2011-05

Sécurisation des systèmes de messagerie électronique, avec le protocole SSL (Secure Sockets Layer)

Gatoto, Placide

UB, FS

UNIVERSITE DU BURUNDI

FACULTE DES SCIENCES

DEPARTEMENT DE PHYSIQUE

SECURISATION DES SYSTEMES DE MESSAGERIE ELECTRONIQUE AVEC LE PROTOCOLE SSL (Secure Sockets Layer)

Par :

GATOTO Placide

Sous la direction de :

Docteur AKIMANA Rachel

Mémoire présenté et défendu publiquement en vue de l'obtention du grade de Licencié en Sciences Physiques

Bujumbura, mai 2011

DEDICACE

A tous ceux qui m'est chers, je dédie ce mémoire.

REMERCIEMENTS

Au terme de ce travail, il m'est agréable de devoir adresser mes vifs remerciements à Madame Dr. AKIMANA Rachel, Chef du département de Physique et Professeur à l'université du Burundi, pour m'avoir proposé le sujet de ce mémoire et en avoir assuré la direction. Sa compétence, sa patience et surtout sa disponibilité malgré ses nombreuses responsabilités ont bien guidé ma marche qui n'était pas toujours sur une bonne voie. Je réitère ma profonde reconnaissance.

Que tous les professeurs, enseignants et autres qui ont participé à ma formation y trouvent aussi l'expression de ma profonde gratitude.

Je ne peux pas terminer sans adresser mes sincères remerciements aux nombreuses et généreuses personnes pour le soutien moral et matériel qu'elles m'ont apporté.

Merci à toutes les personnes qui ne cessent de penser à moi !

TABLE DES MATIERES

LISTE DES TABLEAUX

LISTE DES FIGURES

SIGLES ET ABREVIATIONS

1. 3 DES : Triple Data Encryption Standard.
2. AES : Advanced Encryption Standard.
3. API : Application Program Interface.
4. ARP : Address Resolution Protocol
5. CRL : Certificate Revocation List.
6. CSR : Certificate Signing Request.
7. DES : Data Encryption Standard.
8. DN : Distinguish Name.
9. DNS : Domain Name System.
10. DSA : Digital Signature Algorithm.
11. ESMTP : Extended SMTP.
12. FTP : File Transport Protocol.
13. Go : Giga octet.
14. HTTP : Hyper Text Transfer Protocol.
15. IDEA : International Data Encryption Algorithm.
16. IETF : Internet Engineering Task Force.
17. IMAP : Internet Message Access Protocol.
18. IP : Internet Protocol.
19. ipad : inner padding.
20. IRC : Internet Relay Chat.
21. ISO : International Organization for Standardization.
22. IV : Initial Values.
23. MAC : Message Authentication Code.
24. MD : Message Digest.
25. MDA : Mail Delivery Agent.
26. Mo : Méga octet.
27. MTA : Mail Transfer Agent.
28. MUA : Mail User Agent.
29. opad : out padding.
30. OSI : Open Systems Interconnections.
31. PGP : Pretty Good Privacy.
32. PKCS : Public Key Cryptographic Standards.
33. PKI : Public Key Infrastructures.
34. POP : Post Office Protocol.
35. RC : Rivest's Cod.
36. RFC : Request For Comments.
37. RSA : Rivest Shamir Adleman.
38. S/MIME : Secure/Multipurpose Internet Mail Extensions.
39. Sec : Seconde.
40. SI : Sécurité Informatique.
41. SHA : Secure Hash Algorithm.
42. SMTP : Simple Mail Transport Protocol.
43. SSH : Secure Shell.
44. SSL : Secure Socket Layer.
45. TCP : Transport Control Protocol.
46. TLS : Transport Layer Security.
47. UB : Université du Burundi.
48. UIT : Union Internationale de télécommunication.
49. URL : Uniform Resource Location.
50. VPN : Virtual Private Network.
51. WWW : World Wide Web.

LISTE DES NOTATIONS

1. bit : nombre binaire.
2. D : bloc droit d'un message.
3. exp : exponentiel.
4. f : fonction de confusion.
5. G : bloc gauche d'un message.
6. h : une fonction de hachage.
7. k : clé privée.
8. k : clé publique.
9. K : la clé de chiffrement.
10. K_i : sous-clé de chiffrement.
11. m/M : un message.
12. mod n : désigne le modulo de n.
13. M_S : message pouvant être signé.
14. p_n : la nième probabilité, n étant un nombre naturel.
15. Sig : signature.
16. Xor : désigne un ou exclusif.
17. $\oplus$: désigne aussi un ou exclusif.
18. || : désigne une concaténation.
19. @ : l'arobase.

CHAPITRE 0. INTRODUCTION GENERALE

L'informatique et surtout l'Internet depuis sa création (vers 1950) et suite à son développement spectaculaire voit ses utilisateurs augmenter du jour le jour. Aujourd'hui, la plupart des transactions s'effectuent via Internet. On arrive à un stade où on fait le commerce via Internet (Commerce électronique), où les transactions bancaires se font sur Internet, à ces deux applications s'ajoutent plusieurs autres variétés de communications utilisant le réseau Internet comme moyen de transport.
A voir l'importance des transactions effectuées via Internet, il convient de sécuriser les informations qui circulent à tout moment grâce à ce réseau qu'est Internet.

Un des problèmes les plus importants en matière de sécurisation de ces transactions est le fait que les utilisateurs ont la difficulté à croire quelque chose de mal peut leur arriver jusqu'au jour où cela arrive. La vérité c'est que les mauvaises choses arrivent et se produisent plus souvent qu'on ne le croyait. Peu importe pourquoi ou comment votre entreprise est une cible potentielle d'attaque, le retour à la normale demande généralement beaucoup de temps et d'effort. Supposons qu'un système informatique ne soit pas disponible pendant plusieurs jours ou qu'il y ait perte de toutes les données stockées dans les ordinateurs d'une quelconque entreprise ou encore que le compétiteur le plus féroce puisse obtenir la liste de tous ses clients ainsi que leurs chiffres de vente et d'autres données essentielles. Il se passera combien de temps avant que cette entreprise puisse s'en rendre compte ? Quels seraient les coûts pour l'entreprise ? A-t-elle les moyens de supporter de telles pertes ?

« A voir les dommages causés par une information volée, on arrive à dire qu'une information perdue génère une perte d'activité » [1]. Et cela montre l'importance qu'il faut réserver à la sécurité informatique. Il est rare, voire même imprudent qu'une personne quitte ses locaux sans les fermer à clé. C'est également vrai pour ce qui est de la sécurité de ses informations. Bien sûr, il n'y a aucun moyen d'obtenir un niveau de sécurité absolue. Mais à quoi bon barricader les portes, si les voleurs peuvent entrer par les fenêtres ? Toutefois, il y a moyen d'obtenir un niveau de sécurité acceptable et se préparer en cas d'infraction.

Dans notre travail, nous allons étudier le protocole SSL (Secure Socket Layer) utilisé pour sécuriser la plupart des transactions effectuées sur l'Internet. Nous allons focaliser notre attention sur la sécurisation de la messagerie électronique avec SSL. En effet, la messagerie électronique est l'une des applications utilisant intensément le réseau Internet et qui est aisément adoptable par même les non informaticiens. Ceci pour en arriver au fait que la messagerie électronique est utilisée par beaucoup de navigateurs, qui des fois ignorent même les risques potentiels qu'encourent leurs informations échangées sur Internet. Ce travail a pour objectif d'analyser les faiblesses en matière de sécurité des systèmes de messagerie.

Dans le premier chapitre nous définissons les concepts généraux de la sécurité, ses objectifs et les moyens utilisés pour atteindre ces objectifs. Entre autres moyens utilisés il y a le protocole SSL. C'est ainsi que le deuxième chapitre est consacré au protocole SSL. La raison ayant guidé le choix de SSL est simplement le fait qu'il est beaucoup intégré dans beaucoup d'applications Internet.

Dans le troisième chapitre, nous faisons une étude de cas sur l'utilisation de SSL dans la sécurisation des systèmes de messageries électroniques. Dans ce chapitre, en plus des systèmes de messagerie ouverts (yahoo, gmail, etc) nous allons étudier le système de messagerie électronique institutionnel et plus particulièrement celui de l'université du Burundi. ub .edu .bi. Nous allons terminer notre travail par une brève conclusion.

CHAPITRE.I : LA SECURITE INFORMATIQUE

I.0. Introduction

La sécurité informatique est l'ensemble des moyens techniques, organisationnels, juridiques et humains nécessaires et mis en place pour conserver, rétablir, et garantir la sécurité d'un système d'information. La sécurisation des informations est une activité du management du système d'information.
Le terme « système d'information » désigne ici tout système dont le fonctionnement fait appel, d'une manière ou d'une autre, à l'électricité et destiné à élaborer, traiter, stocker ou présenter de l'information. Les systèmes d'information s'appuient en générale sur des systèmes informatiques pour leur mise en œuvre. Ils comprennent les données numériques et dans certains cas, les données sur papier. De telles données sont des cibles potentielles des attaques de types divers, susceptibles de les altérer ou de les détruire ou encore de les révéler à des tiers qui ne devraient pas normalement en prendre connaissance.[2]
Depuis 1970, l'accès rapide aux informations, la rapidité et l'efficacité des traitements, le partage des données et l'interactivité ont augmenté de façon considérable. Malheureusement c'est également le cas des pannes, des indisponibilités, erreurs, négligences et malveillance surtout avec l'ouverture de nos systèmes aux grands réseaux tels que l'Internet.

Un système d'information pour une entreprise, est généralement défini par un ensemble des données et des ressources matérielles et logicielles de l'entreprise permettant de stocker ou de faire circuler des données. Le système d'information représente un patrimoine essentiel de l'entreprise qu'il convient de protéger.
D'une manière générale, la sécurité informatique consiste à assurer que les ressources matérielles ou logicielles et les données d'une organisation sont uniquement utilisées dans le cadre prévu.
Une fois les risques énoncés, il est souhaitable de déterminer des objectifs de sécurité auxquels la Sécurité Informatique doit se cantonner afin de garantir les droits d'accès aux données et ressources d'un système informatique.

I.1. Les objectifs de la sécurité informatique

Dans beaucoup d'applications, la sécurité informatique se réduit à des procédures offrant des mécanismes d'identification et de contrôle d'accès. Mais en plus de ces mécanismes d'identification et suivant les cas, devraient s'ajouter d'autres fonctions de sécurité comme par exemple le fait d'assurer que les données sensibles ne soient révélées aux utilisateurs non autorisées même si elles transitent sur un réseau non sécurisé, le fait qu'elles ne soient pas altérées ou modifiées ou encore détournées. Avant d'aller plus loin dans ce chapitre, nous aimerions mentionner le fait que les solutions de sécurité dans une application apportent une complexité supplémentaire pouvant affecter les performances de l'application. Ceci est une des raisons qui font que les développeurs d'applications réduisent au minimum (identification) les solutions sécuritaires dans leurs produits.

I.1.1. La confidentialité

La confidentialité des données est assurée lorsque les données ne sont compréhensibles que par les personnes dont l'accès leur est autorisé. Dans le contexte de la communication entre deux parties, la donnée en question peut être une information envoyée d'une partie à l'autre, et la personne ayant le droit d'accéder aux données est le destinataire du message envoyé. La confidentialité peut être compromise par un pirate tentant d'intercepter le message en transit sur le réseau et en extraire des informations utiles.

I.1.2. L'intégrité

L'intégrité d'une donnée est garantie quand la donnée n'est pas altérée durant son transfert ou son stockage .Dans le contexte de la communication entre deux parties, cela signifie que le destinataire de la donnée doit pouvoir la visualiser comme l'expéditeur l'a créée. Il est à noter qu'un message pourrait être altéré durant son transfert soit par un problème lié au support, soit délibérément par une personne mal intentionné. La sécurité informatique étudie et propose des mécanismes permettant de détecter de telles violations de l'intégrité des données.

