibles. Le résultat de cette modification (le message chiffré) est appelé cryptogramme (en anglais ciphertext) par opposition au message initial, appelé message en clair (en anglais plaintext), faire en sorte que le destinataire saura les déchiffrer.

Le fait de coder un message de telle façon à le rendre secret s'appelle chiffrement. La méthode inverse, consistant à retrouver le message original, est appelée déchiffrement.

II.2.1. Les techniques de cryptographie

Pour assurer les objectifs de la cryptographie nous pouvons utiliser des algorithmes basés sur des clés. Ces algorithmes sont définis par plusieurs types de cryptographie ou crypto systèmes.

II.2.2.1 Cryptographie symétrique

Le chiffrement symétrique (aussi appelé à chiffrement à clé privée ou chiffrement à clé secrète) consiste à utiliser la même clé pour le chiffrement et le déchiffrement. C'est le plus facile à comprendre, c'est aussi la méthode de chiffrement la plus facile à réaliser et qui consomme le moins de ressources se calcul et de band passante.

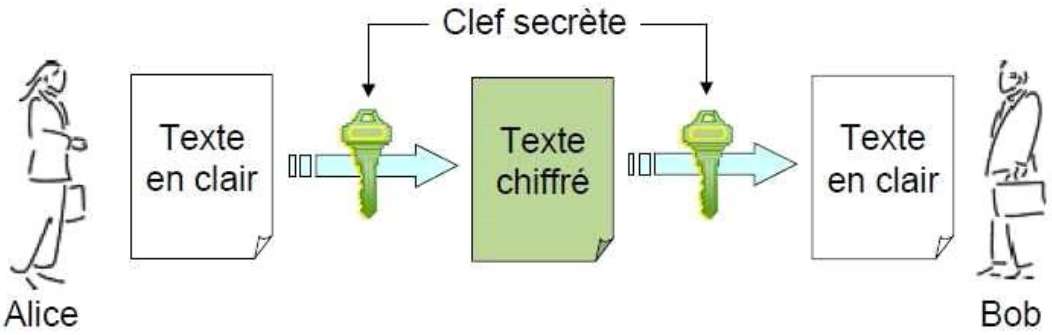


Figure II.1 : Chiffrement symétrique

Les deux hôtes qui doivent échanger des données confidentielles (secrètes) disposent tous deux d'une clé identique. L'émetteur chiffre les données avec, puis les envoie au récepteur. Ce dernier déchiffre avec la même clé pour récupérer des données lisibles.

Cette méthode assure la confidentialité des données, celui qui intercepterait la communication ne pourra pas lire les données échangées tant qu'il n'aura pas pu se procurer la clé. Il n'y a aucune authentification de faire sur l'émetteur comme sur le récepteur, sauf si deux personnes seulement disposent de la clé.

Le principal souci avec cette méthode, c'est qu'il faut s'échanger la clé et lors de cet échange, sans précautions particulières, n'importe quoi peut se produire.

Il y a deux catégories de systèmes à clé privée : les chiffrements par blocs et les chiffrements de flux. [7].

II.2.2.2. Chiffrement par bloc (Block Cipher)

C'est une des deux grandes catégories de chiffrements modernes en cryptographie symétrique.

Il consiste à un découpage des données en blocs de taille généralement fixe (souvent une puissance de deux comprise entre 32 et 512 bits). Les blocs sont ensuite chiffrés les uns après les autres. Il est possible de transformer un chiffrement de bloc en un chiffrement par flot en utilisant un

mode d'opération comme ECB (chaque bloc chiffré indépendamment des autres) ou CFB (on chaîne le chiffrement en effectuant un XOR entre les résultats successifs).

Un exemple de taille de bloc et de clé utilisés par les algorithmes les plus connus:

DES : blocs de 64 bits, clé de 56 bits.

IDEA : blocs de 64 bits, clé de 128 bits.

AES : blocs de 128 bits, clé de 128 à 256 bits. [7]

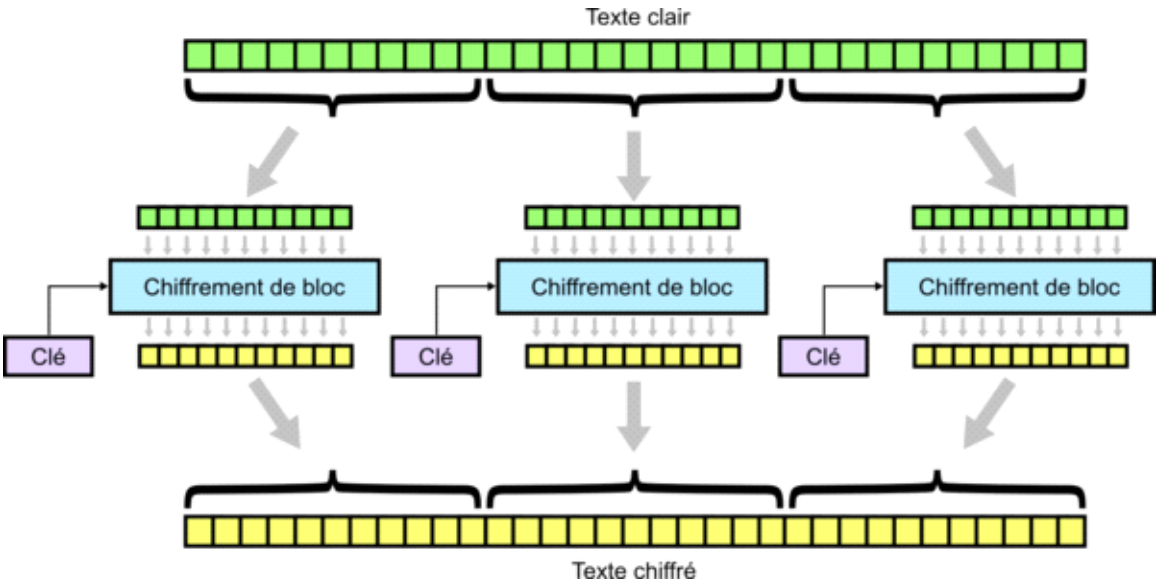


Figure II.2 : Chiffrement par bloc

II.2.2.3. Chiffrements de flux (Stream Cipher)

Les algorithmes de chiffrement de flux peuvent être définis comme étant des algorithmes de chiffrement par blocs, où le bloc a une dimension unitaire (1 bit, 1 octet, etc.) ou relativement petite. Leurs avantages principaux viennent du fait que la transformation (méthode de chiffrement) peut être changée à chaque symbole du texte clair et du fait qu'ils soient extrêmement rapides. De plus, ils sont utiles dans un environnement où les erreurs sont fréquentes car ils ont l'avantage de ne pas propager les erreurs (diffusion). Ils sont aussi utilisés lorsque l'information ne peut être traitée qu'avec de petites quantités de symboles à la fois. Quelques algorithmes de cryptographie symétrique par flot:

A5 : utilisé dans les téléphones mobiles de type GSM pour chiffrer la communication par radio entre le mobile et l'antenne-relais la plus proche.

RC4, le plus répandu, conçu par Ronald Rivest, utilisé notamment par le protocole WEP, un algorithme récent de Eli Biham – E0 utilisé par le protocole Bluetooth. [7]

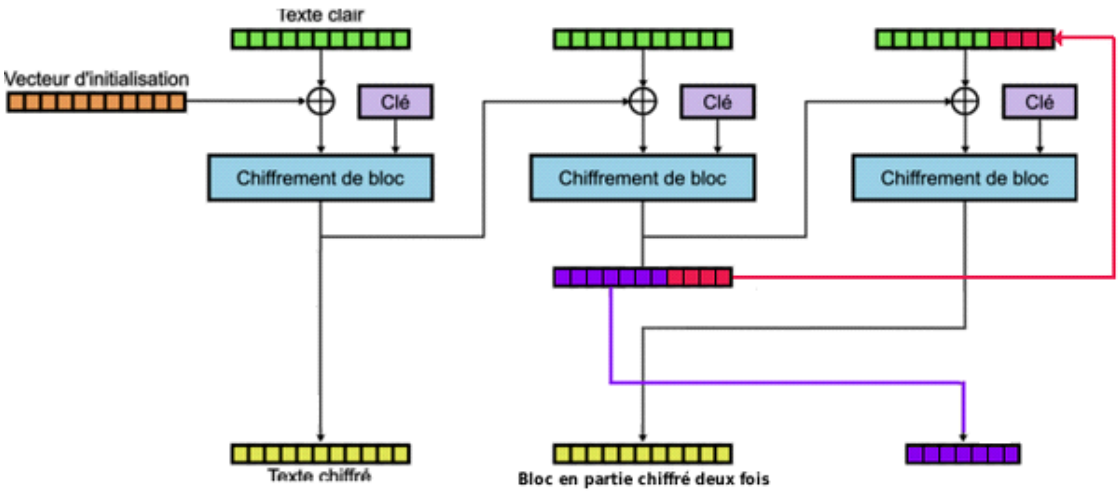


Figure II.3 : Chiffrements de flux (Stream Cipher)

II.2.2.4. DES (Data Encryption Standard)

Jusqu'aux années 1970, seuls les militaires possédaient des algorithmes à clé secrète fiables. Devant l'émergence de besoins civils, le NBS (National Bureau of Standards) lança le 15 mai 1973 un appel d'offres dans le Registre Fédéral (l'équivalent du Journal Officiel américain) pour la création d'un système cryptographique. Le cahier des charges était le suivant :

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

Les efforts conjoints d'IBM, qui propose Lucifer fin 1974, et de la NSA (National Security Agency) conduisent à l'élaboration du DES (Data Encryption Standard), L'algorithme de chiffrement le plus utilisé au monde durant le dernier quart du XXIe s. Le DES fut publié comme standard par le NBS le 15 janvier 1977.

Le DES a progressivement été abandonné à la fin des années 1990. Il a dans un premier temps été remplacé par le triple DES, qui consiste en trois applications de DES à la suite avec 2 clés différentes (d'où une clé de 112 bits).

Figure II.4 : schéma bloc DES

Si la sécurité était largement assurée, le triple DES était malheureusement trois fois plus lent que le DES. C'est pourquoi, en janvier 1997, le NIST (National Institute of Standards and Technologies) lance un nouvel appel pour créer un successeur au DES. Une nouvelle saga commence pour l'AES (Advanced Encryption Standard).

II.2.2.5. AES (Advanced Encryption System)

Avec le temps, et les progrès de l'informatique, les 256 clés possibles du DES n'ont plus représenté une barrière infranchissable. Il est désormais possible, même avec des moyens modestes, de percer les messages chiffrés par DES en un temps raisonnable. En janvier 1997, le NIST (National Institute of Standards and Technologies) des Etats-Unis lance un appel d'offres pour élaborer l'AES, Advanced Encryption System. Le cahier des charges comportait les points suivants :

- évidemment, une grande sécurité.
- une large portabibilité: l'algorithme devant remplacer le DES, il est destiné à servir aussi bien dans les cartes à puces, aux processeurs 8 bits peu puissants, que dans des processeurs spécialisés pour chiffrer des milliers de télécommunications à la volée.
- la rapidité.
- une lecture facile de l'algorithme, puisqu'il est destiné à être rendu public.

- techniquement, le chiffrement doit se faire par blocs de 128 bits, les clés comportant 128,192 ou 256 bits.

Au 15 juin 1998, date de la fin des candidatures, 21 projets ont été déposés. Certains sont l'œuvre d'entreprises (IBM,...), d'autres regroupent des universitaires (CNRS,...), les derniers sont écrits par à peine quelques personnes. Pendant deux ans, les algorithmes ont été évalués par des experts, avec forum de discussion sur Internet, et organisation de conférences. Le 2 octobre 2000, le NIST donne sa réponse : c'est le Rijndael qui est choisi, un algorithme mis au point par 2 belges, Joan Daemen et Vincent Rijmen. Depuis, le Rijndael, devenu AES, a été largement déployé et a remplacé progressivement le DES.

II.2.2.6. RC4 (RivestCipher 4)

RC4 est un algorithme de chiffrement à flot destiné aux applications logicielles. Il a été conçu par R. Rivest en 1987 pour les laboratoires RSA. Malgré un certain nombre de faiblesses, il est encore très utilisé aujourd'hui, notamment du fait de sa vitesse élevée (7 cycles par octet sur un Pentium III par exemple). Il est employé par exemple dans SSL/TLS, protocole permettant d'assurer la confidentialité des transactions Web, dans la norme de chiffrement WEP (Wired Equivalent Privacy) pour les réseaux sans fil, IEEE 802.11b.

L'algorithme a une longueur de clé variable (de 1 à 256 octets). Cependant à cause des lois d'exportation, la clé a souvent une longueur de 40 bits. La clé est utilisée pour initialiser une "table d'états" de 256 octets. La table d'état est employée pour la génération d'octets pseudo-aléatoires et ensuite pour produire le flux pseudo-aléatoire avec lequel le texte clair sera transformé avec l'opération de l'OÙ-Exclusif.

II.2.2.7. Cryptographie asymétrique

Le principe de base de cette méthode et que chaque personne dispose d'un jeu de clé comportant :

- Une clé privée : elle est unique et confidentielle, elle appartient exclusivement à l'hôte concerné, il ne la distribue à personne, aucun double de cette clé ne doit être crée.
- Une clé publique : elle est unique également. Mais tout le monde peut s'en procurer une copie, il suffit d'aller la chercher chez un dépositaire.dit « tiers de confiance » (c'est un organisme de réputation sérieuse, à qui l'on confiera sa clé publique). Il

s'agit en quelque sort d'un concierge qui garde ces clés publiques et certifie qu'elles appartiennent bien à la personne indiquée.

Ce qui est chiffré avec une clé publique ne peut être déchiffré qu'avec la clé privé correspondante, Ce qui est chiffré avec la clé privé ne peut être déchiffré qu'avec la clé publique correspondante.

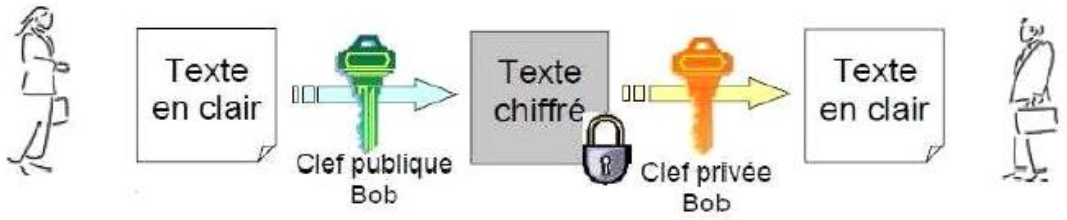


Figure II.5 : Chiffrements asymétrique

Un chiffrement asymétrique est défini par trois algorithmes

- Algorithme de génération des clés;
- Algorithme de chiffrement;
- Algorithme de déchiffrement ;

Certains algorithmes asymétriques comme RSA offre aussi des opérations pour la génération de signature numérique et sa vérification. L'utilisation d'une paire de clés publique/privée permet d'assurer la confidentialité, l'authentification, l'intégrité et l'échange de la clé secrète. Cependant les algorithmes asymétriques ne réalisent tous ces fonctions. [7]

II.2.2.8. RSA (Rivest, Shamir et Adleman)

La méthode de cryptographie RSA a été inventée en 1977 par Ron Rivest, Adi Shamir et Len Adleman, à la suite de la découverte de la cryptographie à clé publique par Diffie et Hellman. Le RSA est encore le système cryptographique à clé publique le plus utilisé de nos jours. Il est intéressant de remarquer que son invention est fortuite : au départ, Rivest, Shamir et Adleman voulaient prouver que tout système à clé publique possède une faille.

II.3. Fonctions de hachage

Une fonction de hachage est aussi appelée fonction de hachage à sens unique ou "one-way hash function" en anglais est une fonction qui associe à un grand ensemble de données un ensemble beaucoup plus petit .Le résultat de cette fonction est par ailleurs aussi appelé somme de contrôle, empreinte, résumé de message, condensé ou encore empreinte cryptographique lorsque l'on utilise une fonction de hachage cryptographique. Les fonctions de HASH peuvent être utilisées pour l'authentification de message, dérivation de clé, génération de nombre pseudo aléatoire et génération de la signature numérique. [7]

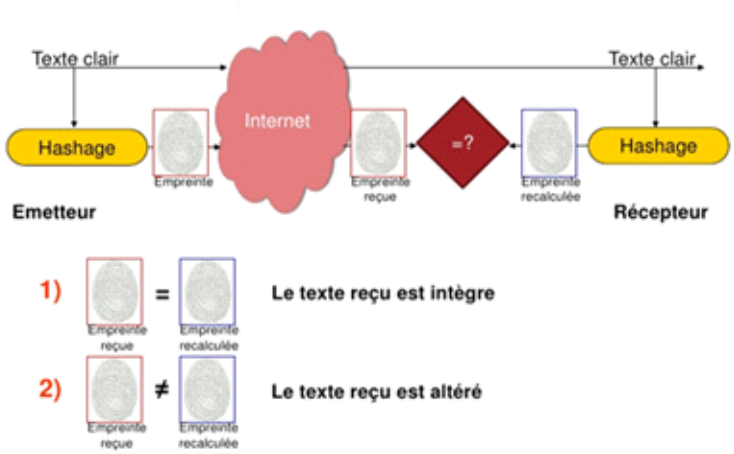


Figure II.6 : Fonctions de hachage

II.3.1. MD5 (MD signifiant Message Digest)

MD5 Développé par Rivest en 1991, MD5 crée une empreinte digitale de 128 bits à partir d'un texte de taille arbitraire en le traitant par blocs de 512 bits. Il est courant de voir des documents en téléchargement sur Internet accompagnés d'un fichier MD5, il s'agit du condensé du document permettant de vérifier l'intégrité de ce dernier).

II.3.2. Algorithme de hachage sécurisé SHA

L'algorithme SHA a été conçu par NSA, standardisé par NIST et publié comme FIPS PUB 180 en 1993. Une version révisée a été délivrée en 1995 comme FIPS PUB 180-1.

La version actuelle est FIPS PUB 180-2 avec une notice de changement pour SHA-224 en février 2004.

La liste des algorithmes SHA est : SHA-1, SHA-256, SHA-384 et SHA-512. A cause des calculs intensifs dans SHA-384 et SHA-512, ces deux algorithmes ne sont pas recommandés pour être utilisés sur les dispositifs mobiles.

SHA-1 est l'algorithme le plus utilisé dans la famille des algorithmes SHA. SHA-1 prend en entrée un message de taille inférieure à 2st bits et produit une empreinte de 160 bits. SHA-256 prend aussi en entrée un message de taille inférieure à 2st bits mais produit une empreinte de 256 bits. Il y a une nouvelle attaque sur SHA-1 sans utiliser la force brute, qui trouve des collisions à l'intérieur de 269 opérations hash.

A cause de l'avancement technologique et les attaques, NIST a planifié d'abandonner SHA-1 la fin 2010, et recommande des fonctions HASH plus fortes comme SHA-256.

Les algorithmes SHA-256, SHA-384 et SHA-512 sont relativement nouveaux et sont aussi standardisés par NIST mais ils sont plus intensifs en calcul et crypto analyses. [7]

II.3.3. Code d'authentification de message MAC

Un code d'authentification de message (MAC, Message Authentication Code) est un code accompagnant des données dans le but d'assurer l'intégrité de ces dernières, en permettant de vérifier qu'elles n'ont subi aucune modification, après une transmission par exemple.

Le concept est relativement semblable aux fonctions de hachage. Il s'agit ici aussi d'algorithmes qui créent un petit bloc authentificateur de taille fixe. La grande différence est que ce bloc authentificateur ne se base plus uniquement sur le message, mais également sur une clé secrète.

Tout comme les fonctions de hachage, les MAC n'ont pas besoin d'être réversibles. En effet, le récepteur exécutera le même calcul sur le message et le comparera avec le MAC reçu.

Le MAC assure non seulement une fonction de vérification de l'intégrité du message, comme le permettrait une simple fonction de hachage mais de plus authentifie l'expéditeur, détenteur de la clé secrète. Il peut également être employé comme un chiffrement supplémentaire (rare) et peut être calculé avant ou après le chiffrement principal, bien qu'il soit généralement conseillé de le faire avant. [7]

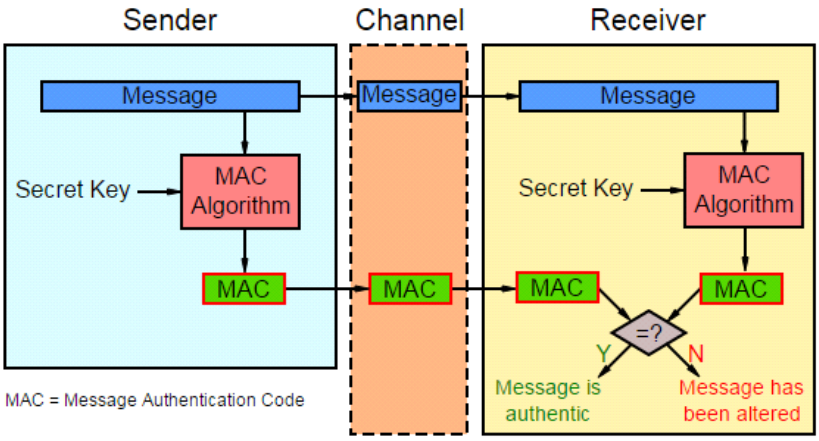


Figure II.7 : Code d'authentification de message MAC

II. 3.4. HMAC

Un HMAC, de l'anglais keyed-hash message authentication code (code d'authentification d'une empreinte cryptographique de message avec clé), est un type de code d'authentification de message (CAM), ou MAC en anglais (Message Authentication Code), calculé en utilisant une fonction de hachage cryptographique en combinaison avec une clé secrète. Comme avec n'importe quel CAM, il peut être utilisé pour vérifier simultanément l'intégrité de données et l'authenticité d'un message. N'importe quelle fonction itérative de hachage, comme MD5 ou SHA-1, peut être utilisée dans le calcul d'un HMAC ; le nom de l'algorithme résultant est HMAC-MD5 ou HMAC-SHA-1. La qualité cryptographique du HMAC dépend de la qualité cryptographique de la fonction de hachage et de la taille et la qualité de la clé.