I.1.3. L'authentification et l'identification

a) L'authentification

L'authentification consiste à s'assurer qu'une personne dit la vérité sur un ou plusieurs de ses attributs. Dans le contexte de la communication entre deux parties, un attribut d'une partie pourrait être son nom, sa raison sociale, ou tout autre élément de ce qu'elle est, ce qu'elle sait, ou ce qu'elle possède, qui pourrait assurer l'autre partie de l'identité de son partenaire. Pour un système informatique, l'authentification est la procédure qui consiste à vérifier l'identité d'une entité (personne, ordinateur...), afin d'autoriser l'accès de cette entité à des ressources (systèmes, réseaux, applications...). L'authentification permet donc de valider l'authenticité de l'entité en question.[3]
Lors de l'accès à un site web de vente en ligne par exemple, le consommateur voudrait, au moment du paiement, être sûr qu'il donne son argent à la boutique en ligne dont il connaît la réputation, et non à un escroc qui encaisserait l'argent sans suite. C'est pour cela qu'il a besoin de l'authentifier pour être sûr que le site web est bien la fameuse boutique en question, d'après son nom.

b) L'identification

L'identification est la procédure par laquelle une entité présente ses attributs à un système informatique auquel il veut s'identifier. Ces attributs font l'objet par la suite d'authentification. En pratique, l'identification permet à une entité de se faire reconnaître du système par un élément dont on l'a doté. S'identifier c'est communiquer une identité préalablement enregistrée. Il existe plusieurs façons de s'identifier:

-identification faible : c'est une identification qui se fait par un mot de passe qui a été préalablement enregistré au système où on veut s'identifier.

-identification forte : dans cette catégorie on trouve les méthodes d'identifications dites challenge-response et les One Time password. Dans le premier cas, on a un prouveur (l'entité qui s'identifie) qui prouve sa connaissance d'un secret au vérificateur (le système informatique) sans pour autant le révéler. A chaque session d'identification, le vérificateur pose une question différente (un challenge) au prouveur, à laquelle ce dernier répond grâce à la connaissance de son secret (response). Dans le deuxième cas, on a un système de mots de passe mais qui change à chaque session d'identification.

I.1.4. La non répudiation

La non répudiation permet de s'assurer qu'un acte (un message, édition d'un document, etc.) n'est remis en cause par aucune des parties concernées. Concernant la sécurité numérique, la non répudiation permet de vérifier que l'expéditeur et le destinataire sont bien les parties qui disent avoir envoyé ou reçu le message. La non répudiation à l'origine prouve que les données ont été envoyées par l'expéditeur attendu, et la non répudiation à l'arrivée prouve qu'elles ont été reçues par le destinataire. Généralement, la non répudiation est atteinte en utilisant la technologie de signature digitale qui sera vue dans les sections suivantes (voir I.2.4). La cryptographie aussi dite science du secret est la science qui s'active à fournir des moyens logiques permettant d'atteindre ces objectifs que nous venons d'énoncer.

I.2. La cryptographie : fondement de la sécurité informatique

I.2.1. Introduction

Le mot cryptographie est utilisé pour désigner l'ensemble de techniques permettant de rendre des messages inintelligibles sans une action spécifique. « La cryptographie (cacher et écrire) est l'art et la science de transformer systématiquement un message écrit, dit clair en un autre message également écrit et illisible dit chiffré ou cryptogramme qui ne présente aucun sens pour toute autre personne que l'entité connaissant la convention employée » [19]. Définie de cette façon, il y a lieu de réduire les fonctions de la cryptographie au simple chiffrement. Néanmoins, la cryptographie englobe, en plus du chiffrement, d'autres fonctions de la sécurité informatique que nous allons décrire dans la section suivante.

I.2.2. Les fonctions de la cryptographie

Historiquement, la première utilisation des techniques cryptographiques était d'assurer la confidentialité de l'information sensible. Cependant, il faut noter que l'on utilise aujourd'hui des techniques de ce type pour assurer d'autres fonctions qui s'ajoutent à cette première à savoir l'authentification, l'intégrité et la non répudiation.

I.2.3. Le chiffrement

Le chiffrement est une méthode de sécurisation des données permettant d'assurer la confidentialité des données. Il consiste à chiffrer un message envoyé par un utilisateur. Cette méthode s'appelle communément le cryptage. Elle permet de protéger les données contre des mises à jour non désirées, des consultations abusives ou encore des suppressions non souhaitées. [4]

On distingue deux grandes familles d'algorithmes de chiffrement tels que le chiffrement symétrique où les clés de chiffrement et de déchiffrement sont les mêmes et le chiffrement asymétrique où on utilise des clés différentes pour le chiffrement et le déchiffrement.

I.2.3.1. Algorithmes de chiffrement symétrique (ou à clé secrète)

Les algorithmes à clé symétrique (ou secrète) sont basés sur l'utilisation d'une même clé pour chiffrer et déchiffrer le message. L'un des concepts fondamentaux de la cryptographie symétrique est la clé, qui est une information devant permettre de chiffrer et de déchiffrer un message et sur laquelle peut reposer toute la sécurité de la communication.

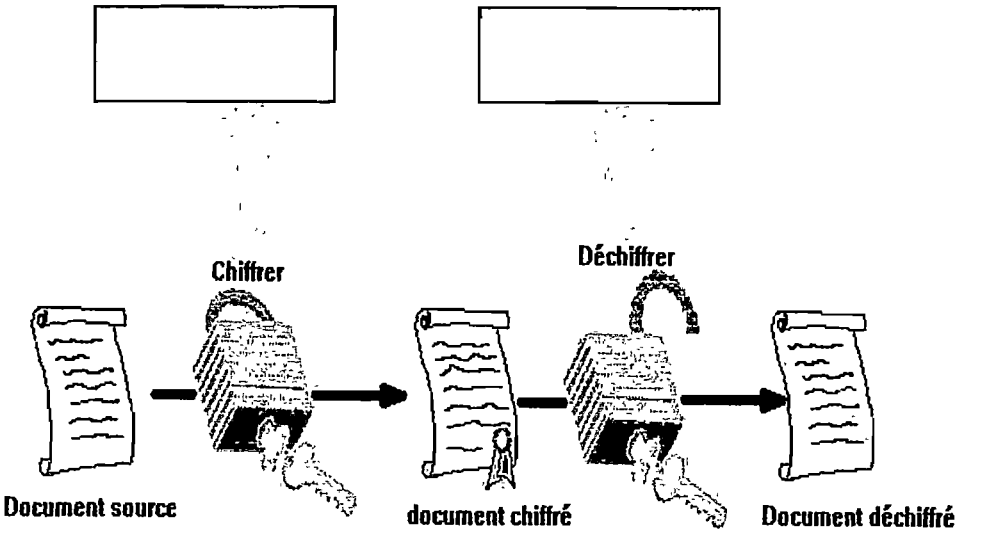
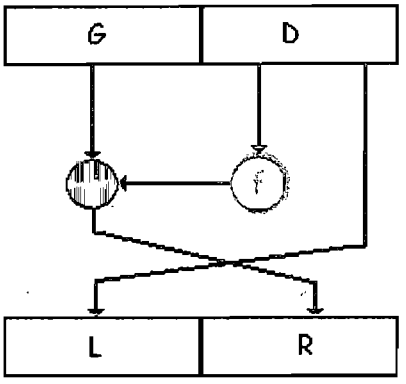


Figure I.1 : Représentation du fonctionnement des clés secrètes [5]

Le problème de cette technique est lié à la distribution de la clé. En effet, la clé doit rester confidentielle, et doit être transmise au correspondant de façon sûre. Nous tenons aussi à dire qu'en cryptographie symétrique on distingue deux grandes catégories de chiffrements : le chiffrement par bloc (BC : cipher bloc chainning) qui découpe les données en blocs de taille généralement fixe avant de les chiffrer. L'autre est le chiffrement par flot ou chiffrement de flux (en anglais stream cipher) qui arrive à traiter les données de longueur quelconque et il n'a pas besoin de les découper en blocs.
Dans le système de cryptographie symétrique, les algorithmes de chiffrement les plus utilisés sont : DES, 3DES, AES,...

a) Les schémas de Feistel

Le schéma de Feistel suppose qu'on a une fonction f " aléatoire" qui prend comme argument un mot de n bits, et renvoie un mot de n bits (qui donne l'impression d'avoir été choisi au hasard). L'algorithme de chiffrement va procéder en chiffrant des blocs de 2n bits, qu'on partage en 2, partie gauche G, partie droite D. L'image du bloc (G,D) par le schéma de Feistel est le bloc (L,R), avec $L=D$, et $R=G\ \ Xor\ f(D)$ Cette transformation est cette fois bijective, car si on a un tel couple (L,R), on retrouve (G,D) par $D=L$ et $G=R\ \ Xor\ f(L)$ Bien sûr, la partie droite n'a pas été transformée (juste envoyée à gauche). C'est pourquoi on répète le schéma de Feistel un certain nombre de fois (on parle de tours et le DES en comporte 16).



1 xor 1 = 0
0 xor 0 = 0
1 xor 0 = 1
0 xor 1 = 1

On a donc toujours b xor b =0, et donc
a xor b xor b = a.

Ceci explique la réversibilité du schéma de Feistel.

Figure I.2. Le schéma de Feistel. [6]

La plupart des algorithmes à clé secrète de la fin du XX^{ème} siècle étaient des schémas de Feistel.
L'avènement de l'AES marque la fin de la prédominance de tels algorithmes. Pour un lecteur passionné, il peut consulter le détail d'AES à la référence. [7]

b) Le fonctionnement de DES

La clé du DES est une chaîne de 64 bits (succession de 0 et de 1), mais en fait seuls 56 bits servent réellement à définir la clé. Les bits 8, 16, 24, 32, 40, 48, 56, 64 sont des bits de parité (=bits de détection d'erreur). Le 8ème bit est fait en sorte que sur les 8 premiers bits, il y ait un nombre impair de 1. Par exemple, si les 7 premiers bits sont 1010001, le 8ème bit est 0. Ceci permet d'éviter les erreurs de transmission.

Il y a donc pour le DES 2^{56} clés possibles, soit environ 72 millions de milliards possibilités. Les principales phases de l'algorithme sont :

- **Phase 1 : Préparation et diversification de la clé.**
  Le texte est découpé en blocs de 64 bits. On diversifie aussi la clé K, c'est-à-dire qu'on fabrique à partir de K 16 sous-clés $K_1,....,K_{16}$ à 48 bits. Les K_i sont composés de 48 bits de K, pris dans un certain ordre.
- **Phase 2 : Permutation initiale.**
  Pour chaque bloc X, de 64 bits du texte, on calcule une permutation finie $y=P(x)$.
  y est représenté sous la forme $y=G_0D_0$, G_0 étant les 32 bits à gauche de y, D_0 les 32 bits à droite.

- **Phase 3 : Itération**
  On applique 16 rondes d'une même fonction. A partir de $G_{i-1}D_{i-1}$ (pour i allant de 1 à 16), on calcule G_iD_i en posant :
  - $G_i = D_{i-1}$
  - $D_i = G_{i-1}\ Xor\ f(D_{i-1}, K_i)$

  Xor est le ou exclusif bit à bit, et f est une fonction de confusion, suite de substitutions et de permutations.

- **Phase 4 : Permutation finale.**
  On applique à $G_{16}D_{16}$ l'inverse de la permutation initiale. $Z = P^{-1}(G_{16}D_{16})$ est le bloc de 64 bits chiffré à partir de X.