Une fonction itérative de hachage découpe un message en blocs de taille fixe et itère dessus avec une fonction de compression. Par exemple, MD5 et SHA-1 opèrent sur des blocs de 512 bits. La taille de la sortie HMAC est la même que celle de la fonction de hachage (128 ou 160 bits dans les cas du MD5 et SHA-1), bien qu'elle puisse être tronquée si nécessaire.

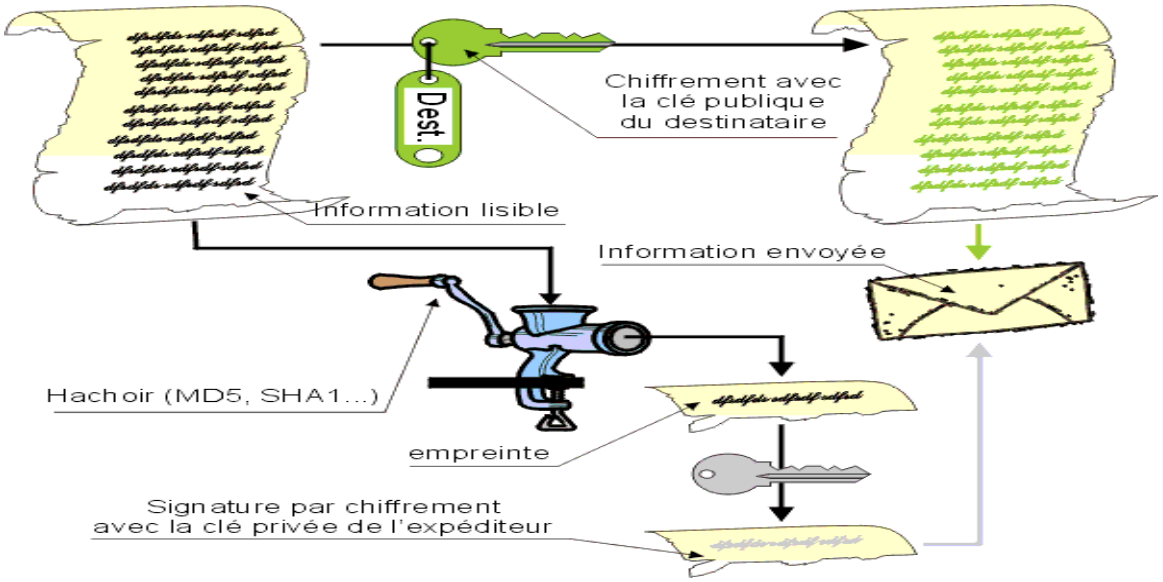


Figure II.8 : Code d'authentification d'une empreinte cryptographique de message avec clé

Autres fonctions MAC :

CBC-MAC est une méthode pour transformer un chiffrement de bloc en MAC. CBC-MAC est considérée sécurisée si le chiffrement utilisé est sécurisé. Cependant il est dangereux d'utiliser la même clé secrète pour le chiffrement CBC et l'authentification CBC-MAC.

UMAC est une fonction plus rapide que HMAC mais elle est limitée par une taille faible de l'empreinte (64 bits) et la complexité de ces différentes implémentations.

Donc HMAC est plus facile à utiliser et plus robuste comparé aux autres algorithmes MAC.

Il est recommandé d'utiliser HMAC comme fonction. [7]

II.4. Certificats

Les certificats permet d'associer une clé publique à une entité (une personne, une machine,….) afin d'en assurer la validité. Le certificat est en quelque sorte la carte d'identité de la clé publique, délivré par un organisme appelé autorité de certification (souvent notée CA pour Certification Authority).

L'autorité de certification est chargée de délivrer les certificats, de leur assigner une date de validité (équivalent à la date limite de péremption des produits alimentaires), ainsi que

de révoquer éventuellement des certificats avant cette date en cas de compromission de la clé(ou du propriétaire).

Les certificats sont des petits fichiers divisés en deux parties :

- La partie contenant les informations.
- La partie contenant la signature de l'autorité de certificat.

La structure des certificats est normalisée par le standard X509 de TUIT (plus exactement x509v3), qui définit les informations contenues dans le certificat : [7]

- **Version :** indique à quelle version de X509 correspond ce certificat
- **Numéro de série :** Numéro de série du certificat
- **Algorithme de signature:** identifiant du type de signature utilisée
- **Emetteur :** Distinguished Name (DN) de l'autorité de certification qui a émis ce certificat.
- **Valide à partir de:** la date de début de validité de certificat
- **Valide jusqu'à :** la date de fin de validité de certificat
- **Objet:** Distinguished Name (DN) de détenteur de la clef publique
- **Clé publique :** infos sur la clef publique de ce certificat
- **Contraintes de base :** extensions génériques optionnelles
- **Utilisation de la clé :** l'objet d'utilisation de la clé
- **Algorithme thumbprint :** alogrithme de signature
- **Thumbprint :** signature numérique de l'autorité de certification sur l'ensemble des champs précédents.

L'ensemble de ces informations (information+clé publique du demandeur) est signé par l'autorité de certification, cela signifie qu'une fonction de hachage crée une empreinte de ces informations, puis ce condensé est chiffré à l'aide de la clé privée de l'autorité de certification, la clé publique ayant été préalablement la règlement diffusée afin de permettre aux utilisateurs de vérifier la signature avec la clé publique de l'autorité de certification aux utilisateurs de vérifier la signature avec la clé publique de l'autorité de certification.

Exemple de certificat :

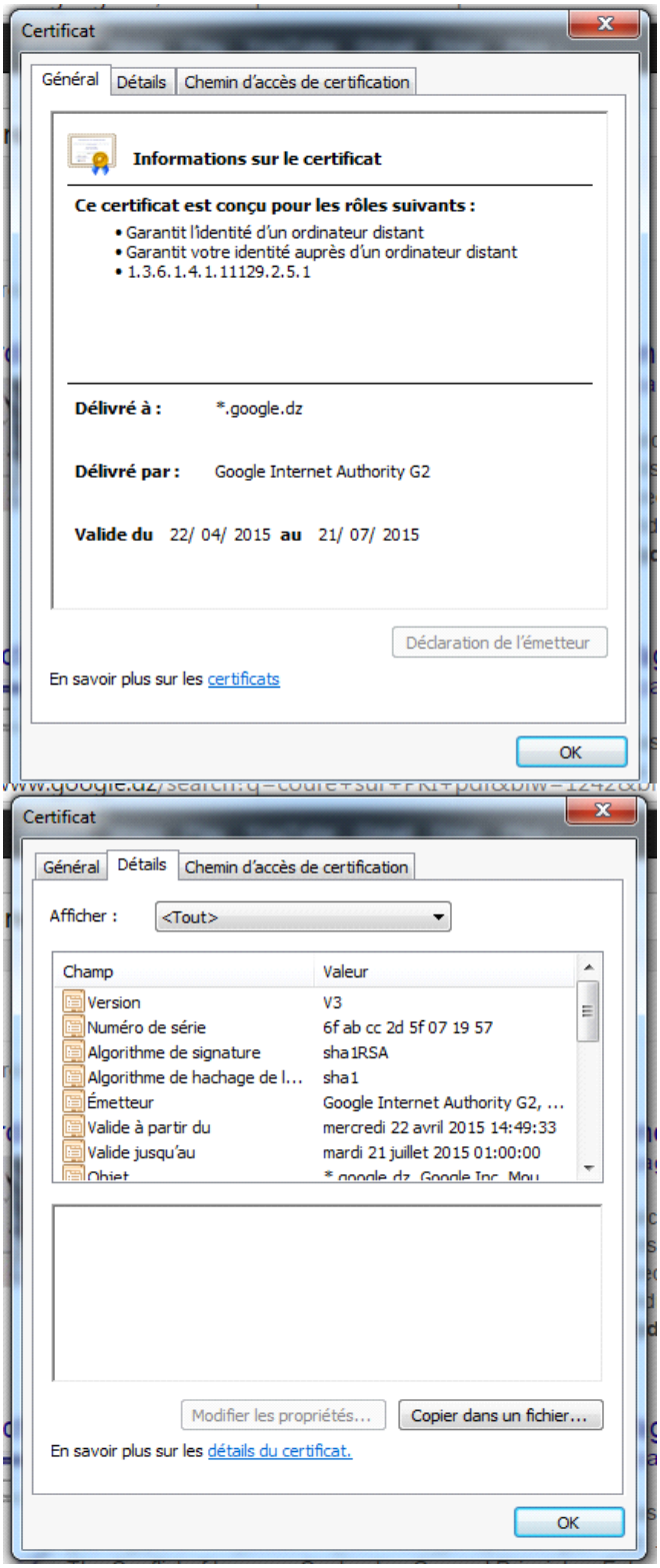


Figure II.9 : Certificat numérique

II.5. Gestion des clés avec PKI :

PKI (Public Key Infrastructure) est un système de gestion des clefs publiques qui permet de gérer des listes importantes de clefs publiques et d'en assurer la fiabilité, pour des entités généralement dans un réseau. Elle offre un cadre global permettant d'installer des éléments de sécurité tels que la confidentialité, l'authentification, l'intégrité et la non-répudiation tant au sein de l'entreprise que lors d'échanges d'information avec l'extérieur.

Une infrastructure PKI fournit donc quatre services principaux:

- fabrication de bi-clés.
- certification de clé publique et publication de certificats.
- Révocation de certificats.
- Gestion la fonction de certification.

Une infrastructure à clé publique utilise des mécanismes de signature et certifie des clés publiques qui permettent de chiffrer et de signer des messages ainsi que des flux de données, afin d'assurer la confidentialité, l'authentification, l'intégrité et la non-répudiation.

- **Signature**

Dans la signature nous avons une bi-clé : une clé (privé) pour la création de signature et une clé (publique) pour la vérification de signature, pour signer un message voici comment se passe:

1) A l'aide de la clé privée de signature de l'expéditeur, une empreinte connue sous le nom "message digest" est générée par hachage en utilisant l'algorithme SHA-1 ou MD5, le plus utilisé étant SHA-1. Cette empreinte est ensuite cryptée avec cette clé privée de signature.

2) On joint au message l'empreinte et le certificat contenant la clé publique de signature.
3) Le destinataire vérifie la validité du certificat et son non révocation dans l'annuaire.
4) Le destinataire transforme l'empreinte avec la clé publique de signature ainsi validée. Cette opération permet de s'assurer l'identité de l'expéditeur.

5) Ensuite le destinataire génère une empreinte à partir de message reçu en utilisant le même algorithme de hachage. Si les deux empreintes sont identiques, cela signifie que le message

n'a pas été modifié.Donc la signature vérifie bien l'intégrité du message ainsi que l'identité de l'expéditeur.Exemples d'algorithme de signature: RSA, DSA.[8]

Organisation d'une PKI :

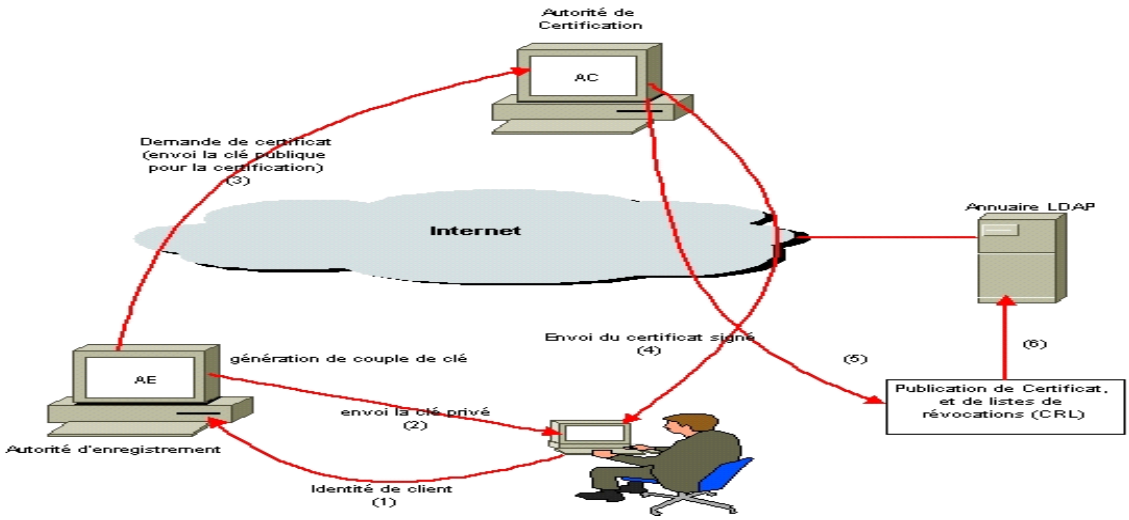


Figure II.10 : Organisation d'une PKI

II.6. Les outils de la sécurité

II.6.1. Pare-feu

Un pare-feu (appelé aussi coupe-feu, garde-barrière ou firewall en anglais), est un système permettant de protéger un ordinateur ou un réseau d'ordinateurs des intrusions provenant d'un réseau tiers (notamment internet). Le pare-feu est un système permettant de filtrer les paquets de données échangés avec le réseau, il s'agit ainsi d'une passerelle filtrante comportant au minimum les interfaces réseau suivante :

- une interface pour le réseau à protéger (réseau interne) ;
- une interface pour le réseau externe.

Le système firewall est un système logiciel, reposant parfois sur un matériel réseau dédié, constituant un intermédiaire entre le réseau local (ou la machine locale) et un ou plusieurs réseaux externes. Il est possible de mettre un système pare-feu sur n'importe quelle machine et avec n'importe quel système pourvu que :

- La machine soit suffisamment puissante pour traiter le trafic.

- Le système soit sécurisé.
- Aucun autre service que le service de filtrage de paquets ne fonctionne sur le serveur.

Dans le cas où le système pare-feu est fourni dans une boîte noire « clé en main », on utilise le terme d'« Appliance ».

Fonctionnement d'un système pare-feu

Un système pare-feu contient un ensemble de règles prédéfinies permettant :

- D'autoriser la connexion (allow) ;
- De bloquer la connexion (deny) ;
- De rejeter la demande de connexion sans avertir l'émetteur (drop).

L'ensemble de ces règles permet de mettre en œuvre une méthode de filtrage dépendant de la politique de sécurité adoptée par l'entité. On distingue habituellement deux types de politiques de sécurité permettant :

- soit d'autoriser uniquement les communications ayant été explicitement autorisées :
- soit d'empêcher les échanges qui ont été explicitement interdits.

La première méthode est sans nul doute la plus sûre, mais elle impose toutefois une définition précise et contraignante des besoins en communication. [9]

II.6.2. DMZ

Une zone démilitarisée (ou DMZ, de l'anglais demilitarized zone) est un sous-réseau séparé du réseau local et isolé de celui-ci et d'Internet par un pare-feu. Ce sous-réseau contient les machines étant susceptibles d'être accédées depuis Internet.

Le pare-feu bloquera donc les accès au réseau local pour garantir sa sécurité. Et les services susceptibles d'être accédés depuis Internet seront situés en DMZ.

En cas de compromission d'un des services dans la DMZ, le pirate n'aura accès qu'aux machines de la DMZ et non au réseau local.

Lorsque certaines machines du réseau interne ont besoin d'être accessibles de l'extérieur (serveur web, un serveur de messagerie, un serveur FTP public, etc.), il est souvent nécessaire de créer une nouvelle interface vers un réseau à part, accessible aussi bien du réseau interne que de l'extérieur, sans pour autant risquer de compromettre la sécurité de l'entreprise. On parle ainsi de « zone démilitarisé » (notée DMZ pour DeMilitarized Zone) pour désigner cette

zone isolée hébergeant des applications mises à disposition du public. La DMZ fait ainsi office de « zone tampon » entre le réseau à protéger et le réseau hostile.

Les serveurs situés dans la DMZ sont appelés « bastions » en raison de leur position d'avant poste dans le réseau de l'entreprise.
La politique de sécurité mise en œuvre sur la DMZ est généralement la suivante :

- Traffic du réseau externe vers la DMZ autorisé ;
- Traffic du réseau externe vers le réseau interne interdit ;
- Traffic du réseau interne vers la DMZ autorisé ;
- Traffic du réseau interne vers le réseau externe autorisé ;
- Traffic de la DMZ vers le réseau interne interdit ;
- Traffic de la DMZ vers le réseau externe refusé. [9]

II.6.3. IDS

Un système de détection d'intrusions (IDS, de l'anglais Intrusion Detection System) est un périphérique ou processus actif qui analyse l'activité du système et du réseau pour détecter toute entrée non autorisée et / ou toute activité malveillante. La manière dont un IDS détecte des anomalies peut beaucoup varier ; cependant, l'objectif principal de tout IDS est de prendre sur le fait les auteurs avant qu'ils ne puissent vraiment endommager vos ressources.

Les IDS protègent un système contre les attaques, les mauvaises utilisations et les compromis. Ils peuvent également surveiller l'activité du réseau, analyser les configurations du système et du réseau contre toute vulnérabilité, analyser l'intégrité de données et bien plus. Selon les méthodes de détection que vous choisissez de déployer, il existe plusieurs avantages directs et secondaires au fait d'utiliser un IDS.

II.6.3.1 Types d'IDS

Il est essentiel de comprendre ce qu'est un IDS et les fonctions qu'il offre pour pouvoir déterminer le type le plus approprié à inclure dans votre politique de sécurité informatique. Cette section couvre les concepts derrière les IDS, les fonctionnalités de chaque type d'IDS et l'apparition d'hybrides d'IDS qui emploient plusieurs techniques et outils de détection dans un paquetage.

Certains IDS sont basés sur les connaissances, qui par prévention alertent les administrateurs de sécurité avant qu'une intrusion ne se produise à l'aide d'une base de données d'attaques

courantes. Sinon, il existe les IDS comportementaux qui analysent toutes les utilisations de ressources pour toute anomalie, ce qui est en général un signe positif d'activité malveillante. Certains IDS sont des services autonomes qui fonctionnent en arrière-plan et écoutent de façon passive toutes les activités, enregistrant tous les paquets suspects provenant de l'extérieur. D'autres combinent des outils système standards, des configurations modifiées et une journalisation avec commentaires avec l'expérience et l'intuition de l'administrateur pour créer un outil de détection d'intrusions puissant. L'évaluation de nombreuses techniques de détection d'intrusions peut aider à trouver celle qui est la meilleure pour votre société.

Les types d'IDS les plus courants dans le domaine de la sécurité sont les IDS basés sur l'hôte et les IDS basés sur le réseau. Un IDS basé sur l'hôte est le plus complet des deux et implémente un système de détection sur chaque hôte individuel. Indépendamment de l'environnement réseau sur lequel se trouve l'hôte, il est toujours protégé. Un IDS basé sur le réseau canalise les paquets via un seul périphérique avant de les envoyer vers des hôtes spécifiques. Les IDS basés sur le réseau sont souvent considérés comme moins complets vu que de nombreux hôtes dans un environnement mobile font en sorte qu'il ne soit pas disponible, d'où un dépistage et une protection de paquets réseau inefficaces. [9]

II.7. Les protocoles de la sécurité

II.7.1. Protocole IPsec

IPSec (Internet Protocol Security, RFC 2401) est un protocole de la couche 3 du modèle OSI, tout comme IP; il fut à l'origine développé dans le cadre de la version future de ce dernier protocole, à savoir IPv6. Or, si IPSec existe depuis quelques temps déjà, IPv6 n'est quant à lui pas encore déployé à grande échelle, ce qui aurait put empêcher l'utilisation d'IPSec. Mais il n'en est rien puisque ce dernier a été porté vers la version actuelle d'IP (IPv4) et est maintenant disponible pour tous (les plus récents patchs de sécurité de Windows 2000 l'incluent, et des distributions libres comme FreeSwan pour Linux existent également).

II.7.1.1. Présentation générale du protocole IPsec

IPSec est un protocole destiné à fournir différents services de sécurité. Il propose ainsi plusieurs choix et options qui lui permettent de répondre de façon adaptée aux besoins des entreprises, nomades, extranets, particuliers, etc... Néanmoins, son intérêt principal reste sans conteste son mode dit de tunneling, c'est-à-dire d'encapsulation d'IP qui lui permet entre autres choses de créer des réseaux privés virtuels (ou VPN en anglais). Cette technologie à pour but d'établir une communication sécurisée (le tunnel) entre des entités éloignées,

séparées par un réseau non sécurisé voir public comme Internet, et ce de manière quasi-transparente si on le désire. Dans l'article qui suit, c'est ce mode d'IPSec, le mode tunneling, qui sera décrit en priorité.

De manière succincte, citons quelques propriétés générales des tunnels destinés aux VPNs :

- ✓ Les données transitant sont chiffrées (confidentialité) et protégées (intégrité)
- ✓ Les 2 extrémités sont authentifiées
- ✓ Les adresses sources et destinations sont chiffrées, avec IPSec (IP dans IPSec)
- ✓ Ils peuvent présenter, suivant le protocole, des qualités anti-rejeux ou empêcher les attaques type man-in-the-middle.

Il ne faut pas non plus négliger les aspects pratiques tels que la charge processeur due au chiffrement, le débit théorique possible, l'overhead induit et donc le débit effectif... De plus IPsec n'est pas le seul protocole permettant d'établir des tunnels, il en existe d'autres comme les "point-à-point" tel que L2TP, L2F ou encore PPTP qui peuvent induire un overhead non négligeable surtout lors d'encapsulations successives (L2TPoATM avec AAL5, L2TPoEthernet, L2TPoUDP...). Voir la fiche consacrée au tunneling pour plus de détails.

II.7.2. Protocol SSH (Secure Shell)

Secure Shell (SSH) est à la fois un programme informatique et un protocole de communication sécurisé. Le protocole de connexion impose un échange de clés de chiffrement en début de connexion. Par la suite, tous les segments TCP sont authentifiés et chiffrés. Il devient donc impossible d'utiliser un *sniffer* pour voir ce que fait l'utilisateur.

Le protocole SSH a été conçu avec l'objectif de remplacer les différents programmes rlogin, telnet, rcp, ftp et rsh.

II.7.2.1. Présentation du protocole SSH

Le protocole SSH existe en deux versions majeures : la version 1.0 et la version 2.0.