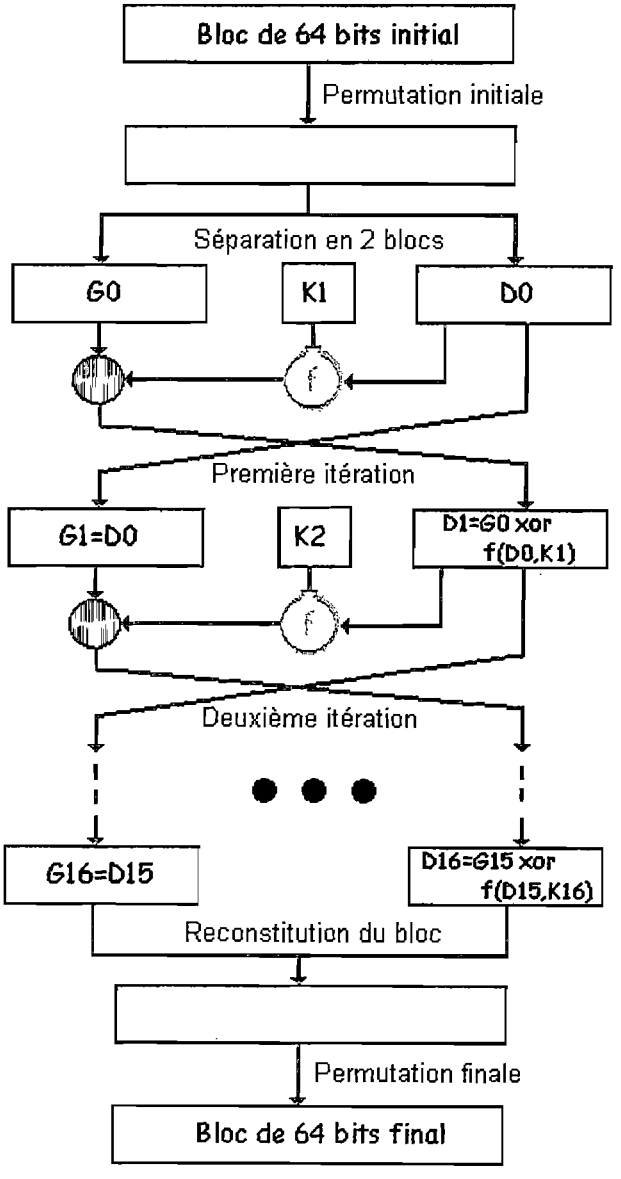


Figure I.3. Les principales phases de l'algorithme du DES. [6]

Toute la sécurité du DES repose sur la fonction de confusion f. C'est en1997 (17 juin) que l'on a démontré que le DES peut être cassé en 3 semaines par une fédération de petites machines sur Internet. Et on estime très officiellement à cette date à quelques secondes le temps nécessaire à un Etat pour percer les secrets d'un message chiffré avec le DES.
La solution a été dans un premier temps l'adoption du triple DES (3DES) c'est-à-dire trois applications de DES à la suite avec 2 clés différentes (d'où une clé de 112 bits) :

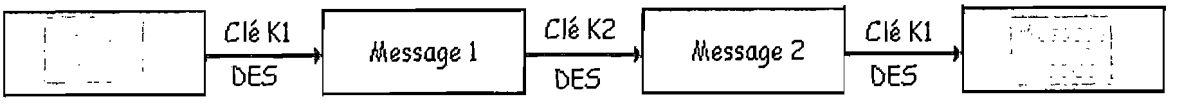


Figure.I.4. Chiffrement triple DES[6]

Si le 3DES est largement suffisant à l'heure actuelle, il est malheureusement trois fois plus lent que le DES [8].

I.2.3.2. Algorithme de chiffrement asymétrique (ou à clé publique)

La cryptographie asymétrique, ou cryptographie à clé publique repose sur l'utilisation d'une clé publique (qui est diffusée) et d'une clé privée (gardée secrète), la première permettant de coder un message et la deuxième de décoder le message que seul le destinataire (en possession de la clé privée) peut décoder, garantissant ainsi la confidentialité du contenu.
Comme son nom l'indique, la clé publique est mise à la disposition de quiconque désire chiffrer un message à envoyer au propriétaire de la dite clé.
La cryptographie asymétrique a été mise au point pour résoudre le problème d'échange de clés dans le sens où il permet d'effectuer le chiffrement sans nécessiter le partage d'une information secrète qu'est la clé.

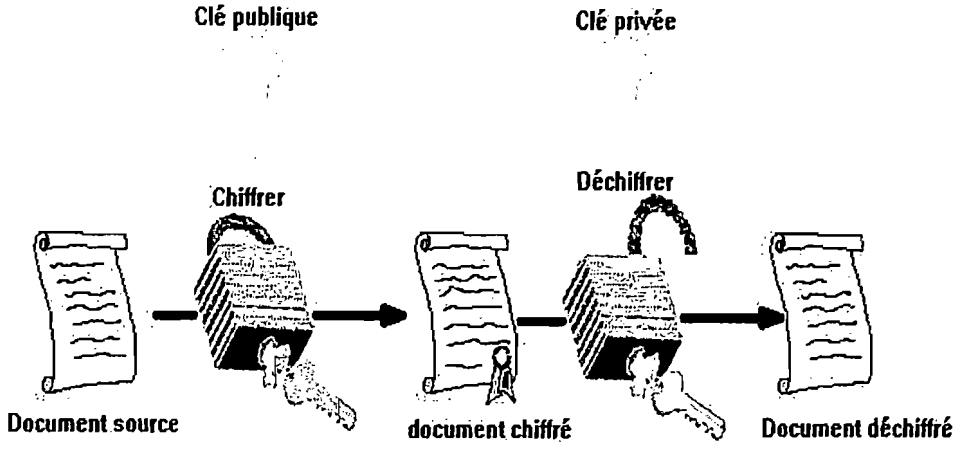


Figure I.5. : Représentation du fonctionnement des clefs asymétriques [5]

Pour le cas, le message ne peut être déchiffré qu'avec la clé privée qui doit être confidentielle.
Les principaux algorithmes de cryptographie asymétrique les plus utilisés sont: RSA, DSA.

- **Fonctionnement de RSA.**

Rivest Shamir Adleman ou RSA (des noms des concepteurs de l'algorithme) est un algorithme asymétrique de cryptographie, très utilisé dans le commerce électronique, et plus généralement pour échanger des données confidentielles sur Internet.

Génération des clés : Choisir p et, q deux nombres premiers distincts.
Notons n leur produit, appelé « module de chiffrement » : $n = p.q$
Calculons l'indicatrice d'Euler de n : $\varphi(n) = (p-1)(q-1)$
Choisir e, un entier premier avec $\varphi(n)$, appelé « exposant de chiffrement ».
Comme e est premier avec $\varphi(n)$, il est, d'après le théorème de Bachet-Bézout **[9]**, inversible mod $\varphi(n)$, c'est-à-dire qu'il existe un entier d tel que $e.d = 1 \bmod \varphi(n)$.
Le couple (n,e) est appelé clé publique, alors que le couple (n,d) est appelé clé privée.

Si M est un entier inférieur à n représentant un message, alors le message chiffré sera représenté par $C \equiv M^e \pmod n$.

Pour déchiffrer C, on calcule
$C^d \pmod n \equiv (M^e)^d \pmod n \equiv M^{ed} \pmod n \equiv M$

On constate que pour chiffrer un message, il suffit de connaître e et n. En revanche pour déchiffrer, il faut d et n .
La sécurité de la RSA repose sur le problème de la factorisation qui consiste à trouver les facteurs premiers d'un grand nombre entier. Le temps que prend cette factorisation croît exponentiellement avec la longueur de l'entier. En effet le 12 décembre 2009, le plus grand nombre factorisé en utilisant une méthode de calculs distribués, était long de 768 bits. Les clés RSA sont habituellement de longueur comprise entre 1024 et 2048 bits. Quelques experts croient possible que des clés de 1024 bits seront cassées dans un proche avenir, mais peu voient un moyen de casser de cette manière des clés de 4096 bits dans un avenir prévisible. On peut donc présumer que RSA est sûr sur cette base si la taille de la clé est suffisamment grande.

I.2.3.3. Les fonctions de hachage

« Une fonction de hachage, appelée aussi fonction de hachage à sens unique ou "one-way hash function" en anglais, est une fonction utilisée en cryptographie pour réduire la taille des données à traiter par la fonction de cryptage »[10]. En effet, la caractéristique principale d'une fonction de hachage est de produire un haché des données, c'est-à-dire un condensé de ces données. Ce condensé est de taille fixe dont la valeur diffère suivant la fonction utilisée. Le but d'un condensé est de représenter des données de façon certaine tout en réduisant la taille utile qui sera réellement chiffrée. Prenons l'exemple de la cryptographie asymétrique; tout le monde admet qu'elle est très sûre, fiable et durable. Néanmoins, sa complexité (calcul sur des nombres premiers de plusieurs centaines de chiffres par exemple) entraîne une inévitable lourdeur d'emploi. On évite donc de l'utiliser pour de grandes masses de données.

Par contre imaginons que nous souhaitions envoyer un fichier par mail, mais que ce fichier est de taille importante. Nous souhaitons de plus rassurer le destinataire sur la provenance de ce fichier et sur son contenu. Plutôt que de chiffrer notre fichier directement

avec notre clé privée, nous allons hacher notre fichier et chiffrer le condensé obtenu avec notre clé privée. Nous enverrons ensuite notre fichier original ainsi que le condensé chiffré (la signature) à notre destinataire. Celui-ci va, lors de la réception, hacher d'une part le fichier reçu et d'autre part déchiffrer le condensé reçu (au moyen de notre clé publique).

S'il n'y a pas égalité entre les 2 résultats, cela signifiera :

- soit que la signature n'est plus la notre, donc que quelqu'un a intercepté le fichier (pour le modifier ou le remplacer, etc.)

- soit que le fichier n'est plus le même que l'original (mais la signature n'a pas été remplacée); dans ce cas, le hachage ne peut plus donner le même condensé ce qui conduit au rejet lors du test de comparaison.

Dans les 2 cas, ni l'intégrité ni l'authentification du fichier n'ont été vérifiées. Il ne faut donc pas faire confiance au fichier. Nous voyons comment dans ce cas simple, l'utilisation d'une fonction de hachage permet de s'assurer de l'intégrité des données et indirectement de les authentifier. Il existe bien sûr de nombreuses autres applications pour les fonctions de hachage, comme les MACs (message authentication code), certificats, etc.

Les fonctions de hachage usuelles sont :

- MD4 et MD5 (Message Digest) furent développées par Ron Rivest. MD5 produit des hachés de 128 bits en travaillant les données originales par blocs de 512 bits.
- SHA-1 (Secure Hash Algorithm 1), comme MD5, est basé sur MD4. Il fonctionne également à partir de blocs de 512 bits de données et produit par contre des condensés de 160 bits en sortie. Il nécessite donc plus de ressources que MD5.
- SHA-2 (Secure Hash Algorithm 2) a été publié récemment et est destiné à remplacer SHA-1. Les différences principales résident dans les tailles de hachés possibles : 256, 384 ou 512 bits. Il sera bientôt la nouvelle référence en termes de fonction de hachage.
- RIPEMD-160 (Ripe Message Digest) est la dernière version de l'algorithme RIPEMD. La version précédente produisait des condensés de 128 bits mais présentait des failles de sécurité importantes. La version actuelle reste pour l'instant sûre; elle produit comme son nom l'indique des condensés de 160 bits. Un dernier point la concernant est sa relative gourmandise en termes de ressources et en comparaison avec SHA-1 qui est son principal concurrent.**[10]**

I.2.3.4. Code d'authentification de message

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 pas subi aucune modification, après une transmission par exemple. Le concept est relativement semblable aux fonctions de hachage. Il s'agit 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.

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, 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.

La fonction HMAC est définie comme suit :

$$HMAC_K(m) = h\big((K \oplus opad) \,||\, h\big((K \oplus ipad) \,||\, m\big)\big)$$

Avec :

- h : une fonction de hachage itérative,
- K : la clé secrète complétée avec des zéros pour qu'elle atteigne la taille de bloc de la fonction h
- m : le message à authentifier,
- "||" désigne une concaténation et "⊕ " un ou exclusif,
- *ipad* et *opad*, chacune de la taille d'un bloc, sont définies par : $ipad = 0\times363636.....36$ et $opad = 0\times5c5c5c.....5c$ Donc, si la taille de bloc de la fonction de hachage est 512, *ipad* et *opad* sont 64 répétitions des octets, respectivement, 0x36 et 0x5c.

I.2.4. L'intégrité et la non répudiation des données

Quand nous recevons un message, nous ne sommes pas sûrs que le message reçu soit celui de l'expéditeur déclaré. De plus, nous ne savons pas si le message n'a pas été modifié ou changé au passage par un tiers mal intentionné mais aussi l'expéditeur peut nier l'avoir envoyé. La solution pour que l'intégrité et la non répudiation soient assurées est apportée par l'utilisation d'une signature numérique.