- La première version permet de se connecter à distance à un ordinateur afin d'obtenir un shell ou ligne de commande. Cette version souffrait néanmoins de problèmes de sécurité dans la vérification de l'intégrité des données envoyées ou reçues, la rendant vulnérable à des attaques actives. En outre, cette version implémentait un système sommaire de transmission de fichiers, et du *port tunneling*.

- La version 2 qui était à l'état de *draft* jusqu'en janvier 2006 est déjà largement utilisée à travers le monde.

Cette version est beaucoup plus sûre au niveau cryptographique, et possède en plus un protocole de transfert de fichiers complet, le SSH file transfer protocol.

Habituellement le protocole SSH utilise le port TCP 22. Il est particulièrement utilisé pour ouvrir un shell sur un ordinateur distant. Peu utilisé sur les stations Windows (quoiqu'on puisse l'utiliser avec PuTTY, mRemote, cygwin ou encore OpenSSH), SSH fait référence pour l'accès distant sur les stations Linux et Unix.

SSH peut également être utilisé pour transférer des ports TCP d'une machine vers une autre, créant ainsi un tunnel. Cette méthode est couramment utilisée afin de sécuriser une connexion qui ne l'est pas (par exemple le protocole de récupérations de courrier électronique POP3) en la faisant transférer par le biais du tunnel chiffré SSH.

Il est également possible de faire plusieurs sauts entre consoles SSH, c'est-à-dire ouvrir une console sur un serveur, puis, de là, en ouvrir une autre sur un autre serveur.

II.7.3. Protocole SSL (Secure Socket Layer) & TLS (Secure Socket Layer)

Transport Layer Security (TLS), et son prédécesseur Secure Sockets Layer (SSL), sont des protocoles de sécurisation des échanges sur Internet. Le protocole SSL était développé à l'origine par Netscape. L'IETF, en a poursuivi le développement en le rebaptisant Transport Layer Security (TLS). On parle parfois de SSL/TLS pour désigner indifféremment SSL ou TLS.

TLS (ou SSL) fonctionne suivant un mode client-serveur. Il permet de satisfaire aux objectifs de sécurité suivants :

- l'authentification du serveur ;
- la confidentialité des données échangées (ou session chiffrée) ;
- l'intégrité des données échangées ;
- de manière optionnelle, l'authentification du client (mais dans la réalité celle-ci est souvent assurée par le serveur).

Le protocole est très largement utilisé, sa mise en œuvre est facilitée du fait que les protocoles de la couche application, comme HTTP, n'ont pas à être profondément modifiés pour utiliser une connexion sécurisée, mais seulement implémentés au-dessus de SSL/TLS, ce qui pour HTTP a donné le protocole HTTPS.

Un groupe de travail spécial de l'IETF a permis la création du TLS et de son équivalent en mode UDP, le DTLS. Depuis qu'il est repris par l'IETF, le protocole TLS a connu trois versions, TLS v1.0 en 1999, TLS v1.1 en 2006 et TLS v1.2 en 2008.

II.8. Conclusion

Aujourd'hui il est devenu plus difficile de garantir la sécurité d'un système dans un réseau, pour cela les laboratoires de recherche essayent de développer de nouvelles techniques, dans ce chapitre nous avons présenté un aperçu des différentes solutions et caractéristique des mécanismes de sécurité.

Dans le chapitre suivant, nous allons étudier et décrire le fonctionnement d'une solution de sécurité la plus répondu ces dernière années SSL/TLS.

Table des matières

Chapitre III

Protocole SSL (Secure Socket Layer)/ TLS (Transport Layer Secure)

III.1. Introduction

SSL (Secure Socket Layer) est un protocole de sécurité qui permet d'établir des liaisons authentifiés et chiffrées sur un réseau ouvert tel qu'Internet. SSL facilite la sécurisation des communications sur le réseau en identifiant et en authentifiant les entités communicantes, ainsi que la confidentialité et l'intégrité des données transmises. Etant donné que le protocole SSL empêche l'écoute ou la falsification des informations envoyées sur le réseau, son utilisation est recommandée sur tous les réseaux sur lesquels transitent des informations confidentielles ou propriétaires.

Dans ce chapitre nous présentons une étude détaillée du dit protocole.

III.2. Historique

SSL fut conçu par Netscape pour sécuriser les transactions commerciales sur son navigateur Netscape Navigator 1.1, en remplaçant http par https. La première version, maintenant connue sous le nom de SSL 1.0, de 94 n'a pas été diffusée. La seconde version, SSL 2.0, a été diffusée en novembre 94 et une implémentation a été intégrée dans le navigateur en mars 95.

Peu après, une faille de sécurité dans SSL 2.0 a été mis en évidence, due au générateur de nombres pseudo-aléatoires utilisé, qui permettait de casser une connexion SSL en moins d'une heure. Netscape a développé ensuite une nouvelle version de SSL : SSL 3.0 qui est sortie fin 95.

En mai 96, l'Internet Engineering Task Force (IETF) a mis en place un groupe de travail Transport Layer Security (TLS) pour standardiser les protocoles du type SSL. TLS a été publié comme RFC 2246 en janvier 99, avec peu de différences par rapport à SSL 3.0. Ses spécifications sont rééditées en août 2008 dans la RFC 6101. TLS, comme SSL 3.0, intégrait un mécanisme de compatibilité ascendante avec les versions précédentes Puis, pour des raisons de sécurité, détaillées dans la RFC 6176 parue en 2011, la compatibilité de TLS avec SSL 2.0 fut abandonnée par la suite. [10]

III.3. Présentation des protocoles SSL (Secure Socket Layer)

SSL/TLS est un protocole qui prend place entre le protocole de la couche de transport et la couche application, comme le montre la figure.

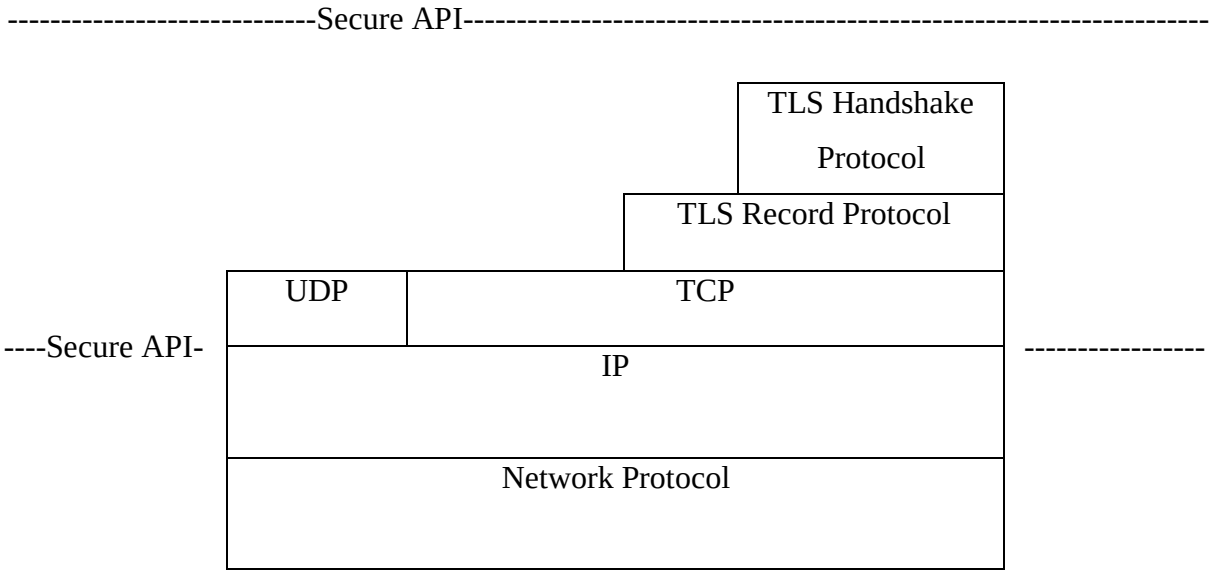


Figure III.1: Place du protocole SSL dans la suite TCP/IP

Le protocole SSL supporte l'utilisation d'un grand nombre d'algorithmes de cryptographie, ou de chiffrement, pour des opérations telles que l'authentification réciproque d'un serveur et d'un client, la transmission de certificat, et l'établissement d'une clef de session. Les clients et les serveurs peuvent supporter différentes suites de chiffrement, c'est à dire différents ensembles d'algorithmes, selon la version de SSL qu'ils intègrent, le protocole de négociation SSL

détermine comment serveur et client choisissent l'algorithme de chiffrement utilisé pour s'authentifier l'un à l'autre, pour transmettre des certificats et pour établir les clefs de session.
Les données qui vont et viennent entre le client et le serveur sont chiffrées à l'aide d'un algorithme symétrique tel que DES ou RC4. Un algorithme de clé publique -généralement RSA- est utilisé pour l'échange des clés de chiffrement et pour les signatures numériques. L'algorithme utilise la clé publique du certificat numérique du serveur. Ce dernier permet également au client de vérifier l'identité du serveur. Les versions 1 et 2 du protocole SSL offrent uniquement l'authentification serveur. La version 3 inclut l'authentification client, utilisant les certificats numériques du client et du serveur. [10]

III.4. Principe de fonctionnement de SSL (Secure Socket Layer)

Une session SSL commence lorsqu'une URL spéciale commençant par https est demandée, le client se connecte alors sur le port 443. Cette opération s'effectue de la façon suivante :
L'authentification du serveur SSL permet au client de confirmer l'identité du serveur : le logiciel côté client utilise la technique standard de cryptographie à clé publique pour vérifier que le certificat du serveur et son identifiant public, ont été produits par un organisme certificateur agréé par le client.
Cette confirmation est importante pour le client si par exemple, s'il envoie son N° de carte bancaire à travers le réseau et veut s'assurer de l'identité du serveur récepteur.
L'authentification du client SSL permet au serveur de confirmer l'identité du client : c'est le même procédé que celui utilisé pour l'authentification du serveur.
Cela peut être important dans le cas où le serveur est une banque, qui veut transmettre une information financière à son client et donc vérifier l'identité du destinataire.
Une connexion cryptée SSL implique que toutes les données échangées entre un client et un serveur soient cryptées par le logiciel expéditeur et décryptées par le logiciel destinataire, ce qui introduit un degré élevé de sécurité. Toutes les données envoyées au moyen d'une connexion cryptée SSL sont protégées par un mécanisme contre les manipulations, qui garantit que les données n'ont pas été modifiées dans leur transit. Le protocole SSL et TLS se décompose en deux couches principales (quatre en réalité) :

- Le protocole SSL handshake
- Le protocole SSL Change Cipher Spec

- Le protocole SSL Alert
- Le protocole SSL Record

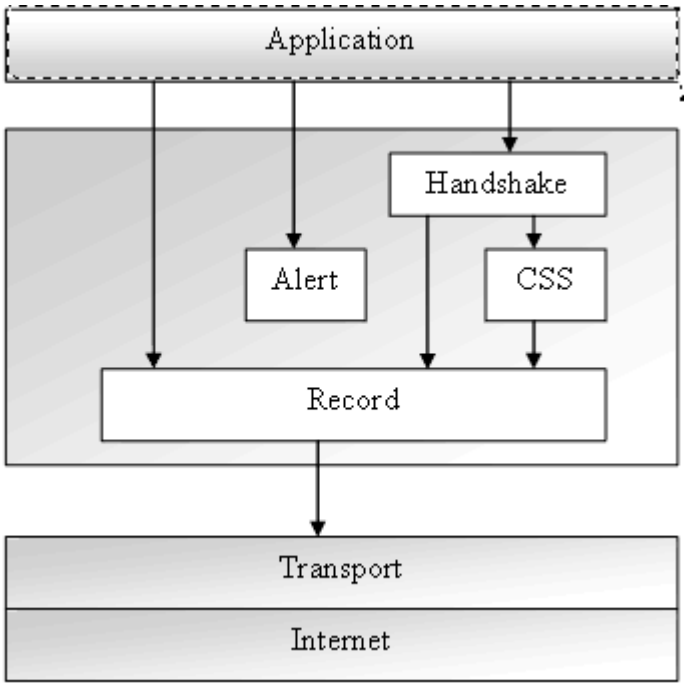


Figure III.2 : Le protocole SSL est constitué des sous protocoles

Le protocole SSL inclut deux sous protocoles :

- ❖ Le protocole SSL handshake : choisit la version de SSL et TLS qui sera utilisée, réalise l'authentification par l'échange de certificats et permet la négociation entre le client et le serveur d'un niveau de sécurité au travers du choix des algorithmes de cryptage. C'est le protocole de configuration de la transaction.
- ❖ Le protocole SSL record : encapsule et fragmente les données. C'est le protocole de transmission des données.

Le protocole SSL record définit le format utilisé pour transmettre les données.

Le protocole SSL handshake implique l'utilisation du protocole SSL record pour échanger une série de messages entre le serveur et le client lorsqu'ils établissent leur première connexion.

Cet échange de message sert à mettre en place les actions suivantes :

- ➢ Authentification du serveur par le client

- Permettre au client et au serveur de choisir les algorithmes de chiffrements qu'ils supportent conjointement
- Authentification du client par le serveur (optionnel)
- Utiliser les techniques de cryptage à clef publiques pour générer des secrets partagés
- Etablir une connexion SSL cryptée
- Le chiffrement utilisé avec SSL

Le protocole SSL supporte l'utilisation d'un grand nombre d'algorithmes de cryptage différents, notamment pour les opérations d'authentification, de transmission de certificats et d'établissement de clés de session.

Le client et le serveur peuvent supporter différents niveaux de chiffrements qui dépendent de la version SSL, des polices d'assurance qui imposent un niveau de cryptage acceptable et des dispositions législatives de chaque pays concernant l'exportation.

En tenant compte de ces informations, le protocole SSL détermine comment le serveur et le client vont négocier l'algorithme de chiffrement qu'ils vont utiliser pour s'authentifier mutuellement, transmettre les certificats et établir leur clé de session.

Les chiffrements qui suivent font référence aux algorithmes suivants :

- DES : Data Encryption Standard,
- DSA : Digital Signature Algorithm, algorithme utilise pour la signature numérique.
- KEA : Key Exchange Algorithme, pour échange de clés.
- MD5: Message Digest algorithm developed by Rivest.
- RC2 and RC4 : Algorithmes de chiffrement.
- RSA : Algorithme de clés publiques pour cryptage et authentification.
- RSA Key Exchange : Algorithme d'échange de clés pour SSL basé sur RSA.
- SHA-1 : Secure Hash Algorithm, fonction de hachage.
- Triple-DES : DES appliqué trois fois.

Les algorithmes d'échange de clés tels que KEA ou RSA Key Exchange régissent la façon dont le serveur et le client vont déterminer les clés symétriques qu'ils vont utiliser durant la session SSL.

Lors de l'échange d'information pendant le protocole SSL handshake, le client et le serveur identifient le niveau de chiffrement le plus élevé commun, c'est celui qu'ils utiliseront pour la session SSL.

Le niveau de chiffrement qu'une société met en œuvre dépend de plusieurs facteurs :

- la sensibilité des donnés impliquées
- la rapidité du chiffrement.
- l'application des lois à l'exportation.

III.5. Analyse de fonctionnement du protocole SSL (Secure Socket Layer)

Une connexion SSL est définie par les paramètres suivants partagés entre un client et un serveur :

❖ les paramètres suivants sont partagés entre un client et un serveur:

✓ **Session identifié** : un octet fixé par le serveur pour identifier la session

✓ **Peer certificat** : un certificat pour le serveur, éventuellement un autre pour le client

✓ **Cipher Spec** : définit l'algorithme de chiffrement symétrique et l'algorithme de condensation

✓ **Master secret** : clé de 48 octets négociée entre le serveur et le client

✓ **Compression method** : NULL pour l'instant (SSLv3/TLS)

✓ **Is resumable** : flag qui indique si de nouvelles connexions peuvent être créées à partir de cette session

❖ les paramètres suivants sont partagés entre un client et un serveur :

✓ **Server and client random** : des octets aléatoires déterminés par le client et le serveur pour chaque connexion

✓ **Server write (send) MAC secret** : clé secrète utilisée par le serveur pour faire les MAC

✓ **Client write (send) MAC secret** : clé secrète utilisée par le client pour faire les MAC

✓ **Server write (send) key** : clé symétrique utilisée par le serveur pour chiffrer les données

✓ **Client write (send) key** : clé symétrique utilisée par le client pour chiffrer les données

✓ **Initialization vectors** : pour un algorithme de chiffrement par bloc en mode CBC. Le premier est fixé lors du handshake, les suivants sont les derniers blocs des précédents fragments chiffrés

✓ **Sequence number** : chaque message est numéroté de part et d'autre

Ces paramètre sont réalise avec les sous protocole de SSL (Secure Socket Layer) comme au na site avant que le protocole SSL se constitué de deux sous protocole mais en réalité sont quatre. [11]

III.5.1. Le protocole SSL handshake : (protocole de négociation)

Ce protocole permet au client et au serveur de s'authentifier mutuellement, de négocie les algorithmes de chiffrement, de négocier les algorithmes de MAC et enfin de négocier les clés symétriques qui vont servir au chiffrement. Chaque message échangé entre le client et le serveur contient trois champs:

- *Type* : indique l'objet du message
- *Lenght* : indique la longueur du message
- *Content* : contient les données transmises

<table>
<tr><td>Content Type = 22 (Handshak)</td><td>Major Version =3</td><td>Minor Version = 1</td><td>Fragment Length</td></tr>
<tr><td>Fragment Length (continued)</td><td>Handshake Type</td><td colspan="2">Handshake Length</td></tr>
<tr><td>Handshake Length (continued)</td><td colspan="3" rowspan="2">Handshake Message</td></tr>
<tr><td colspan="2"></td></tr>
</table>

Figure III.3 : Paquet de négociation SSL

La négociation s'effectue de la façon suivante (figure 3.4) :

1. La phase de négociation est initiée par le client qui envoie un message du type «ClientHello». Ceci indique que le client désire établir une session SSL sur sa connexion TCP. Il envoie un nombre aléatoire (qui sera utilisé pour la détermination des clés), un identificateur de session et la liste des algorithmes de chiffrement et de compression qu'il est capable d'utiliser.

Les trois paramètres négociables sont la version de SSL/TLS, les algorithmes de sécurité et la méthode de compression.

2. Le serveur répond par une série des messages qui définissent les paramètres de sécurité du serveur :

- Il commence par un message du type Server Hello. Celui-ci fait savoir au client que le serveur a bien reçu son Client Hello (en envoyant le même numéro de session), envoie également un nombre aléatoire et indique l'algorithme de compression et l'algorithme de sécurité qu'il a choisi parmi ceux proposés par le client.

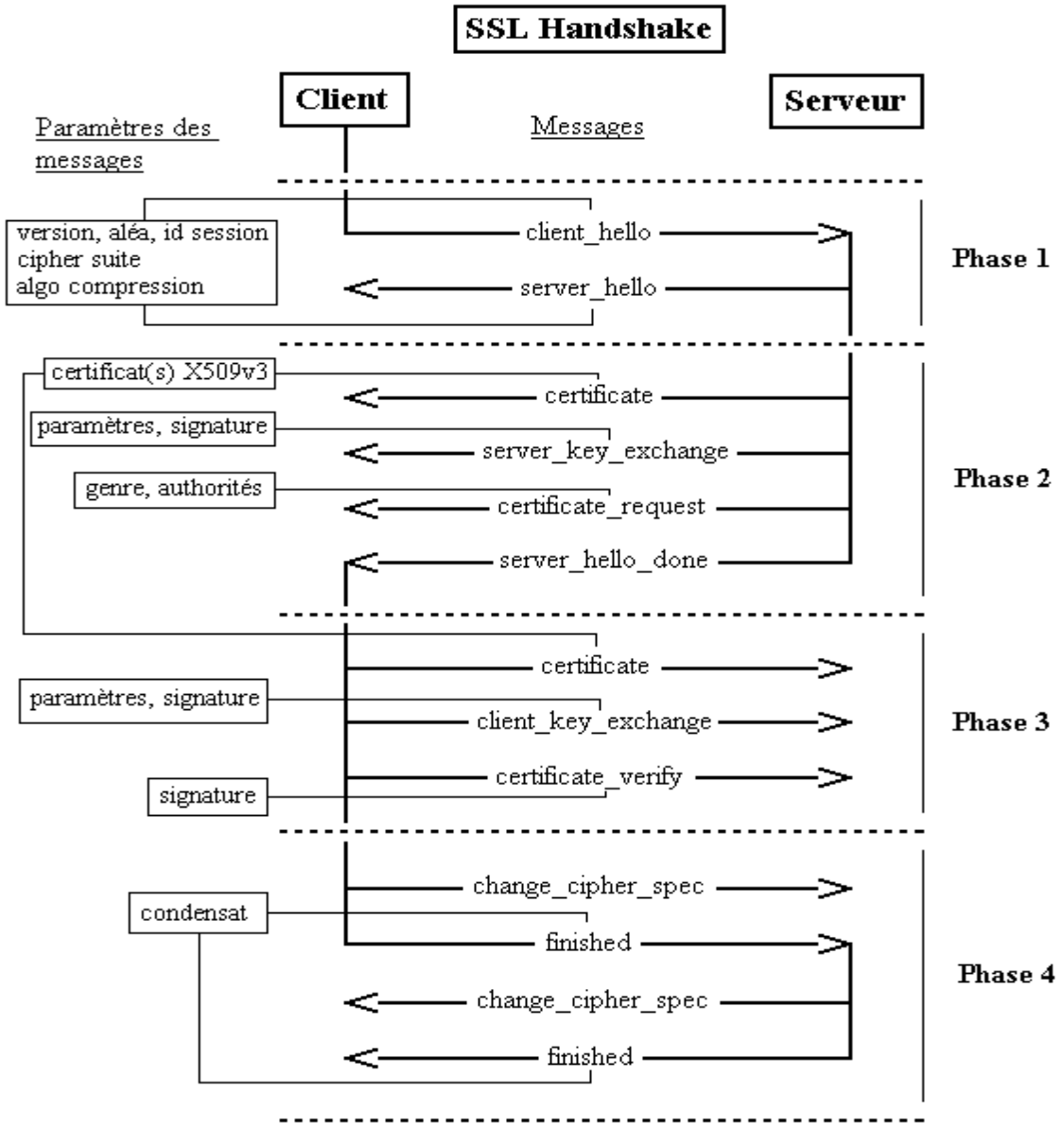


Figure III.4 : l'étape de négociation SSL

- Le serveur envoie ensuite un message du type « Certificate » pour s'identifier si le client la demande.

- Le serveur envoie alors un message du type « Server Key Exchange » pour transmettre les clés de sécurité si le client l'a demandé.

- Le serveur envoie éventuellement un message du type « Certificate Request », s'il désire que le client s'identifie.

- Le serveur indique enfin qu'il a terminé sa série de messages en envoyant un message du type « Server Hello Done ».

Ceci permet au client de savoir qu'il n'a pas à attendre un message du type « Certificate Request ».

3. Le client s'embarque alors dans une série de messages qui définissent les paramètres de sécurité du client :

- Le message « Certificate » identifie le client, uniquement en réponse à un « Certificate Request ».

- Le message « Client Key exchange » permet de transmettre les clés de sécurité.

- Le client confirme qu'il a accepte le certificat envoyé par le serveur (si c'est le cas) en envoyant un message du type « Certificate Verify ».

- Le client envoie alors un message du type « Change Cipher Specification » pour indiquer que les messages suivants seront cryptés.

- Le client termine par un message (crypté) du type « Finished ».

4. Le serveur répond en envoyant :

- un message du type « Change Cipher Specification ».
- suivi d'un message du type « Finished ».

L'ordre des messages est imposé. La suite de message montrée à la figure 3.4 peut être réinitialisée durant la session TCP sécurisée pour renégocier les paramètres de sécurité. Maintenant les données peuvent être échangées. [11]

III.5.1.1. Message Client Hello

Les paramètres du premier message, clienthello, envoyé par le client, sont :

- **Version :** la plus haute version de SSL que puisse utiliser le client.

- **Random :** un horodatage de 32 bits et une valeur aléatoire de 28 octets générée par le client. Ces valeurs servent de sel et sont utilisées pour se prémunir contre les rejets de paquets.
- **Session ID :** Un nombre, qui identifie la connexion. Un zéro signifie la volonté du client d'établir une nouvelle connexion sur une nouvelle session. Un autre nombre signifie la volonté de changer les paramètres ou de créer une nouvelle connexion sur la session existante.
- **CipherSuite :** Une liste, par ordre décroissant de préférence, des algorithmes que supporte le client. Il s'agit des algorithmes d'échange de clé et de chiffrement.
- **Key Exchange :** RSA clé symétrique chiffrée avec clé RSA contenue dans un certificat.
- **Diffie-Hellman :** si le certificat serveur contient des paramètres D-H. Le client n'est pas obligé de présenter un certificat si l'authentification du client n'est pas demandée par le serveur.
- **Diffie-Hellman temporaire :** Une clé symétrique D-H est négociée, les transactions étant signées par des clés RSA ou DSS certifiée par un CA (Certificate). C'est la plus sûre des méthodes parce que la clé générée est unique et authentifiée.
- **Diffie-Hellman anonyme :**(Un simple échange D-H est effectué, sans authentification des partenaires. Vulnérable à "l'attaque du milieu".

<table>
<tr><td>Type =1
(Client Hello)</td><td colspan="2">Message Length</td></tr>
<tr><td>Client Major Ver=3</td><td>Client Minor Ver=1</td><td>Timestamp</td></tr>
<tr><td colspan="2">Timestamp(continued)</td><td></td></tr>
<tr><td colspan="3">28 bytes Random Number</td></tr>
<tr><td colspan="3">32 byte of Session Identifier</td></tr>
<tr><td colspan="2">First Cypher Suite ID</td><td></td></tr>
<tr><td colspan="3">Other Cypher Suite IDs</td></tr>
<tr><td>First Method of Compression</td><td colspan="2">Other Compression Methods</td></tr>
</table>

Figure III.5 : Structure d'un message Client Hello

Après avoir envoyé ces requêtes, le client attend que le serveur réponde en générant une valeur aléatoire, en indiquant la version, et les meilleurs algorithmes qu'il peut utiliser: ce sera la réponse server_hello du serveur. [10]

III.5.1.2. Message Server Hello

Le serveur envoie au client sa version SSL, le chiffrement disponible, une donnée générée aléatoirement.

<table>
<tr><td>Type =2
(Server Hello)</td><td colspan="3">Message Length = 75</td></tr>
<tr><td>Client Major Ver=3</td><td>Client Minor Ver=1</td><td colspan="2">Timestamp</td></tr>
<tr><td colspan="2">Timestamp(continued)</td><td colspan="2" rowspan="2"></td></tr>
<tr><td colspan="2">28 bytes Random Number</td></tr>
<tr><td colspan="4">32 byte of Session Identifier</td></tr>
<tr><td colspan="2">Cypher Suite ID</td><td>Compression Method</td><td></td></tr>
</table>

Figure III.6 : Structure d'un message Server Hello

Lors de cette phase, le serveur commence par envoyer son certificat. Ce message peut contenir une chaîne de certificats. Le serveur envoie un message server_key_exchange. Les signatures sont effectuées en utilisant les fonctions de condensation, comprennent les sels initiaux et sont chiffrées grâce aux clés privées (ceci assure que le serveur détient la clé privée associée à la clé publique que contient le certificat).

<table>
<tr><td>Type =11</td><td>Message Length</td></tr>
<tr><td colspan="2">Sender's X509v3 Certificat (optional)</td></tr>
<tr><td colspan="2">Subsequent X.</td></tr>
</table>

Figure III.7 : Structure d'un message de Certificat

Ensuite, le serveur peut demander au client un certificat: c'est le message certificate_request.[10]

III.5.1.3. Message « Server Hello Done »

Il comprend l'en-tête SSL, suivi de l'en-tête des messages de négociation, dont le type est égal à 14 et la longueur du message toujours égale à 4.

Type =14 (Server Hello Done)	Message Length = 4

Figure III.8 : Structure d'un message Server Hello Done

Le serveur envoie le message server-done, qui signifie la fin de requête de serveur et que le serveur se met en attente.

Le client doit vérifier que le certificat envoyé par le serveur est valide, et que les autres paramètres sont corrects. Si le serveur a demandé au client d'envoyer un certificat, le client envoie un message certificate, contenant le certificat (s'il n'a pas de certificat, il envoie un message no_certificate).

Le client envoie ensuite un message client_key_exchange, qui dépend de l'algorithme choisi :
RSA : le client génère une clé secrète de 48 octets qui servira à générer la clé définitive, et la chiffre avec la clé publique du serveur.
D-H anonyme ou temporaire : ce sont les paramètres D-H du client qui sont envoyés.
Diffie-Hellman : Dans ce cas les paramètres D-H ont été envoyés avec le certificat, et ce message est donc vide. [10]

III.5.1.4. Message client Key Exchange

Il comprend l'en tète SSL, suivi de l'en-tête des messages de négociation, dont le type est égal à 16 pour un client et à 12 pour un serveur.

Type =12/16 (Key Exchange)	Message Length
Key Exchange Algorithm	Key Parameters

Figure III.9 : Structure d'un message Key Exchange

Pour finir cette connexion, le client envoie un message certificate_verify (sauf si le certificat ne contient que des paramètres D-H, qui ne peut servir à signer).
Le message consiste en un condensat des messages échangés pendant le handshake, et de la clé symétrique générée, le tout signé avec la clé privée du client. Le but de ce message est de s'assurer que le client est bien en possession de la clé privée correspondant à la clé publique envoyée dans le certificat.

III.5.1.5. Message « Change Cipher Spec »

Un message Change Cipher Spec est constitué d'en-tête SSL, dont le type est égal à 20, suivi des données sur un octet, dont la valeur n'a pas d'importance et qui est en général égal à 1.

Ce message indique que l'expéditeur utilisera désormais les paramètres négociés (algorithmes de sécurité, clés et méthode de compression) dans les messages futurs.

III.5.1.6 Message « Finished »

Le message Finished est le premier message qui apparait sous forme cryptée.

III.6. Le protocole SSL Alert

Ce protocole spécifie les messages d'erreur que peuvent s'envoyer clients et serveurs. Les messages sont composés de 2 octets, le premier est soit warning soit fatal. Si le niveau est fatal, la connexion est abandonnée. Les autres connexions sur la même session ne sont pas coupées mais on ne peut pas en établir de nouvelles.

Le deuxième octet donne le code d'erreur.

Les erreurs fatales sont:

- **unexpected_message** : message non reconnu
- **bad_record_mac** : MAC non correct
- **decompression_failure** : la fonction de décompression à reçu une mauvaise entré**e**
- ***handshake_failure*** : impossible de négocier les bons paramètres
- **illegal_parameter** : un champs était mal formaté ou ne correspondait à rien

Les warnings sont:

- **close_notify** : annonce la fin d'une connexion
- **no_certificate** : réponse à une demande de certificat s'il n'y en a pas.
- **bad_certificate** : le certificat reçu n'est pas bon (e.g. signature non valide)
- **unsupported_certificate** : le certificat reçu n'est pas reconnu...vive la compatibilité:)
- **certificate_revoked** : certificat révoqué par l'émetteur
- **certificate_expired** : certificat expiré
- **certificate_unknown** : tous les problèmes concernant les certificats et non listés au dessus. [11]

III.7. Le protocole SSL Record

Ce protocole permet de garantir:

- La confidentialité des données transmises (c'est le Handshake qui permet de négocier une clé symétrique partagée)
- L'intégrité des données transmises (c'est encore le Handshake qui négocie une clé partagée qui sert à faire des MAC sur les données)

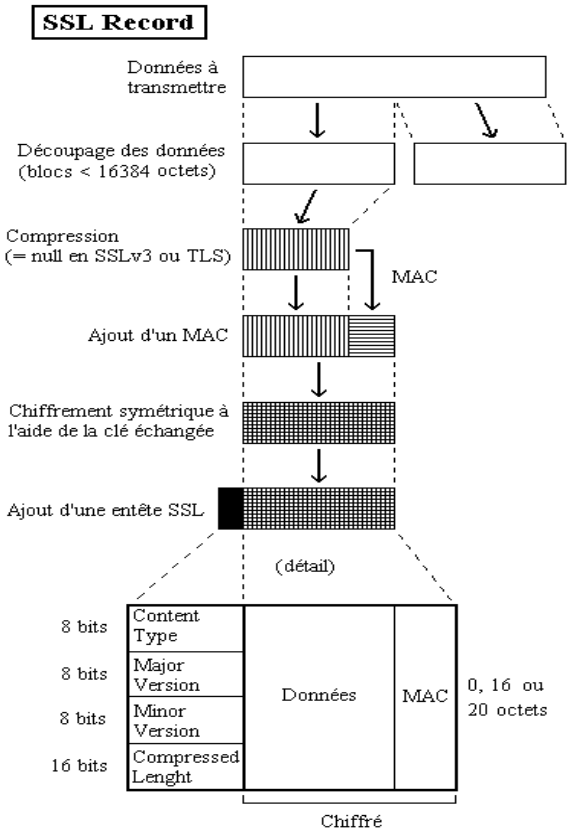


Figure III.10 : Fonctionnement du protocole Record

Dans l'entête qui est ajouté:

- *Content Type* : le plus haut protocole utilisé pour traiter ce fragment
- *Major Version* : plus haute version de SSL utilisée (3 pour SSLv3)
- *Minor Version* : plus basse version utilisée (0 pour SSLv3)
- *Compressed Lenght* : la longueur en octets des données (éventuellement compressées) à chiffrer. [11]

III.8. Conclusion

Dans ce chapitre, nous avons étudié une solution de sécurité parmi les plus répandues actuellement, en l'occurrence le protocole SSL/TLS déployé pour sécuriser les transactions bancaires, le e-commerce ainsi que les communications entre browsers et serveurs. Il s'agit d'un protocole qui est déployé à grande échelle car se trouvant déjà sur la plupart des plates-formes de la toile, aussi bien au niveau des serveurs qu'au niveau des clients.

Afin de pouvoir comprendre son fonctionnement, le protocole a été analysé en détail. Ceci nous aidera dans l'implémentation de la solution envisagée dans ce mémoire.

Table des matières

Chapitre IV

Implémentation du protocole SSL \TLS avec Openssl

IV.1. Introduction

Ce chapitre est consacré à l'implémentation d'une solution SSL visant à sécuriser les communications entre un client et serveur web. Pour ce faire, nous allons faire une émulation du réseau internet par un réseau local. La solution sera testée donc localement, mais sa validité sur le réseau internet va de soi.

En utilisant des certificats numériques qui assurent les services d'identification et d'authentification, des algorithmes de cryptage pour assurer la confidentialité des données, et des fonctions de hachages pour assurer l'intégrité, nous sommes en mesure de garantir la protection des communications par le biais de SSL/TLS même en milieu hostile.

Dans ce qui suit nous allons présenter les différentes étapes réalisées pour l'aboutissement de ce travail d'implémentation.

IV.2. Présentation des outils utilisés

En raison d'absence d'équipements pour tester la solution en réseau local, nous avons opté pour un environnement virtuel, pour cela nous avons utilisé les éléments suivants :

- Vmawre Workstation 10
- Ubunutu 14.04 LTS
- Apache2
- Openssl
- Windows 7

IV.2.1. PrésentationVmawre Workstation

VMware Workstation constitue pour les développeurs, testeurs et autres professionnels de l'informatique un puissant logiciel de création et d'utilisation de machines virtuelles qui permet d'exécuter plusieurs systèmes d'exploitation simultanément sur un même poste de

travail. Les utilisateurs peuvent exécuter Windows, Linux, NetWare ou Solaris x86 dans des machines virtuelles sécurisées et transportables. VMware Workstation offre des performances sans précédent et des fonctionnalités avancées telles que l'optimisation de la mémoire et la capacité de gérer des configurations.

VMware Workstation permet aux administrateurs système et autres professionnels de l'informatique de tester les logiciels, les nouvelles applications, les mises à jour et les correctifs d'OS dans des machines virtuelles, et ce en amont du déploiement en environnement de production sur PC ou serveurs physiques.

Les services Hotline peuvent créer et référencer une bibliothèque virtuelle des différentes configurations d'utilisateurs, pour contribuer à une résolution plus rapide des problèmes identifiés.

IV.2.2. Présentation d'Apache 2

Apache ou Apache HTTP Server est un programme libre mais sous licence Apache, le projet est déjà bien avancé puisqu'il a maintenant près de 20 ans et est disponible en version 2. C'est le serveur web le plus répandu dans le monde. Le serveur Apache a pour fonction d'envoyer une page statique Web au client qui en a fait la demande. La page statique web est identifiée par un URL côté serveur. Le client est identifié par une adresse IP unique. On appelle page statique, comparé aux écrans de gestion, une page qui ne contient aucune donnée variable mais des constantes, des images ou vidéos qui sont des fichiers Stream statiques de toute nature (suite d'octets en ASCII). On appelle page dynamique, une page qui contient des données variables issues généralement de fichiers. Un site internet est typiquement composé de pages statiques. Un relevé mensuel de banque en ligne est composé essentiellement de données dynamiques issues de fichiers. Le serveur Apache ne sait qu'envoyer des pages statiques à celui qui en a fait la demande via un URL

IV.2.3. Présentation d'openssl

openSSL est une boîte à outils cryptographiques implémentant les protocoles SSL et TLS qui offre :

- une bibliothèque de programmation en C permettant de réaliser des applications client/serveur sécurisées s'appuyant sur SSL/TLS.
- une commande en ligne (openssl) permettant

- La création de clés RSA, DSA (signature)
- La création de certificats X509
- Le calcul d'empreintes (MD5, SHA, etc, ...)
- Le chiffrement et déchiffrement (DES, IDEA, RC2, RC4, etc, ...)
- La réalisation de tests de clients et serveurs SSL/TLS
- La signature et le chiffrement de courriers (S/MIME)

IV.3. Implémentation SSL avec openssl

En pratique, les certificats présentés dans le fonctionnement du protocole SSL sont délivrés par une autorité de certification (CA) tierce reconnue par les entités impliquées dans l'échange d'informations. Malheureusement, ce service de certification est payant.

Dans notre travail, nous allons créer notre propre CA qui va générer des certificats pour l'ensemble des entités du réseau.

IV.3.1. Installation d'une machine virtuelle

La première étape dans ce projet est d'installer une machine virtuelle, nous avions le choix entre la Vmware Workstation ou Oracle VM VirtualBox.

Dans ce projet nous avons choisi la Vmawre Workstation, bien développée et facile à manipuler, disponible sur le lien suivant : https://www.vmware.com/go/downloadworkstation-fr

IV.3.2. Installation de l'Ubunutu 14.04 LTS

La deuxième étape consiste à télécharger le système ubuntu 14.04 LTS sous forme image (.ISO), disponible sur le lien suivant : https://www.ubuntu-fr.org/telechargement

On a choisit ce système comme un serveur Web sur notre réseau local.

IV.3.3. Installation de l'Apache2

La troisième étape consiste à télécharge et installer un serveur web qui permet de gérer les demandes d'un client. Pour ce la nous avons choisi Apache2 une version qui contient l'outil de SSL, et compatible avec Ubuntu 14.04 LTS.

- tout d'abord pour télécharge et installer l'Apache2 il faut ouvrir le terminal sous Ubuntu 14.04 LTS et écrire la commande suivant :

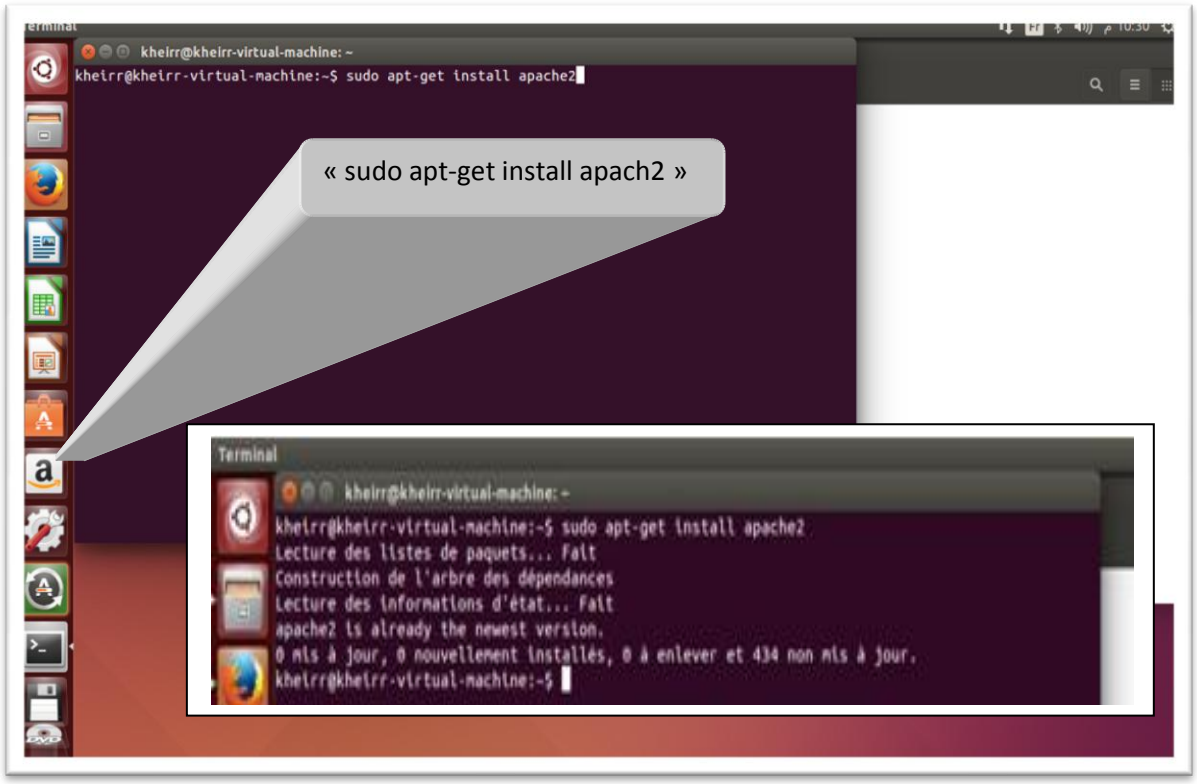


Figure IV.1 : téléchargement et installation de l'Apache2

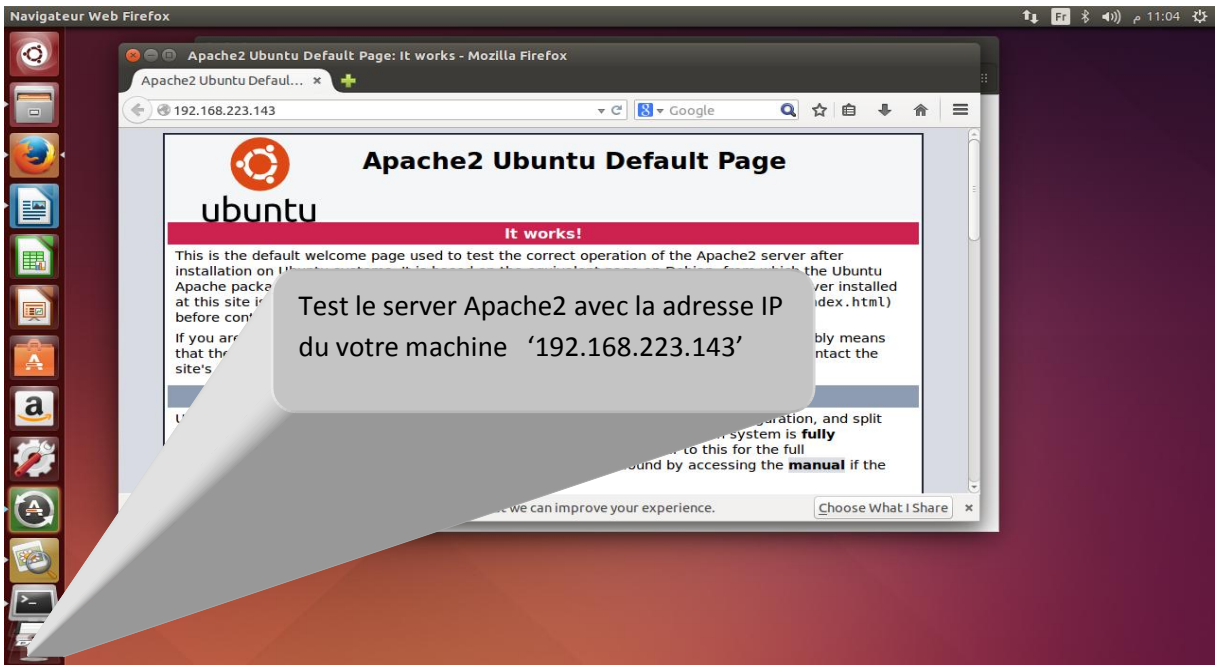


Figure IV.2 : test le fonctionnement du serveur Apache2

- Créer sous le répertoire etc/apache2 un répertoire ssl pour enregistrer les certificats et les clés privé et publique, utilisant la commande :