I.2.4.1. Signature numérique

La signature numérique est un mécanisme permettant d'identifier l'auteur d'un document électronique et de garantir son intégrité, et cela d'une façon similaire à la signature manuscrite d'un document papier **[4]**. Un mécanisme de signature numérique doit présenter les propriétés suivantes :

- Il doit permettre au lecteur d'un document d'identifier la personne ou l'organisme qui a apposé sa signature.

- Il doit garantir que le document n'a pas été altéré entre l'instant où l'auteur l'a signé et le moment où le lecteur le consulte.

Pour qualifier un document d'authentique, il faut qu'il remplisse les conditions suivantes :

-Authentique : L'identité du signataire doit pouvoir être retrouvée de manière certaine.
-Infalsifiable : La signature ne peut pas être falsifiée. Quelqu'un ne peut pas se faire passer pour un autre.
-Non réutilisable: La signature n'est pas réutilisable. Elle fait partie du document signé et ne peut être déplacée sur un autre document.
-Inaltérable : Un document signé est inaltérable. Une fois qu'il est signé, on ne peut plus le modifier.
-Irrévocable : La personne qui a signé ne peut le nier.

La signature électronique est apparue avec la cryptographie asymétrique. Elle se différencie de la signature écrite par le fait que la signature manuscrite est composée du patronyme sous un graphisme particulier. Elle relie physiquement le patronyme à la personne physique qui l'exécute, et cette combinaison est considérée comme unique puisque l'écriture est une caractéristique biométrique de l'individu. Elle est durable dans le temps. Mais la signature électronique est fonction du contenu du message. Pour l'obtenir, on applique une fonction de hachage au message à signer puis on chiffre le haché. Le résultat obtenu est la signature électronique.

I.2.4.2. Algorithmes de signature numérique

Les algorithmes de signature numérique permettent de détecter des atteintes à l'intégrité d'un message et de prouver l'authenticité de l'émetteur. D'une façon générale, on soumet le message à signer à une fonction de condensation à sens unique puis on signe le condensé. Les deux principaux algorithmes utilisés sont le RSA et le DSA. DSA est un algorithme de signature numérique à clé publique. Contrairement à RSA, il ne permet pas de faire du chiffrement. La taille de la clé peut être définie de 512 à 1024 bits. DSA est considéré comme sûr avec des clés de 1024 bits. De telles clés nécessiteraient un temps long pour être cassées par des personnes malveillantes.

Nous l'avons déjà signalé, la signature électronique est un procédé permettant aussi de garantir la non répudiation, c'est-à-dire que l'expéditeur a bien envoyé le message, autrement dit elle empêche l'expéditeur de nier avoir expédié le message ou que l'auteur qui a signé un document est bien l'auteur du document.

Il faut signaler qu'il existe deux classes de schémas de signatures électroniques à savoir :

La signature électronique avec appendice où le message original doit être fournit à l'algorithme de vérification et la signature électronique avec recouvrement où le message original est récupéré à partir de la signature électronique. **[11]**

a) Signature avec appendice

Soient M un ensemble fini de messages, S un ensemble fini de signatures et K un ensemble fini de paires de clés (publique et privée).

Pour toute paire de clé publique et privée (k,k'), il existe un algorithme de signature avec appendice $Sig_{k'}$ et un algorithme de vérification correspondant Ver_k tels que la signature d'un message x est :

$y = Sig_{k'}(x) : M \rightarrow S$

et

$Ver_k(x,y) : M \times S \rightarrow \{vrai, faux\}$

b) Signature avec recouvrement

Soient M un ensemble fini de messages, M_s un ensemble fini de messages pouvant être signés, S un ensemble fini de messages signés et K un ensemble fini de paires de clés (publiques et privées).

Pour toute paire de clés publiques et privées (k,k') il existe un algorithme de signature avec recouvrement $Sig_{k'}$ qui applique $M_s \rightarrow S$, une fonction de redondance $R : M \rightarrow M_s$ et un algorithme de vérification correspondant $Ver_k : S \rightarrow M_s$ tels que la signature d'un message x est : $y = Sig_{k'}(R(x))$ et $x' = Ver_k(x)$.

Si x' n'appartient pas à M_s alors la signature est rejetée, sinon la signature est acceptée et le message $x = R^{-1}(x')$ est récupéré.

I.2.5. Mécanismes d'identification dérivée des méthodes cryptographiques

a)Infrastructure à clés publiques

Une infrastructure à clés publiques (ICP) ou infrastructure de Gestion de Clés (IGC) ou encore Public Key Infrastructure (PKI), est un ensemble de composants physiques (des ordinateurs, des équipements cryptographiques logiciels ou matériel), de procédures humaines (vérifications, validation) et de logiciels (système et application) en vue de gérer les certificats numériques ou certificats électroniques. **[12]**

Une infrastructure à clés publiques délivre un ensemble de services pour le compte de ses utilisateurs. Ces services sont les suivants :

- enregistrement des utilisateurs (ou équipement informatique) ;
- génération de certificats ;
- renouvellement de certificats ;
- révocation de certificats ;
- publication de certificats ;
- publication des listes de révocation (comprenant la liste des certificats révoqués) ;
- identification et authentification des utilisateurs (administrateurs ou utilisateurs qui accèdent à l'ICP).

b) Rôle d'une infrastructure à clé publique

Une ICP délivre des certificats numériques. Les ICP se composent de quatre entités distinctes à savoir :

- L'autorité de certification (AC ou CA) qui a pour mission de signer les demandes de certificat (CSR : Certificate Signing Request) et de signer les listes de révocation (CRL : Certificate Revocation List).
- L'autorité d'enregistrement (AE ou RA) qui a pour mission de générer les certificats, et d'effectuer les vérifications d'usage sur l'identité de l'utilisateur final.
- L'autorité de dépôt (Repository) qui a pour mission de stocker les certificats numériques ainsi que les listes de révocation (CRL).
- L'entité finale (EE : End Entity). C'est l'utilisateur ou le système qui est le sujet d'un certificat (En général, le terme « entité d'extrémité » est préféré au terme « sujet » afin d'éviter la confusion avec le champ Subject).

En complément, on peut ajouter l'autorité de séquestre (Key Escrow). Cette entité a un rôle particulier. En effet lorsqu'on génère des certificats de chiffrement, on a l'obligation légale de fournir aux autorités un moyen de déchiffrer les données chiffrées pour un utilisateur de l'ICP. C'est là qu'intervient le séquestre, cette entité a pour mission de stocker de façon sécurisée les clés de chiffrement qui ont été générées par l'ICP, pour pouvoir les restaurer le cas échéant.

c) Les certificats numériques

Un certificat numérique est un « document numérique » qui permet d'authentifier une clé publique, c'est-à-dire de prouver qu'elle appartient à une personne ou une entité.

Le certificat est une véritable carte d'identité numérique : elle regroupe toutes les informations sur le titulaire, qu'il s'agisse d'un particulier ou d'une société, dans un fichier numérique respectant un format spécifique. Il permet de mettre en relation une clé publique avec l'identité du titulaire.
Les certificats peuvent servir dans plusieurs cas :
*Certificat client : Il permet d'authentifier le client auprès d'un serveur. Ce dernier peut ainsi attribuer des droits particuliers à son interlocuteur.

*Certificat serveur : utilisé notamment dans le protocole SSL, il permet au client de s'assurer de l'identité de la société à laquelle il a affaire.
Notons qu'un certificat comporte une durée de validité, par exemple un ou deux ans.
Ici, nous pouvons nous demander ce que représentent le CLIENT et le SERVEUR. Lorsque deux ordinateurs sont reliés avec un réseau quelconque, celui qui attend la connexion joue le rôle de serveur et celui qui initie la connexion est le client. Le client s'applique donc à l'ordinateur qui est contrôlé par l'utilisateur.

d) Structure d'un certificat numérique

Les certificats sont des petits fichiers divisés en deux parties : La partie contenant les informations et la partie contenant la signature de l'autorité de certification.
La structure des certificats est normalisée par le standard X.509 de l'UIT (plus exactement X.509v3), qui définit les informations contenues dans le certificat à savoir:

- La version de X.509 à laquelle le certificat correspond ;
- Le numéro de série du certificat ;
- L'algorithme de chiffrement utilisé pour signer le certificat ;
- Le nom (DN, pour Distinguished Name) de l'autorité de certification émettrice ;
- La date de début de validité du certificat ;
- La date de fin de validité du certificat ;
- L'objet de l'utilisation de la clé publique ;
- La clé publique du propriétaire du certificat ;
- La signature de l'émetteur du certificat (thumbprint).

L'ensemble de ces informations (informations + clé publique du demandeur) est signé par l'autorité de certification.

Création des certificats électroniques

Il existe deux façons distinctes de créer des certificats électroniques : le mode centralisé et le mode décentralisé.

- le mode décentralisé est le mode le plus courant : il consiste à faire créer, par l'utilisateur (ou, plus exactement par son logiciel ou carte à puce) le biclé cryptographique et à joindre la partie publique de la clef dans la CSR. L'Infrastructure n'a donc jamais connaissance de la clé privée de l'utilisateur, qui reste confinée sur son poste de travail ou dans sa carte à puce.

- le mode centralisé consiste en la création du biclé par l'AC : au début du cycle de la demande, la CSR ne contient pas la clé publique, c'est l'AC qui la produit. Elle peut ainsi avoir de bonnes garanties sur la qualité de la clé (aléa) et peut en détenir une copie protégée. En revanche, il faut transmettre à l'utilisateur certes son certificat (qui ne contient que des données publiques) mais aussi sa clé privée. L'ensemble de ces deux données est un fichier créé sous le format PKCS#12. Son acheminement vers l'utilisateur doit être entrepris avec beaucoup de précaution et de sécurité, car toute personne mettant la main sur un fichier PKCS#12 peut détenir la clé de l'utilisateur.

Le mode décentralisé est préconisé pour les certificats d'authentification (pour des questions de coût, parce qu'il est plus simple de refaire un certificat en mode décentralisé qu'à recouvrer une clé) et de signature (parce que les conditions d'exercice d'une signature juridiquement valide prévoient que le signataire doit être le seul possesseur de la clé : en mode décentralisé, l'ICP n'a jamais accès à la clé privée).

Le mode centralisé est préconisé pour les certificats de chiffrement, car, lorsqu'un utilisateur a perdu sa clé (par exemple, sa carte est perdue ou dysfonctionne), un opérateur peut, au terme d'une procédure de recouvrement, récupérer la clé de chiffrement et la lui remettre. Chose qui est impossible à faire avec des clés qui n'ont pas été séquestrées. Comme un certificat a la naissance (la création) , il a aussi la fin.

Il existe deux possibilités pouvant occasionner la fin de la vie d'un certificat numérique :

- le certificat numérique expire (chaque certificat numérique contient une date de « naissance » et une date de « péremption »).
- le certificat est révoqué, pour quelque raison que ce soit (perte de la clé privée associée, etc.) et dans ce cas, l'identifiant du certificat numérique est ajouté à une liste

de certificats révoqués (CRL pour Certificate Revocation List) pour informer les applications qu'elles ne doivent plus faire confiance à ce certificat.

e) Obtention d'un certificat numérique

Un certificat numérique naît après qu'une demande de certificat a abouti. Une demande de certificat est un fichier numérique (appelé CSR pour Certificate Signing Request) qui est soumis à une autorité d'enregistrement par un utilisateur final ou par un administrateur pour le compte d'un utilisateur final.

Cette demande de certificat est examinée par un Opérateur d'Autorité d'Enregistrement. Cette position est une responsabilité clé : c'est lui qui doit juger de la légitimité de la demande de l'utilisateur et accorder, ou non, la confiance de l'organisation. L'Opérateur doit suivre une série de procédures, plus ou moins complètes, consignées dans deux documents de référence qui vont de pair avec la création d'une IGC qui sont la Politique de Certification (PC) et la Déclaration des Pratiques de Certification (DPC). Ces documents peuvent exiger, en fonction des enjeux de la certification, des vérifications plus ou moins poussées : rencontre face-à-face, validation hiérarchique, etc. L'objectif de l'Opérateur d'AE est d'assurer que les informations fournies par l'utilisateur sont exactes et que ce dernier est bien autorisé à solliciter la création d'un certificat. Une fois son opinion formée, l'Opérateur de l'AE valide la demande ou la rejette. S'il la valide, la demande de certificat est alors adressée à l'Autorité de Certification (AC). L'AC vérifie que la demande a bien été validée par un Opérateur d'AE digne de confiance et, si c'est le cas, il signe la CSR. Une fois signée, une CSR devient un certificat.