Figure IV.3 : création d'un fichier SSL

IV.3.4. Génération du certificat

Comme l'outil « openssl » et implémente dans la bibliothèque du système de « ubunutu 14.04 LTS » alors an ce lance directement au terminal juste il faut l'appelé avec la commande « openssl »

- Dans cette étape nous avons choisi de générer une clé privée avec l'algorithme de chiffrement RSA de 2048 bits avec le nom 'certificat' pour la demande de certificat avec la commande :

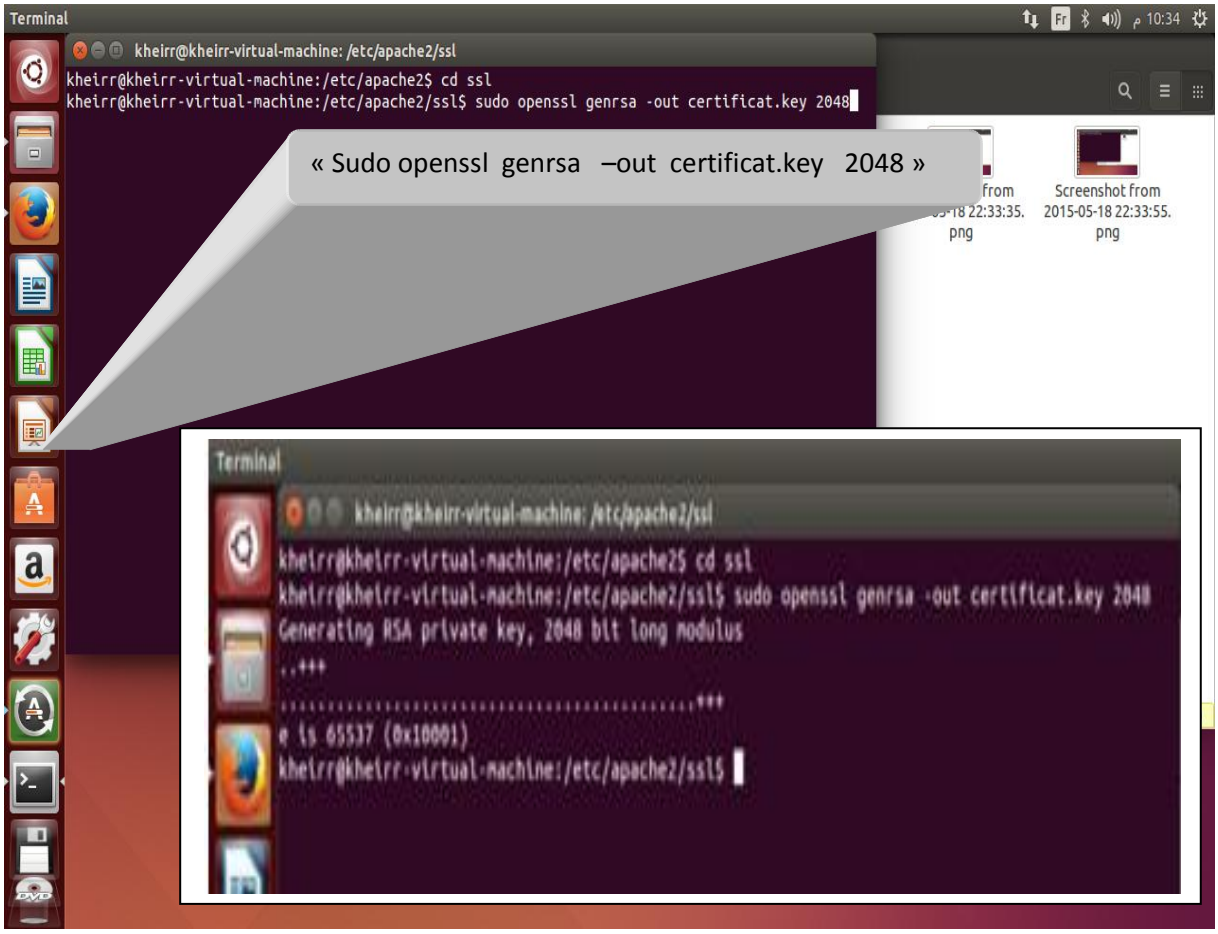


Figure IV.4 : génération d'une clé privée avec l'algorithme de chiffrement RSA

IV.3.5. Stockage de la partie publique dans un fichier à part

Comme on utilise l'algorithme hybride (bi clé) dans SSL alors la clé privé contient la clé publique pour les séparer nous avons utilisé la commande suivante :

« sudo openssl rsa –in certificat.key -pubout -out certificat.pub »

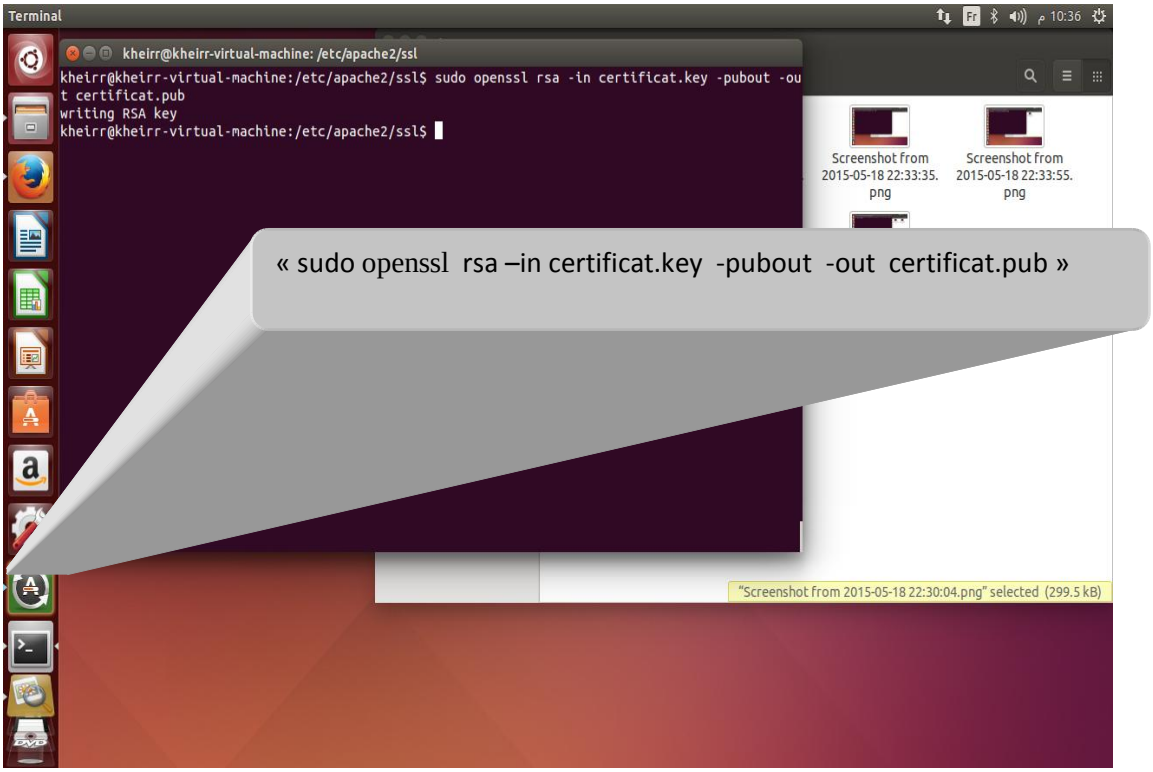


Figure IV.5 : Stockage de la partie publique dans un fichier à part

IV.3.6. Création du certificat serveur

- Donc ici on arrive à la génération de la demande du certificat qui sera signé par le CA (Certificate Autority), ou comme avec notre cas par un certificat auto signé.
- Rappelons juste que nous travaillons sur le répertoire /etc/apache2/ssl, donc avec la commande suivante nous allons créer une demande de certificat

« sudo openssl req -new -key certificat.key -out certificat.csr »

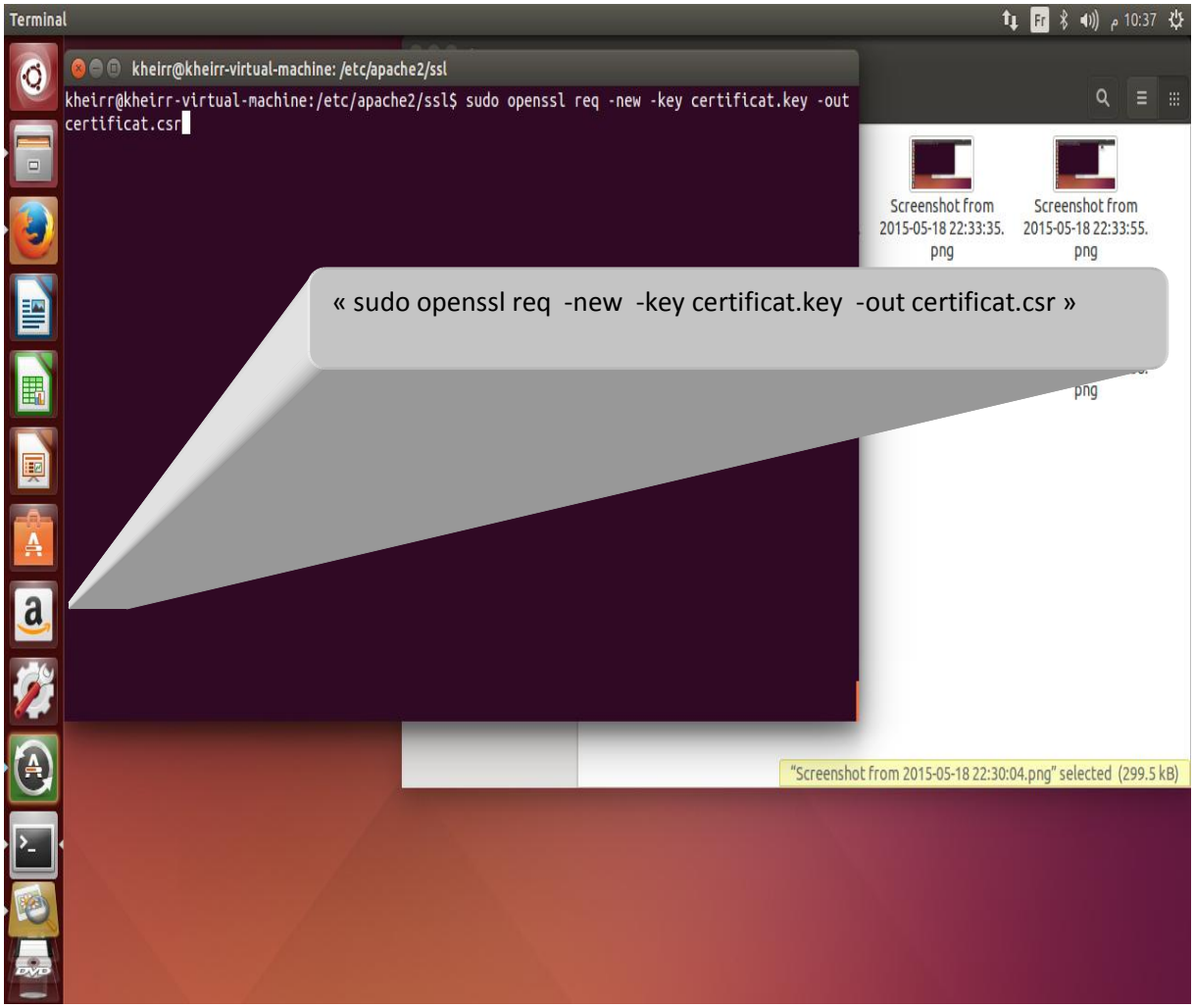
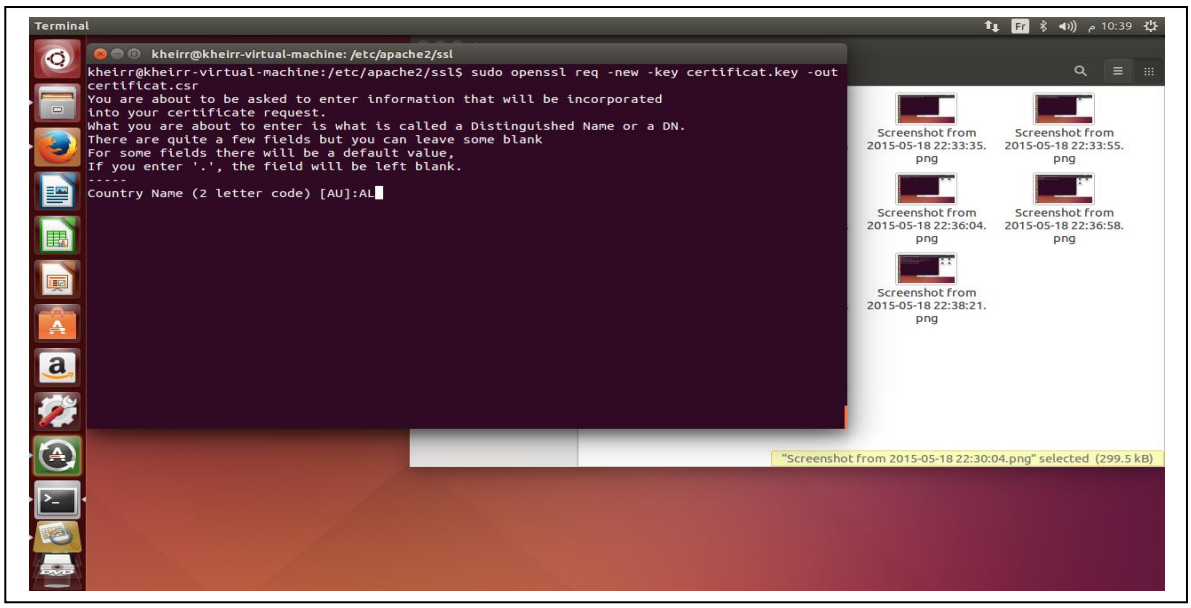
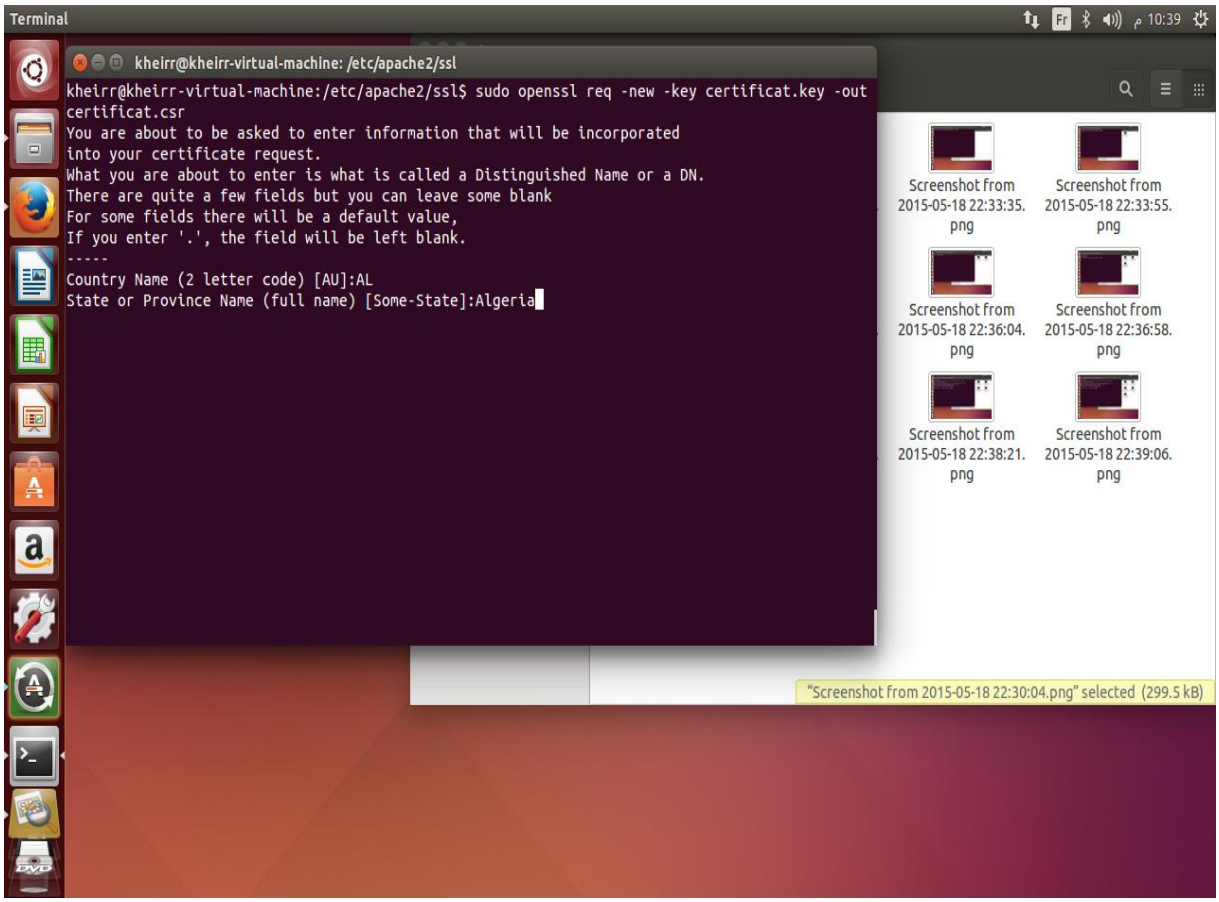


Figure IV.6 : génération de la demande du certificat

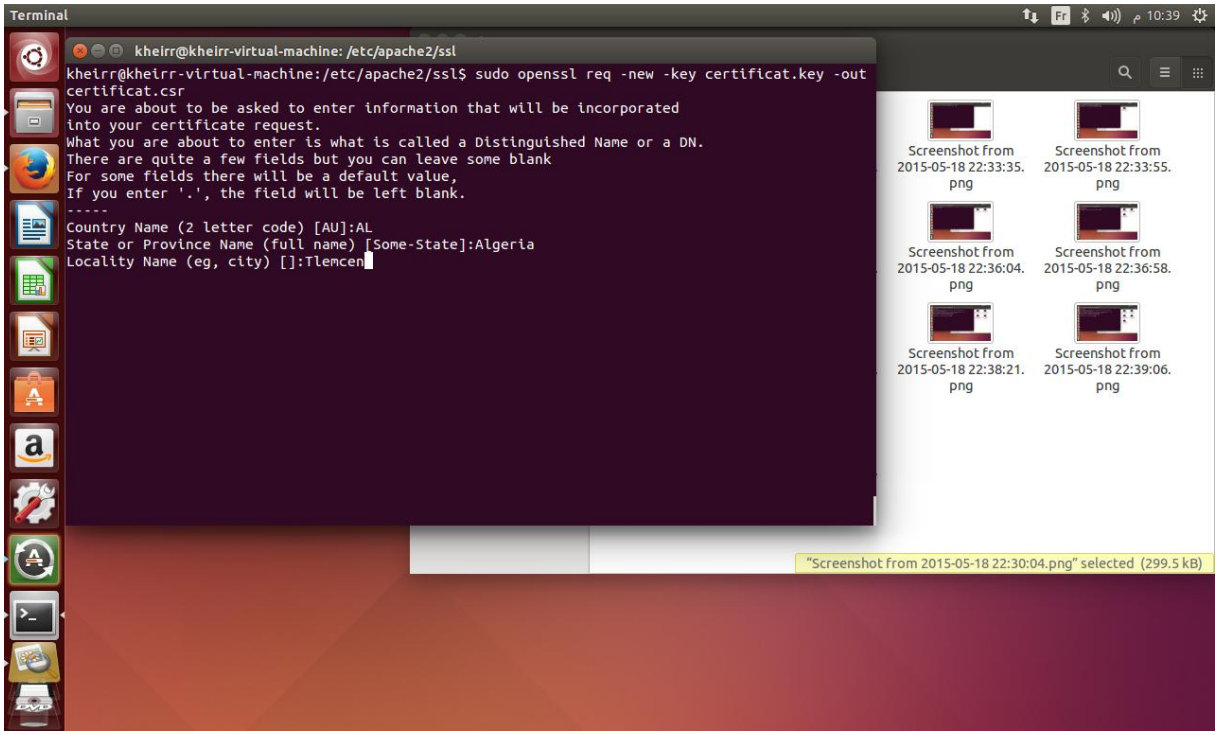
- Maintenant nous allons remplir nos informations sur le certificat
  Pour le premier champ, il est dédie pour les deux lettre du nom du pays par exemple **AL** pour **Algérie**