I.3. Conclusion

L'internet est aujourd'hui préféré par beaucoup d'utilisateurs comme moyen de communication. Les données échangées sont de grande importance qu'elles nécessitent d'être sécurisées. Dans ce chapitre, nous l'avons expliqué la sécurité informatique est une nécessité pour protéger les transactions effectuées avec l'Internet. La S.I se fondent sur un grand nombre d'objectifs lui permettant d'atteindre ces premiers. Les principaux objectifs de la SI sont l'intégrité, la confidentialité, l'identification et la non répudiation. Pour atteindre ces objectifs, la S.I se base sur la cryptographie. La cryptographie donne des moyens permettant d'atteindre les objectifs de la S.I tels que : le chiffrement, le hachage, la signature numérique et le certificat

La question qui se pose ici est de savoir par quel moyen les fonctionnalités offertes par la cryptographie pour assurer la S-I sont mises en place. Cela est garanti par l'utilisation des protocoles de sécurité et parmi ces protocoles, le SSL est aujourd'hui le plus préféré par les utilisateurs. Le chapitre deux sera consacré au protocole SSL pour voir comment, et par quel moyen il vient assurer la S-I.

CHAPITRE II.APPORT DU PROTOCOLE SSL AUX OBJECTIFS DE LA SECURITE INFORMATIQUE

II.1. Introduction au protocole SSL

II.1.1. Présentation générale

Le protocole SSL, qui tient pour Secure Socket Layer, littéralement « couche de sockets sécurisée », a été inventé par la société Netscape au milieu des années 1990. La première version de SSL a été développée en 1994. L'objectif de Netscape était de créer un canal sécurisé où les données pourraient transiter entre un client et un serveur, indépendamment de la plateforme et du système d'exploitation.
Quoique SSL soit destiné à l'origine uniquement à sécuriser les transactions entre un client et un serveur web, la spécification a été conçue de façon à ce que les autres protocoles de niveau application puissent l'exploiter. La spécification SSL 1.0 ne fut diffusée qu'en interne et aucune application ne supportera SSL 1.0.

En février 1995, Netscape publie la version 2.0 du protocole. Il est implémenté dans la première version de son client web Navigator. SSL 2.0 se fonde sur l'authentification du serveur par le poste client et sur l'utilisation d'un certificat serveur au format X.509 v3. Cette authentification ne nécessite, du côté du poste client, que des calculs en clé publique.
En novembre 1996, Netscape publie la version 3.0 du protocole. Par rapport à SSL 2.0, SSL 3.0 offre en plus la capacité, pour le serveur, d'authentifier le client. Dans ce cas, le client doit pouvoir d'une part exploiter sa clé privée et d'autre part fournir son certificat au format X.509 v3.

L'IETF (Internet Engineering Task Force) propose à son tour un protocole de transfert sécurisé basé sur les concepts de SSL, baptisé TLS 1.0 (parfois nommée SSL 3.1) et décrit dans la RFC 2246. Elle rachète le brevet de Netscape sur le protocole SSL en 2001. Puis en juin 2003, des extensions sont proposées pour TLS sous la forme d'une nouvelle RFC. La dernière RFC 4366, mettant à jour la précédente, est sortie en Avril 2006. **[13]**
A l'heure actuelle, les protocoles SSL 2.0, SSL 3.0 et TLS 1.0 sont utilisés. Nous nous limiterons au protocole SSL, sachant que les grands principes restent vrais pour TLS.

II.1.2. Principes de fonctionnement

Avant d'entrer dans le détail du principe de fonctionnement du protocole SSL; voyons d'abord la notion des modèles en couche (OSI et TCP/IP).

a) Le Modèle OSI

Le modèle OSI (de l'anglais Open Systems Interconnections tenant pour Interconnexion de systèmes ouverts) est un modèle de communications entre ordinateurs proposé par l'ISO (Organisation Internationale de normalisation). Il décrit les fonctionnalités nécessaires à la

communication et à l'organisation de ces fonctions. Le modèle OSI comporte 7 couches qui sont parfois réparties en 2 groupes: les 4 couches inférieures sont orientées communication et sont typiquement fournies par un système d'exploitation; les 3 couches supérieures sont orientées application et réalisées par des bibliothèques ou un programme spécifique. **[13], [14]**

1 - La couche physique

La couche physique s'occupe de la transmission des bits de façon brute sur un canal de communication. Cette couche doit garantir la parfaite transmission des données. Concrètement, cette couche doit normaliser les caractéristiques électriques (un bit 1 doit être représenté par une tension de 5 V, par exemple), les caractéristiques mécaniques (forme des connecteurs, de la topologie...), les caractéristiques fonctionnelles des circuits de données et les procédures d'établissement, de maintien et de libération du circuit de données.

2 - La couche liaison de données

Son rôle est un rôle de "liant": elle va transformer la couche physique en une liaison a priori exempte d'erreurs de transmission pour la couche réseau. Elle fractionne les données d'entrée de l'émetteur en trames, transmet ces trames en séquence et gère les trames d'acquittement renvoyées par le récepteur. Rappelons que pour la couche physique, les données n'ont aucune signification particulière. La couche liaison de données doit donc être capable de reconnaître les frontières des trames. Cela peut poser quelques problèmes, puisque les séquences de bits utilisées pour cette reconnaissance peuvent apparaître dans les données.

3 - La couche réseau

C'est la couche qui permet de gérer le sous-réseau, i.e. le routage des paquets sur ce sous-réseau et l'interconnexion des différents sous-réseaux entre eux. Au moment de sa conception, il faut bien déterminer le mécanisme de routage et de calcul des tables de routage (tables statiques ou dynamiques...).

4- Couche transport

Cette couche est responsable du bon acheminement des messages complets au destinataire. Le rôle principal de la couche transport est de prendre les messages de la couche session, de les découper s'il le faut en unités plus petites et de les passer à la couche réseau, tout en s'assurant que les morceaux arrivent correctement de l'autre côté. Cette couche effectue donc aussi le réassemblage du message à la réception des morceaux.

5- La couche session

Cette couche organise et synchronise les échanges entre tâches distantes. Elle réalise le lien entre les adresses logiques et les adresses physiques des tâches réparties. Elle établit également une liaison entre deux programmes d'application devant coopérer et commande leur dialogue (qui doit parler, qui parle...). Dans ce dernier cas, ce service d'organisation s'appelle la gestion du jeton. La couche session permet aussi d'insérer des points de reprise dans le flot de données de manière à pouvoir reprendre le dialogue après une panne.

6- La couche présentation

Cette couche s'intéresse à la syntaxe et à la sémantique des données transmises: c'est elle qui traite l'information de manière à la rendre compatible entre tâches communicantes. Elle va assurer l'indépendance entre l'utilisateur et le transport de l'information. Typiquement, cette couche peut convertir les données, les reformater, les crypter et les compresser.

7- La couche application

Cette couche est le point de contact entre l'utilisateur et le réseau. C'est donc elle qui va apporter à l'utilisateur les services de base offerts par le réseau, comme par exemple le transfert de fichier, la messagerie...

b) Modèle TCP/IP

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

1. **Couche Accès réseau** : elle spécifie la forme sous laquelle les données doivent être acheminées quel que soit le type de réseau utilisé.
2. **Couche Internet** : elle est chargée de fournir le paquet de données (datagramme).
3. Couche Transport : elle assure l'acheminement des données, ainsi que les mécanismes permettant de connaître l'état de la transmission.
4. Couche Application : elle englobe les applications standard du réseau (Telnet, SMTP, FTP, ...).

Revenons alors au principe de fonctionnement de SSL. SSL est un protocole qui intervient entre TCP/IP et les différents protocoles applicatifs tels que SMTP, IMAP, POP,... Une de ses principales caractéristiques est de n'est pas être lié à un type protocole particulier mais de pouvoir être utilisé pour insérer une couche de communication sécurisée entre un CLIENT et un SERVEUR, quelle que soit l'application employée.
Ainsi, on retrouve leur positionnement sur le schéma suivant représentant le modèle OSI et TCP/IP.

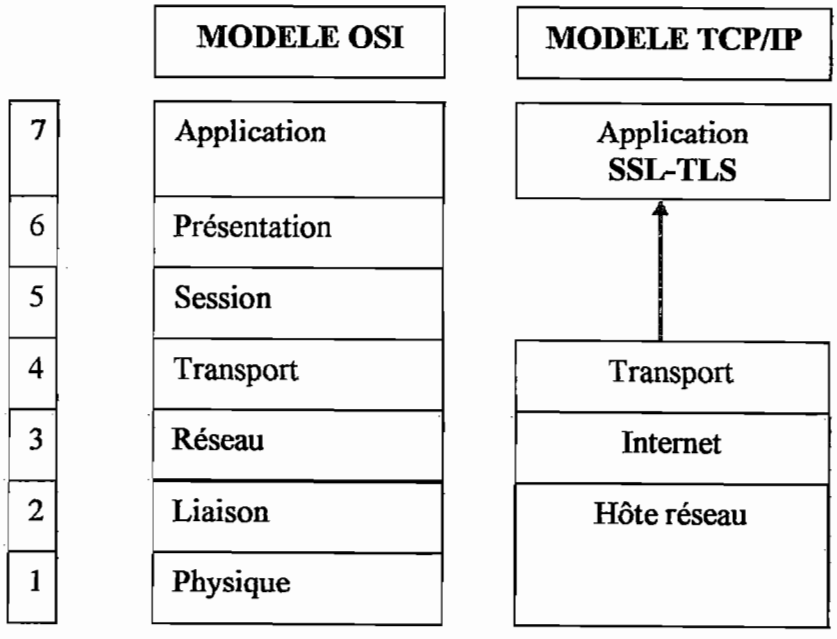


Figure II.1 : Positionnement de SSL-TLS [13]

SSL et TLS se comportent en effet comme une couche intermédiaire supplémentaire, car ils sont indépendants du protocole utilisé au niveau application. Cela signifie donc qu'il peut aussi bien être employé pour sécuriser une transaction web, l'envoi ou la réception d'email, etc. SSL et TLS sont donc transparents pour l'utilisateur et ne nécessitent pas l'emploi de protocoles de niveau Application spécifiques.
Les trois principales caractéristiques du protocole SSL, en matière de sécurité, sont les suivantes :

- L'identification du serveur : Le client lorsqu'il se connecte, via SSL, à un serveur distant, peut avoir la preuve de l'identité du serveur en question grâce à l'utilisation de certificat numérique.
- L'identification du client : Cette fois-ci, c'est le serveur qui peut s'assurer de l'identité de la personne connectée. Cette fonctionnalité n'est généralement pas utilisée dans le cadre du commerce électronique mais plutôt lorsqu'il s'agit de sécuriser l'accès à des informations sensibles, comme des relevés bancaires ou l'intranet d'une entreprise.
- Le chiffrement de la session : L'objectif est d'empêcher que les informations transmises entre le client et le serveur ne soient pas interceptées ou même modifiées par des tiers. Des mécanismes de chiffrement sont utilisés dans ce but.

Il faut savoir que le protocole SSL regroupe quatre sous-protocoles à savoir: SSLHandshake, SSLRecord, CipherSpecSuite et alert.

• Le sous protocole SSLHandshake intervient lorsqu'un client et un serveur veulent ouvrir une session SSL entre eux. Il consiste en l'échange d'un certain nombre de messages associé à des traitements effectués simultanément sur le client et le serveur.

• Quant au SSLRecord, il décrit le format des données, issues des couches applicatives, qui seront transmises au cours d'une session protégée par SSL.

• Le sous protocole ChangeCipherSpec(CCS) qui comprend un et un seul message (1 octet) qui porte le même nom que le protocole. Il permet d'indiquer au protocole record la mise en place des algorithmes de chiffrement qui viennent d'être négociés.

• Le sous protocole SSL Alert génère des messages d'alerte suite aux erreurs que peuvent s'envoyer le client et le serveur. Ces messages sont composés de 2 octets, le premier étant soit fatal soit warning. Si le niveau de criticité du message est fatal, la connexion SSL est abandonnée. Le deuxième octet est utilisé pour le code d'erreur. Nous donnons ici la liste des messages d'erreurs du protocole alert.
Les erreurs fatales sont:

- bad_record_mac: réception d'un MAC erroné.
- décompression_failure: les données appliquées à la fonction de compression sont invalides.
- handshake_failure: impossibilité de négocier les bons paramètres.
- illégal_parameter: un paramètre échangé au cours du protocole Handshake ne correspondait pas avec les autres paramètres.
- Unexpected_message: message non reconnu.

Les warnings sont:

- Bad_certificate: le certificat n'est pas bon.
- Certificate_expired: certificat périmé.
- Certificat_revoked: certificat révoqué.
- Certificat_unknown: certificat invalide pour des raisons précisées au dessus.

- Close_notify: la fin d'une connexion.
- No_certificate: réponse négative à une demande de certificat.
- Unsupported_certificate: le certificat reçu n'est pas reconnu.[15]

Avant d'entrer dans le détail de ces deux premiers protocoles (SSLRecord et SSLHandshake,) considérés comme les principaux sous-protocoles de SSL, définissons d'abord la notion de suites cryptographiques.

II.2. Les suites cryptographiques

Les algorithmes cryptographiques utilisés dans SSL sont classés selon différentes catégories, appelées « cipher suites », ou « suites cryptographiques », associant un algorithme de chiffrement, avec une clé de longueur déterminée, à un algorithme de condensation pour assurer les fonctions d'authentification. De plus, les algorithmes d'échange de clés sont également utilisés afin de générer une clé de session qui servira à chiffrer les transactions. A cet effet, on utilise le plus souvent l'algorithme RSA.
Nous montrons ici les différentes suites cryptographiques utilisant RSA pour échanger des clés et définis pour SSL v3.0 :

- Suite cryptographique la plus forte : Chiffrement triple DES (3DES) avec une clé de 168 bits et une authentification de message via SHA (Secure Hash Algorithm).
- Suite cryptographique forte : Chiffrement RC4 avec une clé de 128 bits et une authentification basée sur MD5 et DES à 56 bits avec SHA .Autorisée seulement aux Etats-Unis.
- Suite cryptographique exportable : Chiffrement RC4 avec une clé de 40 bits et une authentification MD5 et RC2 avec une clé de 40 bits. Ces suites sont incorporées aux versions exportables des navigateurs.
- Suite cryptographique la plus faible : Pas de chiffrement, authentification MD5.Cette suite n'assure que l'authentification et la détection des modifications, pas la confidentialité des informations transmises.[16]

Nous venons de voir que le protocole SSL possède quatre sous protocoles. Alors on peut se demander comment ces derniers interviennent en matière de sécurité informatique. C'est ce que nous allons expliquer dans la section qui suit.

II.3. Le sous protocole SSLHandshake

Toute session est initiée grâce à un sous-protocole de SSL appelée SSLHandshake. Ce dernier a pour objectif de permettre l'identification du serveur et optionnellement du client, grâce à l'utilisation de certificats, puis la génération de clés secrètes afin de chiffrer et détecter des modifications de données au cours de la session.
C'est durant la phase de négociation, exécutée grâce à SSLHandshake, qu'est sélectionnée la suite la plus forte, reconnue simultanément par le client et par le serveur considérés.

Enumérons les principales phases de ce protocole, dans le cas où seul le serveur est identifié :

1. Le client, qui souhaite établir une connexion sécurisée par SSL, envoie au serveur distant un message « ClientHello » contenant la version de SSL qu'il supporte, la liste des suites cryptographiques qu'il reconnaît, par ordre de préférence, une valeur aléatoire, d'une longueur de 28 bits, ainsi que d'autres paramètres tels qu'un identifiant de session.

2. Le serveur envoie un message « ServerHello », ayant une structure identique à celui du client. Il envoie également son certificat, de type X.509.
3. Le client vérifie l'authenticité du certificat serveur. Si cette opération se déroule correctement, le serveur est authentifié et on peut passer à l'étape suivante. Sinon, l'ouverture de la session échoue.
4. Le client établit un pré-secret maître pour la session, avec la collaboration éventuelle du serveur, selon l'algorithme utilisé. Il chiffre ce pré-secret à l'aide de la clé publique du serveur qu'il a obtenue grâce au certificat de ce dernier, et l'envoie au serveur.
5. Le serveur utilise sa clé privée afin de déchiffrer le pré-secret. Le client et le serveur génèrent alors, chacun de son côté, un secret maître d'une longueur de 48 bits, à partir du pré-secret.
6. Le client et le serveur génèrent les clés de session à partir du secret maître qu'ils partagent, permettant de chiffrer les informations transmises et de vérifier leur intégrité.
7. Le client envoie un message informant le serveur de la suite cryptographique qu'il utilise ainsi qu'un message marquant la fin du protocole, chiffré à l'aide de la session générée à l'étape 6.
8. Le serveur envoie le même type de message au client.
9. Le sous-protocole SSLHandshake est terminé. Le client et le serveur peuvent communiquer de façon sécurisée.
10. Echange des données entre le Client et le Serveur.

Voyons maintenant comment se fait la génération des clés de session.

Génération des clés de session

Au cours du sous-protocole handshake, le client génère un pré-secret (premaster-secret) qu'il chiffre et envoie au serveur. C'est à partir de ce pré-secret maître que le serveur et le client vont générer chacun de son côté le secret –maître (master-secret) de la façon suivante :

$$Master-\sec ret = PRF(premaster\ \sec ret, clientrandom + serverrandom)$$

PRF est une fonction aléatoire qui est utilisée pour créer, à partir des secrets, des blocs qui seront découpés pour définir des clés (soit de hachage soit de chiffrement).
Avec ce Master-secret le protocole record a besoin d'un algorithme pour générer des clés, les IVs(Initials values pour le mode CBC) et les secrets du MAC. Il utilise pour cela la fonction pseudo aléatoire PRF :

$$Key-bloc = PRF(master-\sec ret, serverrandom + clientrandom)$$

Du $Key-block$ on déduit :
-le Client_write_MAC _secret key: clé secrète utilisée par le client pour calculer les MACs
-le server write MAC secret key: clé secrète utilisée par le serveur pour calculer les MACs
-le client_write_key : clé symétrique utilisée par le client pour le chiffrement des données
- le server_write_key : clé symétrique utilisée par le serveur pour le chiffrement des données.
-le Client_write_IV : vecteurs d'initialisations pour le chiffrement du côté client
–le Server_write_IV : vecteurs d'initialisations pour le chiffrement du côté du serveur.

Ces clés sont des paramètres qui définissent une connexion SSL (un lien logique entre un client et un serveur) et qui sont rafraîchis pendant une session.

Soit A(.) une fonction définie comme suit:

$A(0) = seed$;

$A(i) = HMAChash(\sec ret, A(i-1))$;

On définit la fonction d'expansion des données *P_hash* de la façon suivante:

$$Phash(\sec ret, seed) = HMAChash(\sec ret, A(1) + data + HMAChash(\sec ret, A(2) + data) + ...$$

Suivant la quantité de données que l'on veut avoir comme outputs de *P_hash*, on appellera itérativement la fonction HMAC autant de fois que c'est nécessaire. Par exemple si on veut produire 64 octets avec la fonction P_SHA-1 (la longueur de l'output est de 20 octets) on appellera la fonction HMAC jusqu'à $A(4)$. On aura en tout 80 octets et on laissera tomber les 16 octets de la dernière itération pour ne garder que les 64 octets nécessaires.
On peut alors définir la fonction PRF pour deux variables (secret et seed) de la façon suivante:

$$PRF(\sec ret, seed) = P\_MD5(\sec ret) + P\_SHA-1(\sec ret, seed).$$

Identification du serveur

L'objectif de l'identification du serveur est de vérifier l'identité du serveur à l'aide de certificat qu'il envoie au client à l'étape 2 du protocole handshake. Cette vérification comporte plusieurs phases :

1. Le client examine en premier la date de validité du certificat et vérifie qu'elle n'a pas été dépassée. Si cette vérification s'effectue correctement, on passe à l'étape 2.Sinon, l'utilisateur est averti du problème.
2. Si le client reconnaît l'autorité de certification ayant délivré le certificat du serveur, on passe à l'étape suivante. Tout client SSL dispose en effet d'une base de données contenant une liste d'AC qu'il reconnaît. Pour chacune d'entre elles, il dispose d'un certificat contenant la clé publique de l'autorité de certification.

3. Il vérifie la signature du certificat serveur à l'aide de la clé publique de l'AC qui l'a délivrée. Si cette signature est valide, on peut considérer que le certificat l'est aussi (il n'a pas été modifié et il est bien signé par l'autorité qui l'a délivré) et que, par conséquent, le serveur est authentifié. Il reste cependant une dernière étape.

4. Le client examine l'adresse de la machine à laquelle le certificat a été délivré et vérifie qu'elle correspond à celle du serveur auquel il est connecté. Cette vérification est importante, car elle permet d'éviter qu'un tiers malveillant vienne s'insérer entre le serveur et le client, ce qui lui permettrait d'obtenir toutes les informations transmises entre eux.

II.4. Le sous protocole SSLRecord

L'intégrité et la confidentialité des données transmises via le protocole SSL sont assurées par le sous-protocole SSLRecord. SSLRecord a pour rôle :

- L'encapsulation des données : il permet aux données d'être transmises et reconnues sous une forme homogène.

- D'assurer la confidentialité : il assure que le contenu du message ne peut pas être lu par un tiers. Les données sont chiffrées en utilisant les clés produites lors de la négociation.
- D'assurer l'intégrité : il permet de vérifier la validité des données transmises, grâce à la signature MAC. Cette signature est elle aussi générée à l'aide des clés produites lors de la négociation.

Le sous-protocole SSLRecord procède par deux processus à savoir :

1º Le processus d'encapsulation qui renferme les phases suivantes :

- Segmentation : Les données sont découpées en blocs de taille inférieure à 16.384 octets.
- Compression : Les données sont compressées en utilisant l'algorithme de hachage choisi lors de la négociation. Noter qu'à partir de SSv3, il n'y a plus de compression.
- Signature MAC (Message Authentication Code), désigne la signature d'un message par un algorithme à clé secrète (16 ou 20 octets) : Une signature de données est générée à l'aide de la clé MAC. Comme elle exploite une fonction de condensation, on parle en réalité de H-MAC (hashed-MAC).
- Chiffrement : Le paquet obtenu est chiffrée à l'aide de la fonction dénie lors de la négociation. Les algorithmes actuellement utilisés pour le chiffrement sont 3-DES168, IDEA1, RC4128,...
- Ajout de l'en-tête : L'en-tête SSL est ajoutée et le paquet est passé à la couche inférieure.

2º Le processus de réception des paquets.

A la réception des paquets, le destinataire effectue les opérations suivantes :

- Vérification de l'entête SSL.
- Dépouillement du paquet.
- Vérification du champ H-MAC (en appliquant la même fonction que celle utilisée pour chiffrer les données puis en comparant le résultat au H-MAC reçu).
- Décompression des données.
- Réassemblage des parties.

Si au cours de cette vérification, il y a un manquement, une alarme est générée grâce au sous-protocole d'alarme (SSLalert).

II.5.Implémentation de SSL/TLS

La majeure partie des implémentations de ssl/tls se trouve dans les navigateurs et serveurs web. Le serveur Apache, notamment, peut exploiter ssl grâce à une implémentation basée sur Openssl. Implémenté en langage C, OpenSSL est une boîte à outil de chiffrement comportant deux bibliothèques (une de cryptographie générale et une implémentant le protocole ssl), ainsi qu'une commande en ligne .Open ssl supporte ssl v2, ssl v3 et tls 1.0. OpenSSL contient des outils de gestion et des bibliothèques en relation avec la cryptographie. Ils sont utiles car ils apportent des fonctions de cryptographie.

Openssl contient C-rehash, openssl, et les bibliothèques libcrypto et libssl.

1) **C-rehash.**

C'est un script Perl qui cherche tous les fichiers d'un répertoire et ajoute des liens symboliques vers leurs valeurs hachées.

2) **Openssl.**

Le programme Openssl est un outil en ligne de commande pour utiliser les différentes fonctions de cryptographie de la bibliothèque d'Openssl à partir du shell.