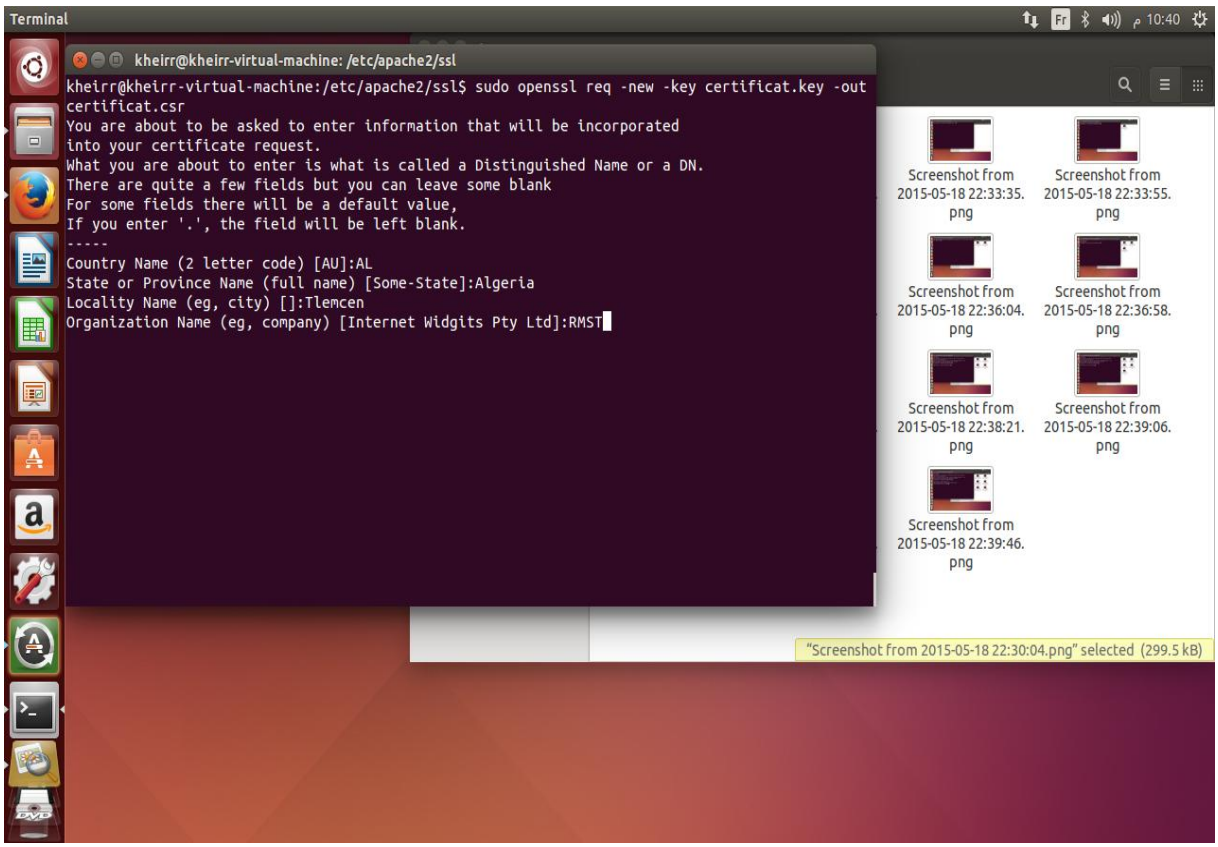
- Le deuxième champ et pour le nom de pays complet : **Algeria**



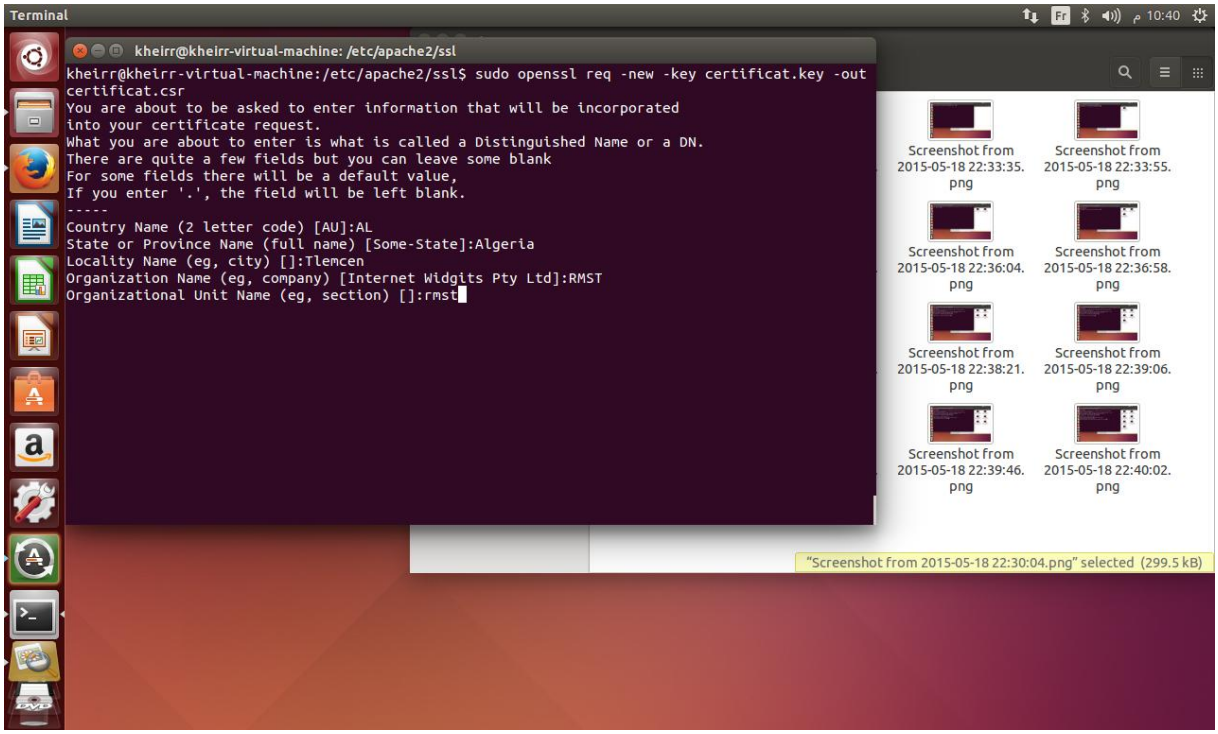
Le troisième champ : nous avous mis Tlemcen comme cité (city)



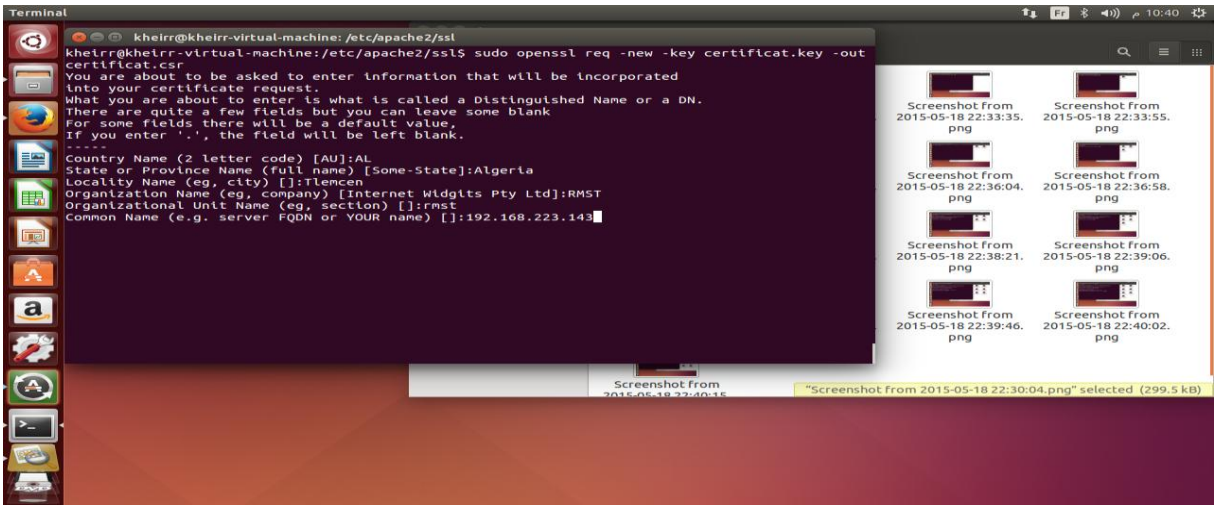
- Quatrième champ pour l'organisation ou compagnie donc nous avons choisi **RMST**



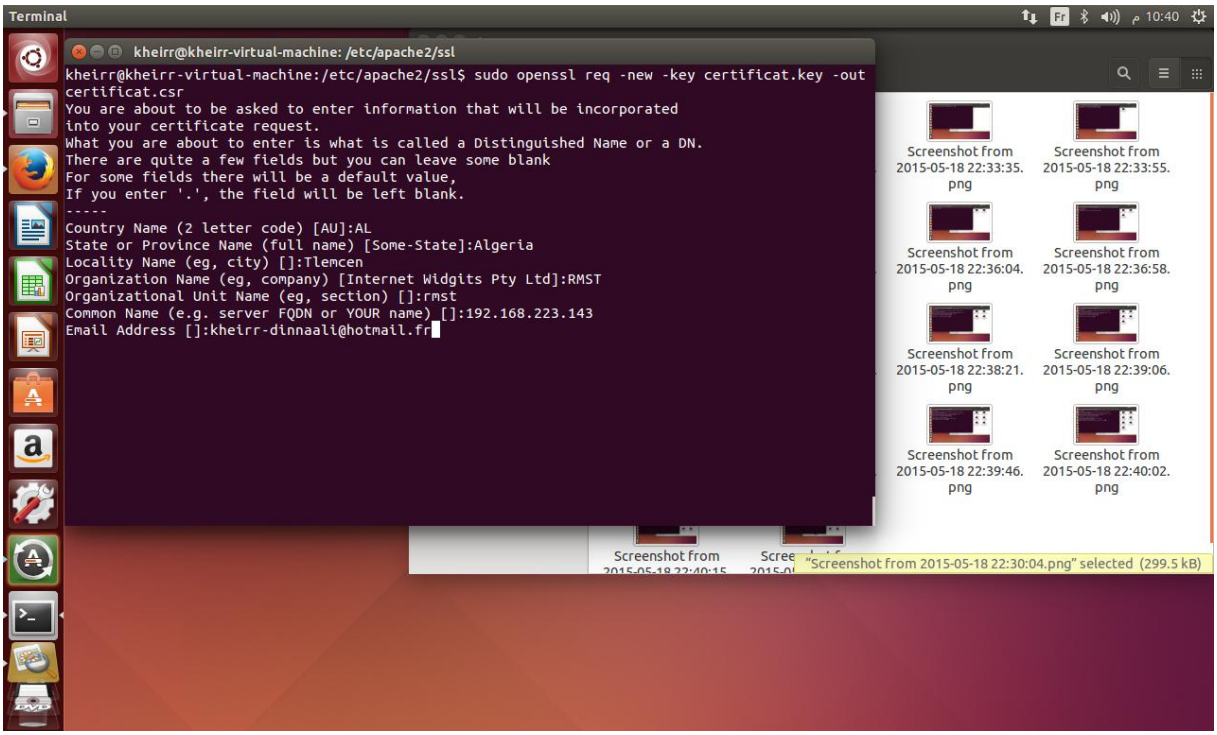
- Cinquième champ nous demande de remplir l'unité d'organisation donc nous avons choisi rmst



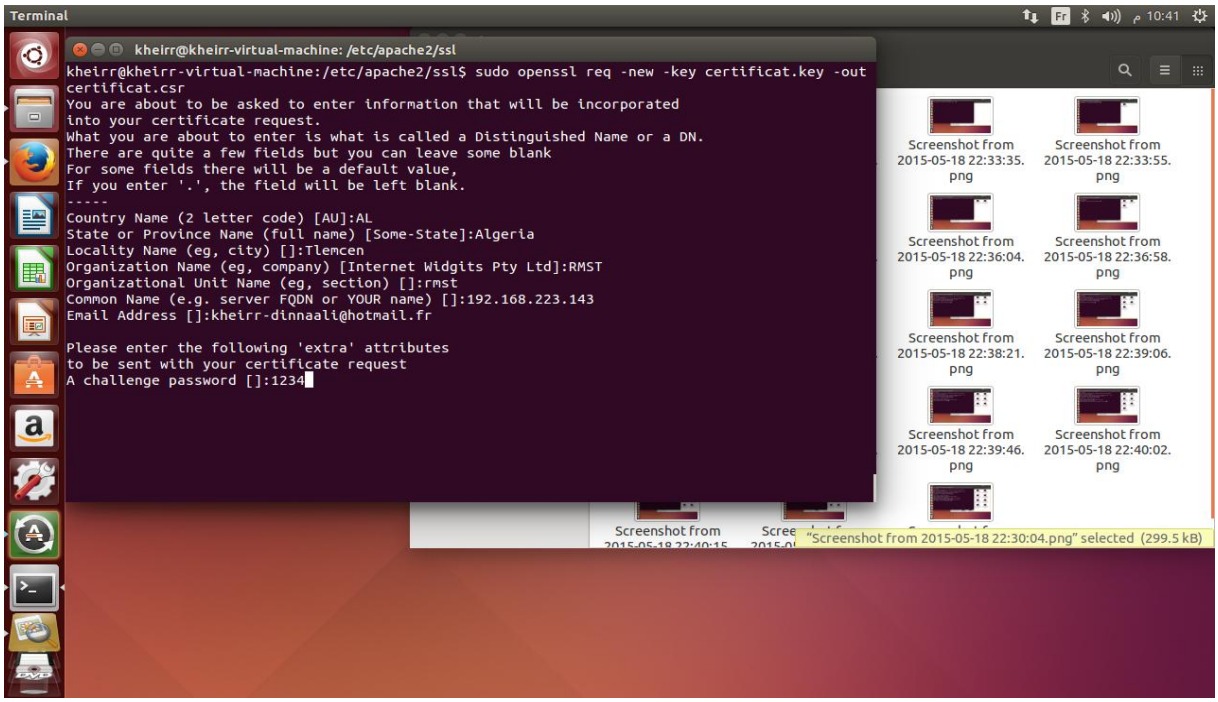
- Sixième champ est très important. dans ce champ, il faut bien vérifier le nom domaine que nous allons renseigner, il doit être identique avec notre site web ou URL. Nous pouvons même utiliser notre adresse IP statique, comme notre cas nous avons pris une adresse IP du server web où nous avons installé Apache2 et comme nous travaillons dans un réseau local, on a choisi l'adresse 192.168.223..



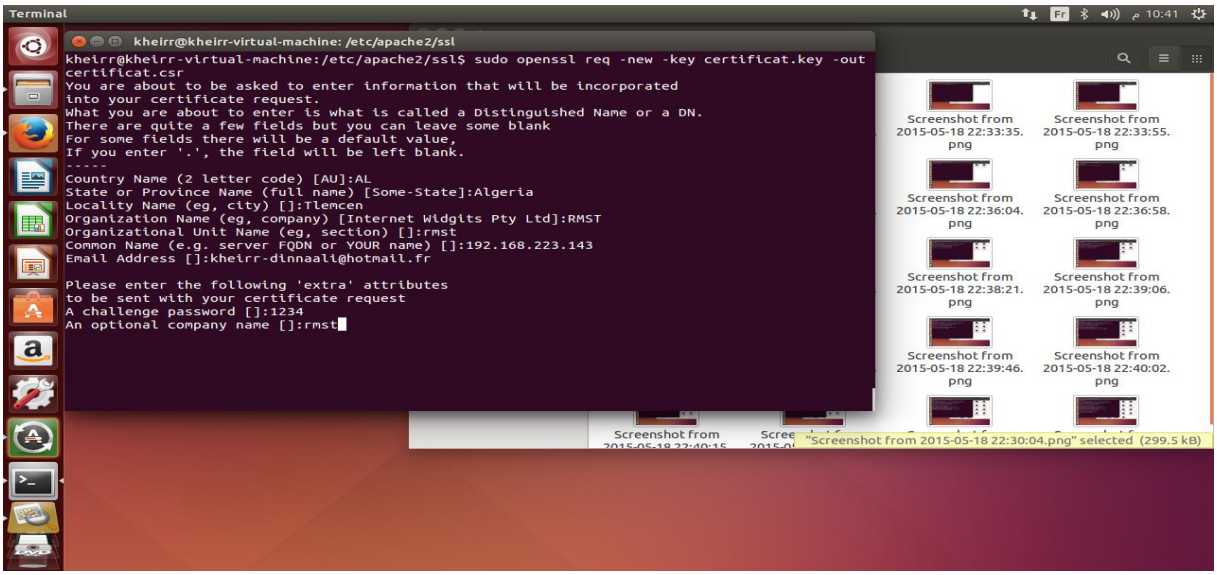
- Septième champ c'est pour l'e-mail



- Huitième champ est aussi très important pour un mot de passe pour la demande de certificat



- A la fin de cette étape il nous demande encore une fois le nom de compagnie et ce juste une option



Pour les clés et la demande de certificat nous les trouvons dans le dossier que nous avant créé /etc/apache2/ssl

Figure IV.7: l'emplacement de certificat et de la clé privée et publique

IV.3.7. Le résultat obtenu par les étapes précédant

- ✓ la clé privée (Private Key)

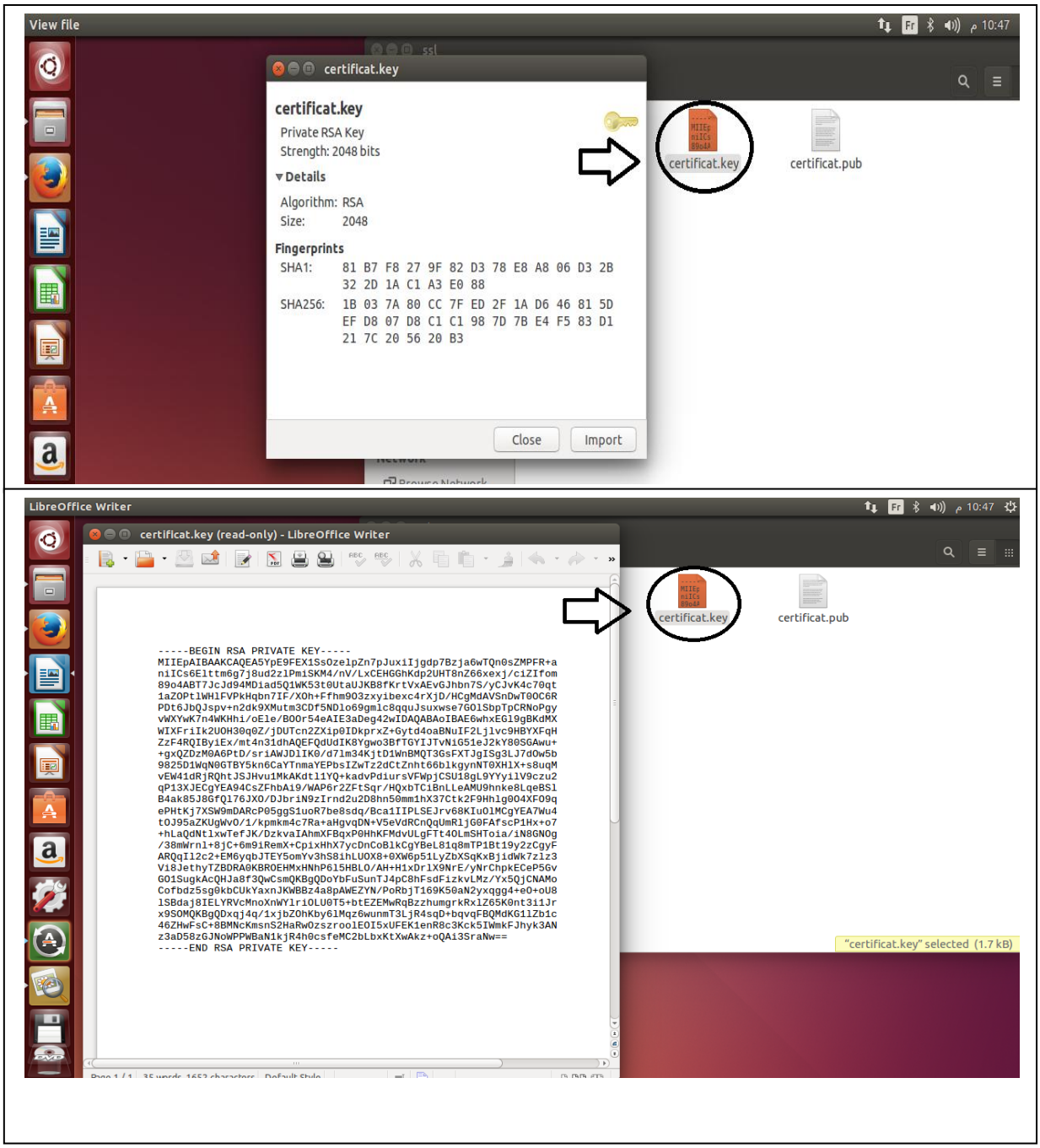


Figure IV.8: clé privé

✓ la clé publique (Public Key)

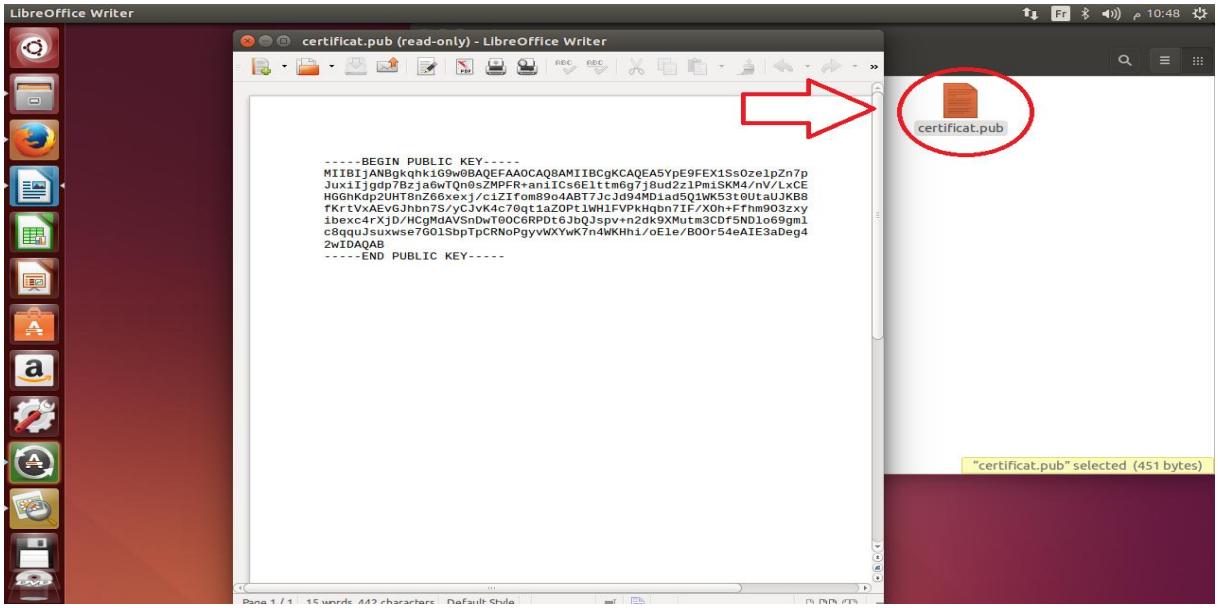


Figure IV.9 : Clé publique pour la demande de certificat

✓ la demande de certificat (certificat request)