3) **Lib crypto.**

La bibliothèque Libcrypto met en œuvre un large éventail d'algorithmes cryptographiques utilisés dans diverses normes Internet. Les services fournis par cette bibliothèque sont utilisés par le OpenSSL implementation du protocole SSL/TLS et S/MIME et ils ont également été utilisés pour mettre en place SSH (SSH : Secure Shell est à la fois un programme informatique et un protocole de communication sécurisé.), Open PGP et d'autres normes cryptographiques.

4) **Libssl** : La bibliothèque libssl d'OpenSSl implémente le protocole SSL (v2/v3) et TLS (v1). Elle apporte un API riche dont la documentation est disponible en lançant man3SSL.

II.6. Conclusion

Le protocole SSL est un protocole utilisé pour assurer la S-I c'est un protocole qui est transparent aux utilisateurs car il n'a pas besoin de protocole spécifique au niveau application pour qu'il soit exploité. Nous l'avons signalé, SSL se situe entre la couche application et la couche transport. Dans le modèle OSI ou TCP/IP, il est comme une nouvelle couche qui vient s'intercaler entre ces deux couches. SSL se subdivise en quatre sous protocoles dont :

- SSLHandshake utilisé pour négocier la session SSL
- SSLRecord qui décrit le format des données à échanger ;
- Change CipherSpec qui indique au SSLRecord la mise en place des algorithmes de chiffrement.
- SSLAlert qui signale les erreurs pouvant se produire entre le client et le serveur.

Le SSL assure l'authentification des utilisateurs grâce au certificat, l'intégrité des données par le chiffrement et la confidentialité et la non répudiation par signature électronique. Aujourd'hui, les utilisateurs de l'Internet orientent fortement leurs communications dans la messagerie électronique. Il y a lieu de savoir comment fonctionnent les systèmes de messagerie électronique. Le troisième chapitre sera consacré à cela.

CHAPITRE III. SSL DANS LES SYSTEMES DE MESSAGERIE.

III.0.Introduction

Nous l'avons déjà expliqué, le protocole SSL est un protocole utilisé pour sécuriser les transactions effectuées via Internet. Il se place entre la couche Application et la couche Transport et crée un canal sécurisé à travers lequel transitent les données échangées. Dans ce chapitre nous allons focaliser notre attention dans la sécurisation de la messagerie avec SSL et plus loin dans le chapitre nous allons définir ce qu'est la messagerie électronique et voir comment s'effectuent l'envoi et la réception des messages. Les protocoles de messagerie les plus utilisés sont SMTP (Simple Mail Transfer Protocol utilisé pour l'envoi des messages), POP3 (Post Office Protocol) et IMAP (Internet Message Access Protocol utilisés pour la réception des messages). Et on terminera par voir comment SSL intervient dans les dialogues SMTP, POP3 et IMAP afin de les sécuriser mais aussi nous verrons les faiblesses de SSL dans cette sécurisation.

III.1. La notion de messagerie électronique

III.1.1. Le courrier électronique : définition

Le courrier électronique est la ressource la plus utilisée de l'Internet. Il permet d'envoyer des messages et des documents à une personne déterminée, grâce à une adresse virtuelle que chaque utilisateur est supposé détenir. Le courrier électronique, ou courriel par contraction, est un service de transmission électronique de messages avec un réseau informatique (principalement l'Internet) dans la boîte aux lettres électronique d'un destinataire choisi par l'émetteur. **[17]**

III.1.2. Serveur de messagerie électronique

Un serveur de messagerie électronique est un logiciel qui a pour rôle de transférer les messages électroniques d'un serveur à un autre. Un utilisateur n'est jamais en contact direct avec ce serveur mais utilise soit un client de messagerie (Application permettant d'envoyer et de recevoir des messages avec vos différents contacts), soit un courriel web (Interface Web rendant possible l'émission, la consultation et la manipulation des courriels directement sur le Web depuis un navigateur) qui se charge de contacter le serveur pour envoyer ou recevoir les messages. Lorsque le serveur expéditeur est dissocié physiquement du serveur de réception le message passe par plusieurs serveurs appelés relais. Les échanges par courriel reposent sur l'utilisation combinée d'un ou plusieurs protocoles de messagerie.

III.1.3. Création et envoi d'un courrier électronique

Chaque personne ayant une connexion Internet peut envoyer ou recevoir des messages, avec ce réseau, en utilisant un système de messagerie de son choix. L'envoi d'un courrier ne garantit pas l'anonymat de l'expéditeur car le réceptionnaire d'un message connaîtra toujours l'adresse de son correspondant. Les adresses électroniques se présentent sous forme d'un login (nom d'utilisateur) suivi de l'arobase (signe "@") puis du nom du fournisseur d'accès ou du domaine (par exemple : placidega@yahoo.fr).
Entre un utilisateur et son serveur, l'envoi d'un courriel se déroule généralement via le protocole SMTP (Simple Mail Transfer Protocol). Puis c'est au serveur d'envoyer le message au serveur du destinataire, cette procédure s'appelle Mail Transfer Agent (MTA).
Les messages électroniques sont semblables aux lettres et se composent de deux parties principales : L'en tête qui contient le nom et l'adresse du destinataire (A :) et aussi de toutes les personnes placées en copie (Cc :), ainsi que l'objet du message. Certains programmes de messagerie affichent également le nom et l'adresse de l'expéditeur ainsi que la date d'envoi du message. L'autre partie est le corps qui contient le message. L'adresse que l'on indique doit être correcte tout comme lorsque l'on envoie des lettres manuscrites. Si l'adresse indiquée est erronée ou comporte une faute de frappe, le message est retourné à l'expéditeur pour cause d'adresse inconnue.

III.1.4. Réception d'un courriel.

La réception d'un courriel s'apparente grandement à celle d'un courrier postal. L'utilisateur en consultant sa boîte aux lettres virtuelle rapatrie ses messages sur sa propre machine.
La phase de rapatriement des messages s'effectue en deux temps : le serveur destinataire doit recevoir le message du serveur de l'expéditeur. Le serveur du destinataire doit être à mesure de gérer la boîte aux lettres et de signaler au serveur expéditeur toute erreur dans la délivrance du courriel. Cette procédure de réception est appelée Mail Delivery Agent (MDA). Lorsque le destinataire final désire accéder à ses messages, il lance une requête au serveur destinataire qui transmet les messages reçus, généralement via le protocole POP3 (Post Office Protocol) ou via le protocole IMAP (Internet Message Access Protocol).

III.2. Les protocoles de messagerie : SMTP, POP3 et IMAP

III.2.1. Introduction

Un protocole est un ensemble de règles et de procédures à respecter pour émettre et recevoir des données sur un réseau. Il en existe plusieurs selon ce que l'on attend de la communication. Certains protocoles sont par exemple spécialisés dans l'échange de fichiers (le FTP), d'autres à la gestion de l'état de la transmission et des erreurs, etc.
Sur Internet, les protocoles utilisés font partie d'une suite de protocoles appelée TCP/IP. Les protocoles de messagerie les plus couramment utilisés sont SMTP, POP3 et IMAP.

III.2.2.Le protocole SMTP

III.2.2.1.Présentation du protocole

SMTP tient pour Simple Mail Transfer Protocol. Il s'agit littéralement de « protocole simple de transfert de courriers »[15]. SMTP est un protocole de communication utilisé dans la première étape, c'est-à-dire pour transférer le courriel de la machine où il a été créé vers les serveurs de messagerie électronique. Le protocole SMTP fonctionne en mode connecté, encapsulé dans une trame TCP/IP. Rappelons que chaque message envoyé est composé de deux parties, l'en-tête et le corps du message. Avant que le message ne soit envoyé, le client et le serveur SMTP doivent s'entendre sur un certain nombre d'informations relatives à l'expéditeur et au destinataire du message. Une fois la vérification de ces informations (par le serveur SMTP) terminée, le corps du message est transféré vers le destinataire.
Avant d'expliquer étape par étape comment s'effectue ce transfert, il faut signaler qu'un même message transite, généralement, par plusieurs serveurs appelés relais SMTP jusqu'à ce qu'il arrive sur la machine où son destinataire pourra en prendre connaissance et entre chaque relais, une nouvelle session SMTP est ouverte. Les principales commandes de SMTP sont reprises dans le tableau ci-dessous.

Commande	Exemple	Description
EHLO	EHLO 193.56.47.125	Identification à l'aide de l'adresse IP ou du nom de domaine de l'ordinateur expéditeur
MAIL FROM:	MAIL FROM: expediteur@domaine.com	Identification de l'adresse de l'expéditeur
RCPT TO:	RCPT TO: destinataire@domaine.com	Identification de l'adresse du destinataire
DATA	DATA message	Corps du mail
QUIT	QUIT	Sortie du serveur SMTP
HELP	HELP	Liste des commandes SMTP supportées par le serveur

Tableau 1 : Les principales commandes SMTP [18]

Illustrons le fonctionnement du protocole en examinant ce qui se passe dès que John clique sur le bouton « envoyer » de son logiciel client. Nous allons décrire le dialogue entre ce logiciel client et le serveur SMTP utilisé par John. Dans ce dialogue et dans tous les autres dialogues que nous allons voir C tient pour client et S pour serveur :

Exemple 1. Transaction avec SMTP [19]

S: 220mail.superisp.fr ESMTP Sendmail 8.8.8/8.8.8 ; Thu,7 May2010 14 :35 :48 +0200(CEST)
C: EHLO john.superisp.fr
S: 250-mail.superisp.fr Hello john.superisp.fr[192.168.16.184], pleased to meet you.
250-EXPN
250-VRFY

250-8BITMIME
250-SIZE
250-DSN
250-ONEX
250-ETRN
250-XUSR
250-HELP
C: MAIL From: < john@superisp.fr > SIZE=64
S: 250 < john@superisp.fr >...Sender ok
C: RCPT TO: < jd@megaisp.fr >
S: 250 < jd@megaisp.fr >...recipient ok
C: DATA
S: 354 Enter mail, end with "." On line by itself.
250MAAO1459 Message accepted for delively
C: QUIT
S: 221 mail.superisp.fr closing connection

Les requêtes effectuées par le serveur sont précédées d'un nombre à trois chiffres, ce sont les réponses aux requêtes du client. Ce nombre est un code de réponse qui indique si la commande précédente s'est déroulée correctement (code commençant par un 2 ou un 3) ou non (4 ou 5). La première ligne est un message de bienvenu envoyé par le serveur. On reconnaît tout d'abord le nom (adresse DNS) de la machine sur laquelle se trouve le serveur SMTP considéré (mail.superisp.fr). Le protocole est indiqué juste après (ESMTP (où le E tient pour extension du protocole SMTP), dans notre exemple). Vient ensuite le nom du logiciel serveur (sendmail) suivi de sa version, de la date et de l'heure locale.
Une fois ce message reçu, le logiciel client va s'identifier en indiquant le nom de la machine de l'utilisateur (john.superisp.fr) via la commande EHLO, ce à quoi le serveur répond en précisant une liste de commandes qu'il reconnaît en plus de celles définies en standard dans le protocole SMTP. VRFY est une commande utilisée pour vérification d'un Nom d'Utilisateur et EXPN pour élargir une liste des destinataires.
8BITMIME indique que les données seront codées sur huit bits selon les normes de MIME.

Le logiciel client va alors commencer la transmission du message en envoyant successivement les adresses de l'expéditeur (via la commande MAIL From:), du destinataire (RCPT TO:) et enfin le contenu du message (non reproduit ici) dans son intégralité, c'est-à-dire en-tête et corps du message, grâce à la commande DATA.
La fin du message est marquée par l'envoi d'une ligne ne contenant qu'un point (.) en première colonne.

Le serveur confirme la transaction en indiquant un identifiant qu'il attribue de façon unique au message considéré (MAA01459), puis il commence la transmission du message vers le destinataire. Le client peut alors fermer la connexion via la commande QUIT. Le message va voyager de relais en relais jusqu'à son arrivée.