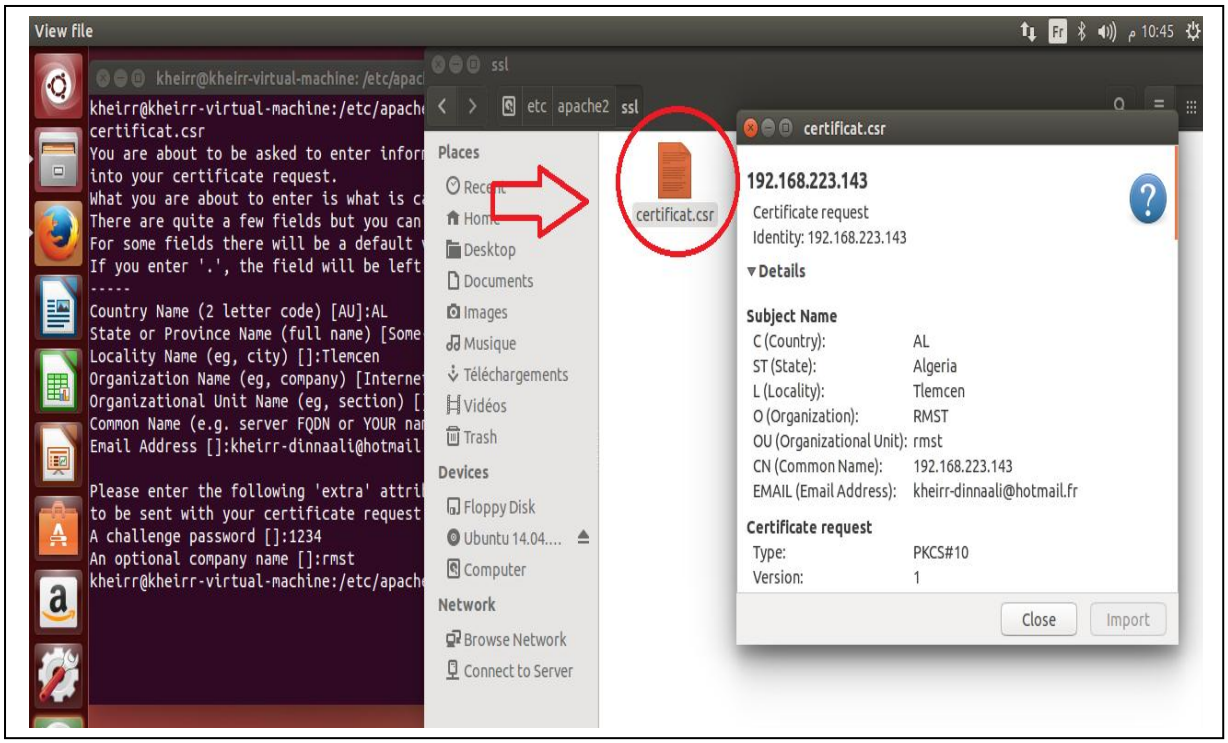


Figure IV.10 : la demande de signature de certificat

Après la génération de la demande de certificat deux choix s'offrent à nous :

- Envoyer le ficher certificat.csr à un organisme (la tierce confiance ou l'autorité de certification CA) et ainsi obtenir le certificat dument signé par la clé privé de l'organisme (après avoir payé).
- Ou bien signer nous-mêmes le certificat, avec un certificat auto signé, et pour signer un certificat nous devons crée notre propre autorité de certification cela implique donc de créer une clé privé et un certificat auto-signé

Nous avons opté pour le dernier choix.

3.8. Création de l'autorité du certificat (certificat auto signé)

C'est le même travail que précédemment, on a besoin d'une clé privé pour le certificat auto signé et après on va créer le certificat auto signé par la demande de certificat 'certificat.csr'.

3.8.1. Création de la clé privée

Avec la même commande nous allons créer la clé privé juste nous allons changer le nom de la clé privé par ca.key.

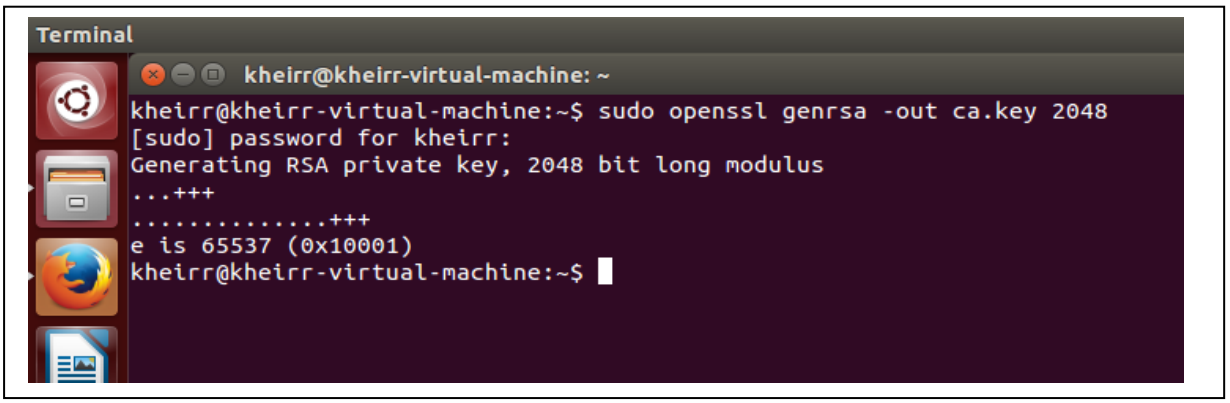


Figure IV.11 : Clé privé pour le certificat autorité

IV.3.8.2. Création du certificat auto signé

Toujours on va utiliser la clé privé précédente ca.key pour crée le certificat auto signé avec la commande « sudo openssl req –new –x509 –days 365 –key ca.key –out ca.crt »

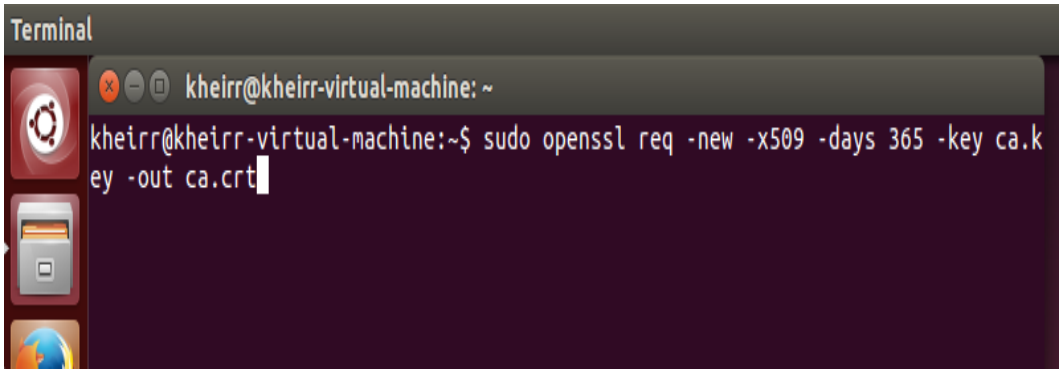


Figure IV.12 : Certificat auto –signé (certificat d'autorité)

Ensuite on doit remplir les sept champs avec nos information comme nous l'avons vu précédemment.

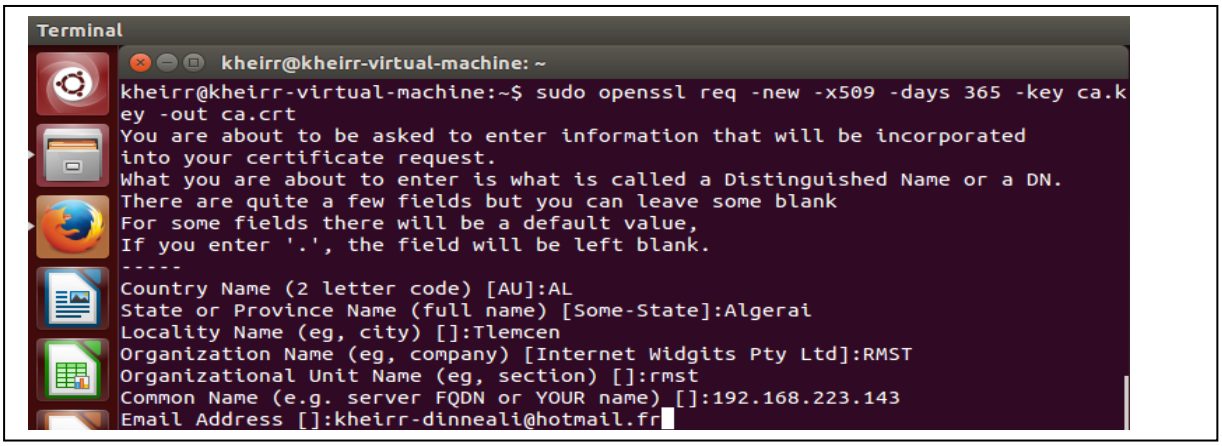


Figure IV.13 : les informations de certificat auto-signé

IV.3.8.3. Le résultat obtenu

- ✓ La clé privée du certificat auto signé

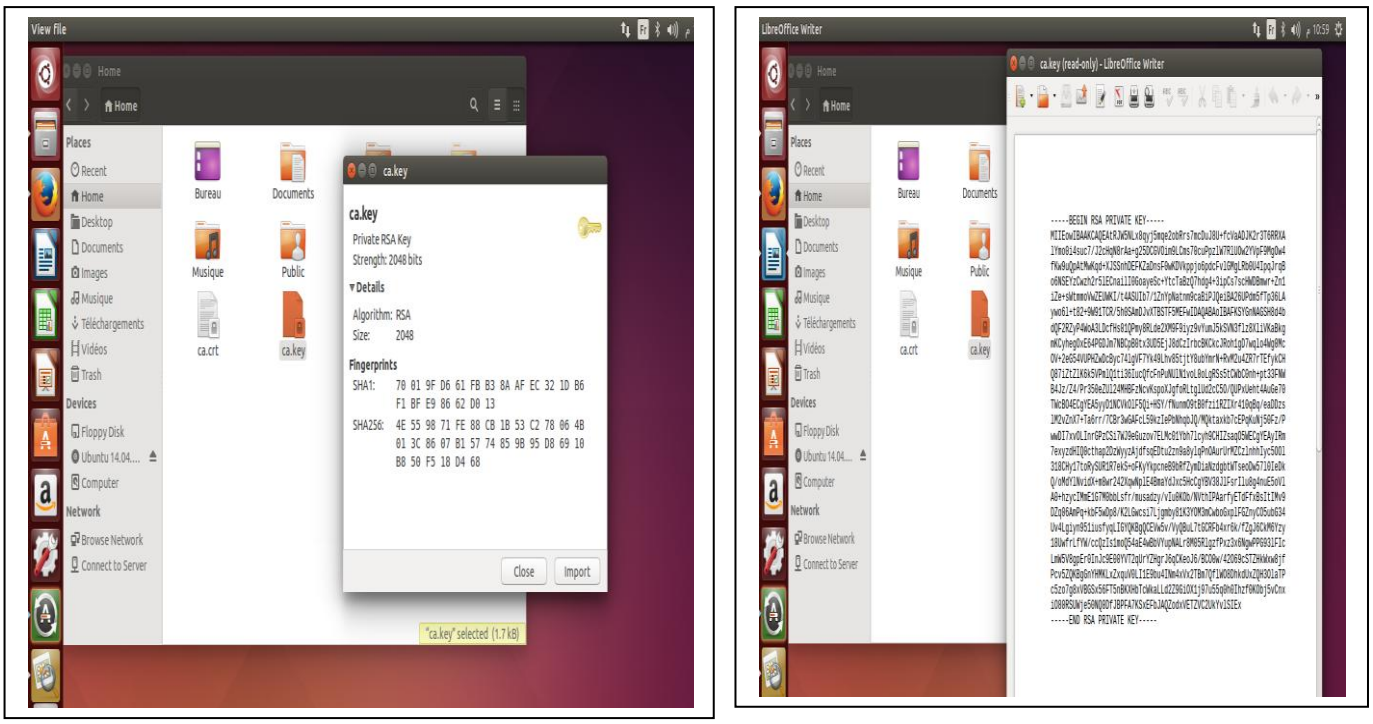


Figure IV.14 : Clé privé de certificat auto-signé

- ✓ La clé publique du certificat auto signé

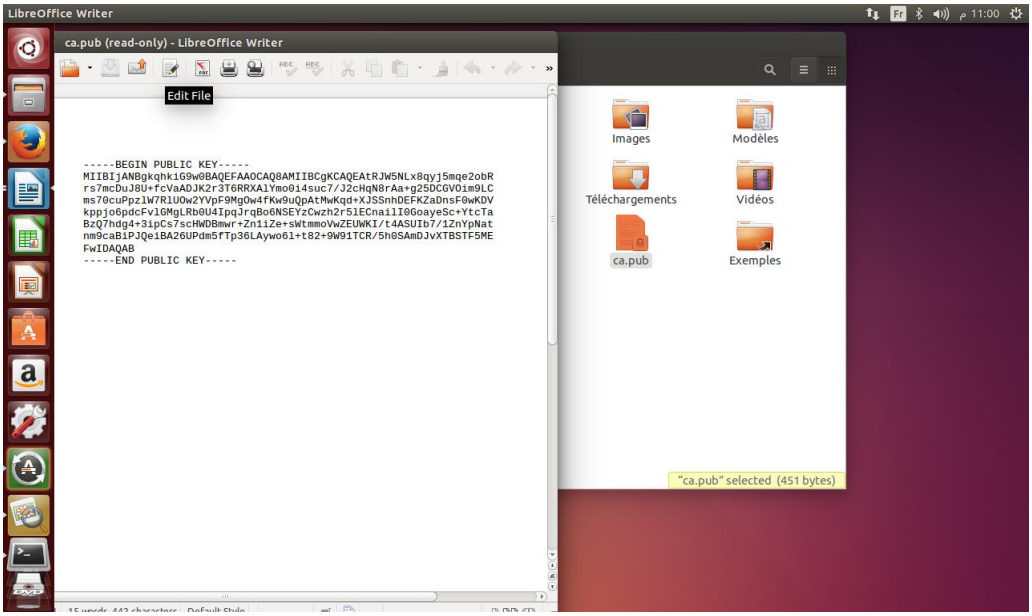


Figure IV.15 : Clé publique de certificat auto signé

✓ Certificat auto signé (notre propre CA)

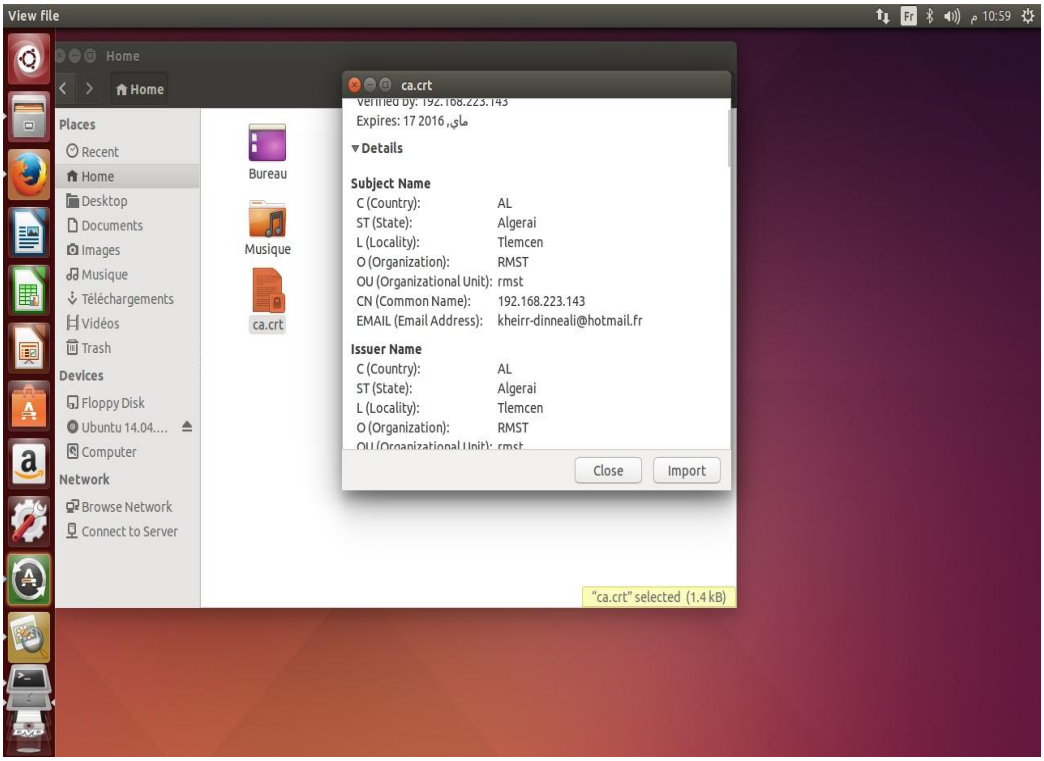


Figure IV.16 : Certificat auto-signé

IV.3.9. La signature du certificat serveur par notre propre CA (Certificat Autority)

C'est notre certificat d'autorité de certification qui ca permettre de signer les certificats créés 'certificat.csr'.

Par la commande « sudo openssl x509 –CA ca.crt –CAKey ca.key –in certificat.csr –req –set_serial 1 –out certificat.crt –days 365»

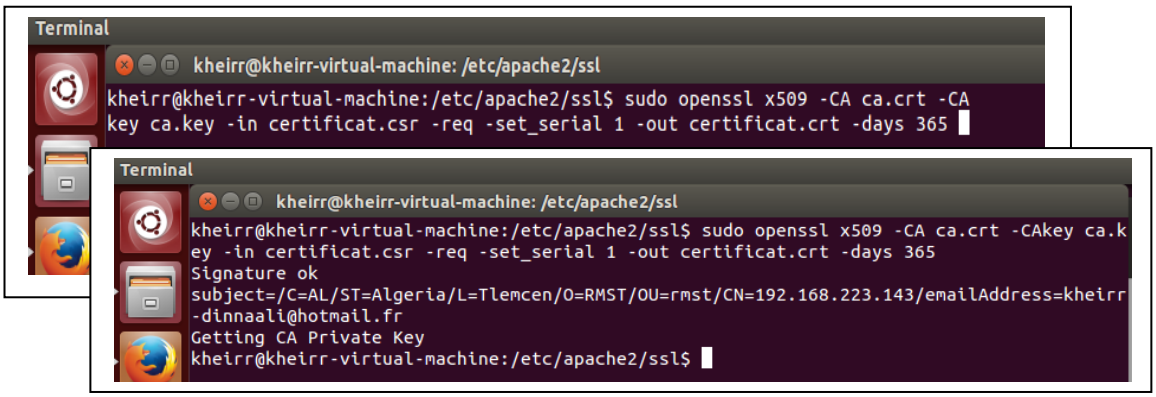


Figure IV.17 : Signature de la demande de certificat par certificat auto-signé

Résultat à la sortie « signature OK »

A la fin nous avons réussi à signer notre certificat 'certificat.csr' du serveur avec notre propre CA (Certificat Autority) à la sortie on obtient 'certifcat.crt' ça veut dire que le certificat est signé

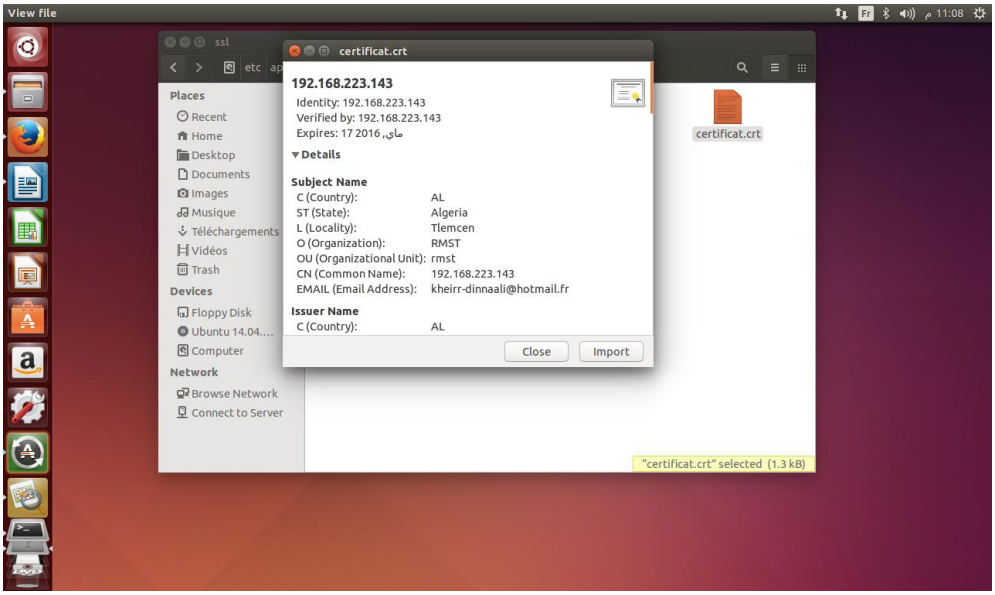


Figure IV.18 : demande de certificat signé par certificat auto-signé

IV.3.10. Configuration du SSL de l'Apache2

Tout d'abord il faut activer ce service ou vérifier qu'il est actif sous le terminal avec la commande « sudo a2nmod ssl »

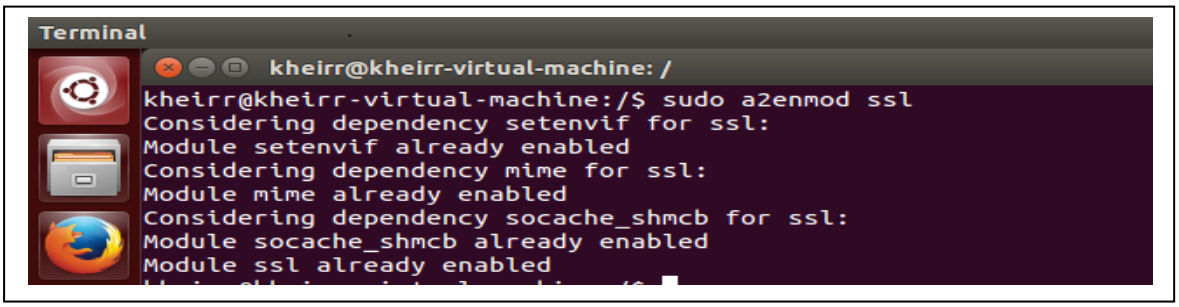


Figure IV.19 : Activation de SSL sous Apache2

Après au redémarre Apache2

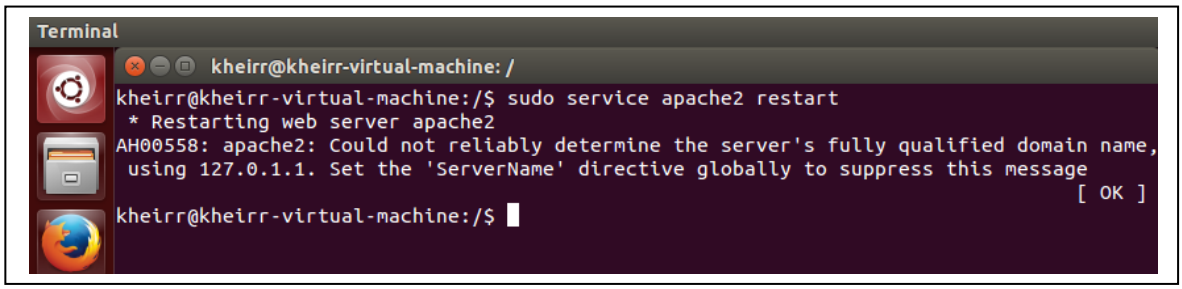


Figure IV.20 : Redémarre Apache2

Ici on commence à configure un fichier qui se trouvé sur Apache2 pour change le port 80 au 443 du protocole ssl pour activer le https

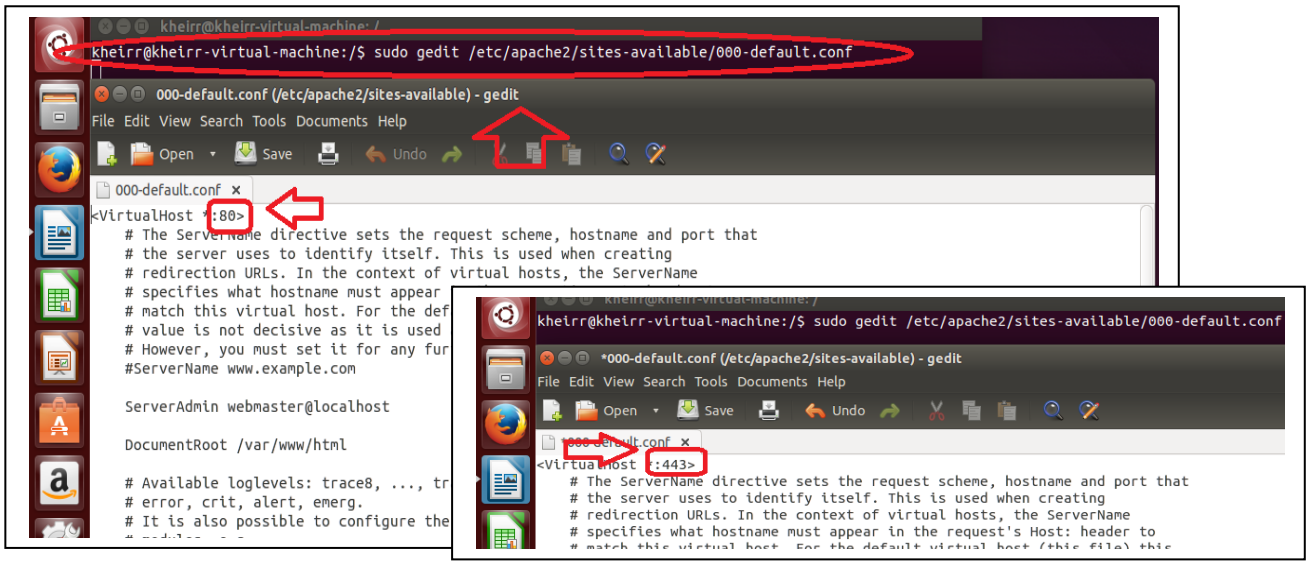


Figure IV.21 : Activation de port 443

Après on montre le chemin où se trouve notre certificat d'autorité avec ces lignes de texte.

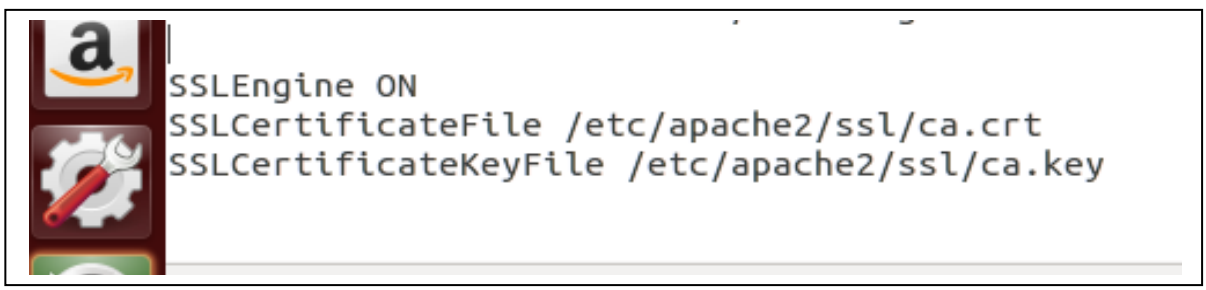


Figure IV.22 : Indication de chemin de certificat et la clé privée

A la fin de cette configuration on va tester avec le navigateur web par le port 443 sous URL https://192.168.223.143

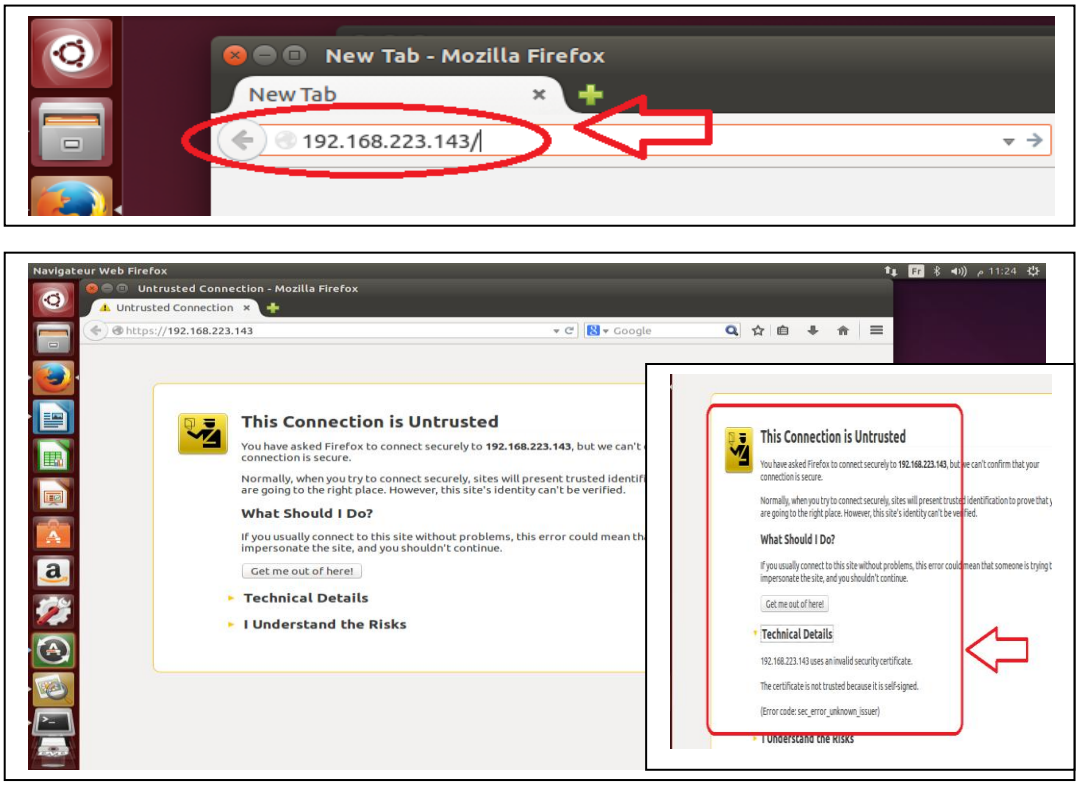


Figure IV.23 : tester le port 443

Note : comme notre certificat est un certificat auto-signé et elle n'est pas signé par les 33 certificats d'autorités supportés alors le navigateur ne fait pas confiance à notre certificat donc il prévient l'utilisateur pour les risques.

4. Tester le protocole SSL avec client Windows 7

Dans cette partie nous allons tester notre travail avec un notre système d'exploitation dans un réseau local avec la machine virtuelle, nous allons choisir Windows 7 comme client, tout d'abord il faut tester la connectivité entre le client et le serveur

4.1. Tester la connectivité de serveur

Le serveur et identifié sur le réseau par l'adresse IP 192.168.223.143 et le client et identifié par l'adresse 192.168.223.139, donc nous allons faire un Ping au serveur 192.168.223.143

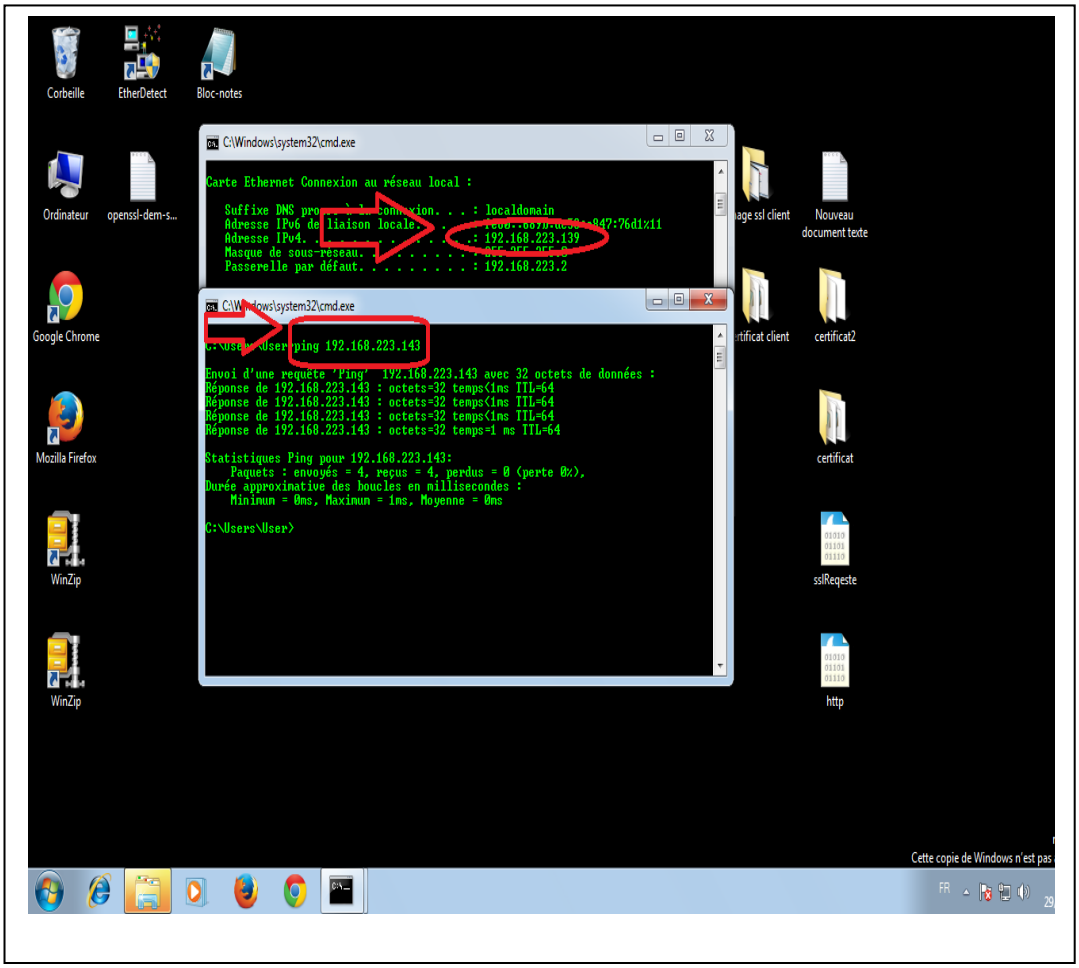


Figure IV.24: Tester la connexion entre un client et le serveur web

4.2. Tester sur le port 443 par Windows 7

Après le test de la connexion nous allons voir le fonctionnement du protocole SSL (port 443) sur le système Windows 7 avec https sur Google chrome et Mozila

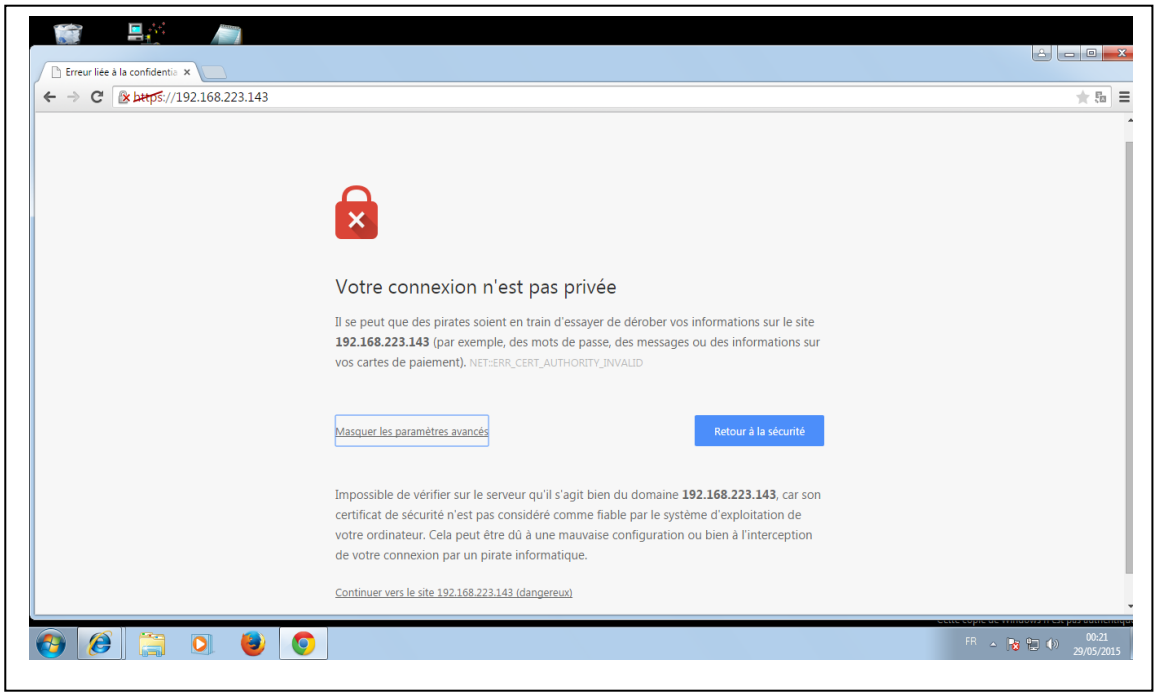


Figure IV.25: Tester le port 443 avec Google Chrome

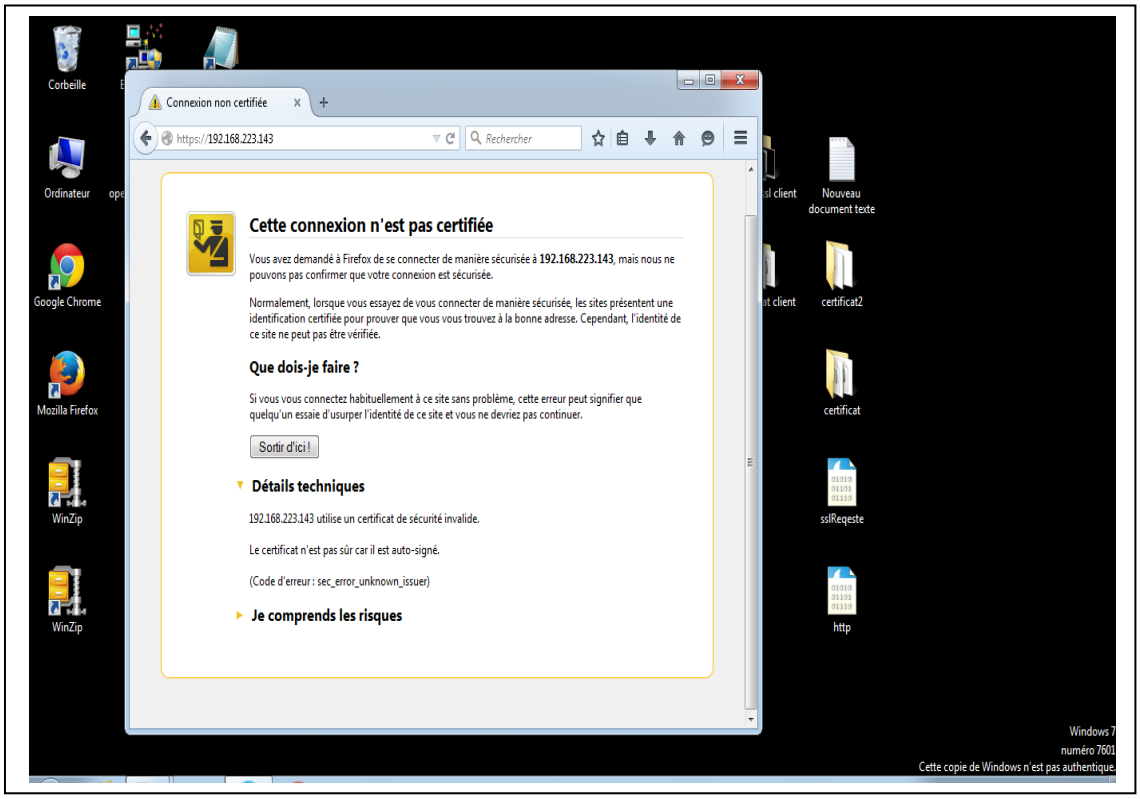


Figure IV.26 : Tester le port 443 avec Mozila

IV.5. Conclusion :

Dans ce chapitre, nous avons présenté les différentes étapes d'implémentation d'une solution de sécurisation SSL/TLS des communications entre clients et serveurs web. Nous avons identifié quelques difficultés sur le plan équipements pour conduire nos tests. C'est la raison pour laquelle nous nous sommes dirigés vers une implémentation sur machines virtuelles. Nous avons utilisé pour cela VMware. Notre objectif au départ, était d'implémenter une solution SSL reposant sur une authentification mutuelle (bidirectionnelle). Nous sommes parvenus à une implémentation dans un seul sens (authentification du serveur uniquement). Nous avons testé vérifié avec succès toutes les fonctionnalités de sécurité SSL/TLS (authentification du serveur par certificat, confidentialité et intégrité) et ce grâce à l'analyseur réseau Wireshark.

Table des matières

Conclusion générale

Nous arrivons au terme de notre travail de projet de fin d'étude portant sur l'étude et l'implémentation d'une solution de sécurisation des communications par SSL/TLS.

Grâce à sa simplicité de déploiement, SSL/TLS est actuellement le protocole d'authentification et sécurisation des échanges le plus déployé. Nous avons commencé par bien comprendre son fonctionnement et analysé les échanges de sécurité impliqués entre un client et un serveur web. Ensuite nous nous passé à l'implémentation avec la boîte à outils cryptographiques Openssl qui permet entre-autres la génération de clé privé et publique et la création des certificats numériques, indispensables dans un échange SSL/TLS.

A la fin de ce travail, il est tout à fait clair que l'objectif fixé a été atteint, mais néanmoins nous voulions également réaliser pour les besoins de certaines applications web, l'authentification mutuelle, nous ne sommes malheureusement pas parvenu à le faire dans le délai qui nous est imparti. C'est donc l'une des perspectives.

Il est inutile de cacher qu'il reste encore beaucoup d'autres travail à faire, tant sur le plan intégration de SSL/TLS dans tous les services réseaux qui ne cessent de se multiplier aujourd'hui et qui sont vulnérables vis-à-vis des attaques aussi actives que passives, tant sur le plan développement d'autres protocoles de sécurité adaptés aux environnements mobiles.

Bibliographie / webographie

[1] : https://www.securiteinfo.com/conseils/introsecu.shtml.

[2] : http://www.picsi.org/parcours_68_312.html (**Thème** : Introduction à la problématique de la sécurité, **Auteur(s) :** Marc Rybowicz.

[3] : https://schaellerclement.wordpress.com/veille-technologique.

[4] : Bouchikhi Mouhamed El Amin & Derfouf Hicham« L'utilisation de RADUIS sous Windows Server 2008 pour sécuriser un réseau sans fil 802.11 » Master universite aboubker belkaid 24 octobre 2011.

[5] : « http://www.nolot.eu/Download/Cours/reseaux/m2pro/ARS0809/ArchiResoSecu-Cours1.pdf » (coure Master 2 Professionnel Informatique de l'université de REIMS Champagne-Ardenne).

[6] : Livre Blanc Network Access Control (Contrôle d'accès au réseau) enterasy 'Secure Networks'.

[7] : Mahammedi Nadjiba & Mahdadi Houda « Implémentation de bechemark d'opération crypto basées ECC pour l'étude et comparaison de courbes elliptiquesF_P et F_{2^n} » Master Académique juin 2013.

[8] : https://www.securiteinfo.com/cryptographie/pki.shtml

[9] : http://www.commentcamarche.net/contents/992-firewall-pare-feu

[10] : http://lacl.univ-paris12.fr/cegielski/sec/ch4.pdf

[11] : Christophe Yayoun & Fabrice Romeland « Le protocole SSL/TLS » Mini-Mémoire, Ingénierie Informatique et Reseaux Janvier 2002