Dès qu'un message est reçu par le serveur SMTP et avant sa transmission au relais suivant, ce serveur ajoute un champ appelé « Received » au début de l'en-tête pour identifier le relais. Le serveur de relais, en recevant le message et après l'ajout du champ « Received » va renvoyer le message comme s'il en était l'expéditeur. Par exemple si un message passe par trois relais, on retrouvera à la réception du message trois champs « Received »

Exemple 2. Présentation du champ « Received » [19]

Received: from john.superisp.fr (john.superisp.fr [192.168.16.184])

by mail.superisp.fr (8.8.8/8.8.8) with ESMTP id MAA01459
for < jd@megaisp.fr >; Thu, 7 may 2010 14:35:48 + 0200 (CEST)
date: The, 7 may 2010 13:55:38 + 0200 (METDST)
From: John Smith <john@superisp.fr >
Subject: Une partie de chasse?
Message-Id: < 19990571234.QAB43249@superisp.fr >
Cher Jean,
.....
.....
A *bientôt*,
John Smith (john@superisp.fr)

Nous retrouvons ici le message de l'exemple 1, mais avec un champ « Received » occupant les trois premières lignes de l'en-tête.
Au début du dialogue, le logiciel client envoie au serveur SMTP/relais la commande « EHLO » suivie du nom de la machine sur laquelle il s'exécute. C'est ce nom que nous retrouvons ici après « from », dans le champ « Received » cette adresse DNS apparaît en deuxième fois entre parenthèse et elle est suivie de l'adresse IP qui lui est associée.
Bien que, dans notre exemple, ces deux adresses soient les mêmes, il est fréquent que cela ne soit pas le cas. La première est en effet librement spécifiée en argument de la commande EHLO, alors que la deuxième est déduite de l'adresse IP de la machine qui s'est réellement connectée au serveur. On distingue ensuite, après le « by », l'adresse DNS du serveur SMTP qui a ajouté ce champ « Received », suivie du nom du logiciel serveur utilisé ainsi que sa version. On retrouve les mêmes informations que celles qui avaient été données par le serveur dans son message de bienvenu lors de l'ouverture de la session SMTP. Ensuite, le protocole utilisé est identique (ESMTP), et on reconnaît l'identifiant qui avait été attribué au message lors de son acceptation, juste après que le client ait envoyé le point marquant la fin de l'émission du message. Enfin, l'adresse du destinataire ainsi que la date de réception du message considéré sont elles aussi indiquées.

Exemple 3. Message tel que parvenu à destination [19]

Received: from mail.superisp.fr (mail.superisp.fr [192.168.1611])
By mail .megaisp.fr (8.8.9/8.8.9) with ESMTP id PAA01045
For < jd@megaisp.fr >; Thu, 7may2010 14:37:02 + 0200 (CEST)
Received: from John.superisp.fr (john.superisp.fr [192.168.16.184])
by mail.superisp.fr (8.8.8/8.8.8) with ESMTP id MAA01459
for < jd@megaisp.fr >; The, 7 May 2010 14:35:48 + 0200 (CEST)
Date: The 7 May 2010 13:55:38 + 0200 (METDST)
From: John Smith < john@superisp.fr >
To: Jean Dupont < jd@megaisp.fr >
Subject: Une partie de chasse?

Message-Id : < 199905071234.QAB43249@superisp.fr >
Cher Jean,
.....
.....
.....
A bientôt,
John Smith (john@superisp.fr)

Le premier « Received » au début de l'en-tête, a donc été ajouté par le dernier serveur SMTP impliqué dans la distribution du message considéré. Sa structure étant semblable à celle présentée plus haut, nous n'allons pas revenir sur sa description. Le message est donc arrivé à destination, sur la machine mail.megaisp.fr et Jean Dupont va pouvoir en prendre connaissance. En fait, deux cas peuvent se présenter : si Jean dispose d'un accès direct, c'est-à-dire s'il est le propriétaire du serveur de réception, il peut y consulter directement son courrier. Sinon, notre internaute va devoir récupérer son message à distance. C'est à ce moment que va intervenir le protocole POP3 qui fait l'objet de la section III.2.3.

III.2.2.2. Authentification SMTP

L'authentification SMTP est une extension du protocole SMTP. C'est un protocole de transfert des courriels sur Internet qui inclut une étape d'authentification au cours de laquelle le client présente une adresse complète au serveur. Le client se connecte explicitement à un serveur (ou relais) qu'il reconnaît pour l'envoi de courriers mais le serveur doit aussi identifier le client. Les serveurs qui supportent le protocole SMTP-AUTH (qui peuvent dialoguer en usant d'une authentification préalable) peuvent être paramétrés pour n'accepter que des clients capables de s'authentifier (clients possédant cette extension).

L'extension SMTP-AUTH permet aussi à un serveur (ou relais) de courriers d'indiquer à un autre serveur que l'expéditeur a été authentifié lors du relayage des courriels. En général, cela nécessite au serveur cible de croire le premier serveur (autrement dit que ce serveur soit déclaré comme serveur de confiance), raison pour laquelle ce protocole est peu utilisé sur Internet. Le destinataire d'un courrier ne peut pas savoir si l'expéditeur a été identifié car cela est transparent pour ce premier. Bien que SMTP-AUTH soit globalement une solution plus sécurisée que le SMTP, il peut cependant comporter des faiblesses. En effet, si les utilisateurs authentifiés sont autorisés à émettre des courriers depuis certaines adresses IP, rien n'indique qu'un utilisateur frauduleux ne pourra pas violer un compte utilisateur pour utiliser le serveur de courriers (par exemple en récupérant un mot de passe).

III.2.3. Le protocole POP

POP tient pour Post Office Protocol ; il s'agit littéralement du « protocole du bureau de poste ». POP est un protocole qui permet à l'utilisateur de récupérer son courrier électronique sur un serveur distant (le serveur POP). Ce protocole a été réalisé en plusieurs versions respectivement POP1, POP2 et POP3 qui est actuellement utilisé comme le standard. POP est souvent utilisé en complément au protocole SMTP de façon à permettre la récupération à distance des messages stockés sur un serveur de messagerie.
Tout comme dans le cas du protocole SMTP, le protocole POP fonctionne grâce à des commandes textuelles envoyées par le client composées d'un mot clé, éventuellement accompagné d'un ou plusieurs arguments.

Les principales commandes POP version3 sont :

Commande	Description
USER identifiant	Cette commande permet à l'utilisateur de s'authentifier. Elle doit être suivie du nom de l'utilisateur. La commande USER doit précéder la commande PASS
PASS : mot de passe	La commande PASS, permet d'indiquer le mot de passe de l'utilisateur dont le nom a été spécifié lors d'une commande USER préalable.
STAT	Information sur les messages contenus sur le serveur.
DELE	Numéro du message à supprimer
LIST [msg] et RETR	LIST indique le numéro du message à afficher et RETR le numéro du message à récupérer.
NOOP	Permet de garder les connexions ouvertes en cas d'inactivité.
TOP <message ID> <n>	Commande affichant n lignes du message dont le numéro est donné en argument. En cas d'une réponse positive (à la requête du client) du serveur, celui-ci renvoie les en-têtes du message, puis une ligne vierge et enfin les premières lignes du message.
UIDL [msg]	Demande au serveur de renvoyer une ligne contenant des informations sur le message éventuellement donné en argument. Cette ligne contient une chaîne de caractères, appelé « listing d'identificateur unique » permettant d'identifier de façon unique le message sur le serveur, indépendamment de la session.
QUIT	La commande QUIT demande la sortie du serveur POP3. Elle entraîne la suppression de tous les messages marqués comme effacés et renvoie l'état de cette action.

Tableau 2 : Les commandes du POP3 [18]

Revenons sur notre exemple de la section précédente du message envoyé avec SMTP et qui se trouve sur le serveur de réception, de la section précédente :
Jean lance son logiciel de messagerie et lance une requête au serveur pour récupérer son courriel. Le logiciel (client) se connecte alors au serveur < mail.megaisp.fr> avec le protocole POP. L'exemple suivant présente le déroulement du dialogue entre le client et le serveur dans le cas du POP3.

Exemple 4. Récupération d'un message avec P0P3 [19]

S: + OK QPOP (version 2.5) at mail.megaisp.fr starting
C: USER jd
S: + OK Password required for jd.
C: PASS secret.

S: + OK jd has 1 message (742 octets).
C: UIDL
S: + OK will command accepted.
S: 1 C26dc4f60 e958de47 ab8042618bb5

S: .
C: RETR 1
S: + OK 742 octets
S: < le serveur POP3 envoie le message 1 >
S: .
C: DELE 1
S: + OK Message 1 has been deleted
C: QUIT
S: + OK Pop server at mail.megaisp.fr signing off.

Contrairement au protocole SMTP, les codes de réponses ne sont pas des nombres mais des chaînes de caractères qui peuvent prendre deux valeurs, + OK pour une réponse positive, - ERR pour une réponse négative (il n'y en a pas dans notre exemple).
La première ligne de ce dialogue est un message de bienvenu envoyé par le serveur. Ce dernier indique la version du logiciel utilisé (ici QPOP) ainsi que l'adresse DNS de la machine.
Le serveur de messagerie va ensuite indiquer le nom d'utilisateur associé à Jean, à savoir jd, via la commande USER. Si ce nom est connu du serveur, il envoie une réponse positive, donc débutant par « + OK ». Le logiciel client spécifie alors le mot de passe associé (commande PASS).
Le serveur répond positivement en indiquant le nombre de messages en attente de récupération ainsi que la taille totale occupée par ceux-ci. Le logiciel de messagerie envoie ensuite la commande UIDL qui demande au serveur de lui envoyer la liste des messages contenus dans la boite aux lettres de l'utilisateur avec, pour chacun d'entre eux, un identifiant unique généré par le serveur.
Ce message est ensuite récupéré grâce à la commande RETR suivie du numéro du message. Le serveur, après avoir indiqué la taille du message en question, l'envoie en entier (en-tête et corps) au client. Ceci fait, le serveur de messagerie l'efface du serveur avec la commande DELE.
Notons cependant que, même si le serveur indique l'avoir effacé, l'ordre de destruction ne sera exécuté physiquement qu'à la fin de la session (le message est seulement marqué comme devant être effacé ultérieurement). Ensuite, le client envoie la commande QUIT qui met fin au dialogue. Cette fois-ci, la communication est terminée et le serveur efface réellement les messages que le client lui a demandé de détruire. Ainsi, Jean va pouvoir enfin prendre connaissance du courrier envoyé par John. Voici l'apparence définitive du message reçu :

Exemple 5. Apparence définitive du message reçu [19]

Received: from mail.superisp.fr (mail.superisp.fr [192.168.16.11])
by mail.megaisp.fr (8.8.9/8.8.9) with ESMTP id PAA 01045
for < jd@megaisp.fr >; The,7 May 2010 14:37:02 +0200 (CEST)
Received: from john.superisp.fr (john.superisp.fr [192.168.16.184])
by mail.superisp.fr (8.8.8/8.8.8) with ESMTP id MAA 01459
for < jd@megaisp.fr >; The,7 May 2010 14:35:48 +0200 (CEST)
Date: The;7 May 2010 13:55:38 +0200 (METDST)
From: John Smith < john@superisp.fr >

To: Jean Dupont < jd@megaisp.fr >
Subject: Une partie de chasse?
Message-Id: < 199905071234.QAB43249@superisp.fr >

X-UIDL: c26dc4f26f60e958de47ab8042618bb5

Cher Jean,
.....
.....
.....
A bientôt,
John Smith (john@superisp.fr)

Le message reproduit ici est en tout point identique au message de l'exemple III.3, à l'exception de la présence d'un champ supplémentaire (X-UIDL:), en dernière ligne de l'entête. On reconnaît aisément l'identifiant unique qui avait été envoyé par le serveur en réponse à la commande UIDL du client (ici : c26dc4f26f60e958de47ab8042618bb5).
Le seul moyen que possède le protocole POP3 pour l'identification de l'utilisateur n'est que le mot de passe et nom d'utilisateur. Il n'est par contre pas sécurisé car les mots de passe, au même titre que les corps du message circulent en clair sur le réseau.

III. 2.4. Le protocole IMAP

« IMAP tient pour Internet Message Access Protocol »[15]. Il s'agit d'un protocole de messagerie permettant l'échange de messages électroniques entre utilisateurs sur Internet. Le protocole IMAP est né en 1986 à l'université de Stanford et il a subi de multiples révisions. Actuellement on est à sa quatrième version, IMAP4 rev1 décrite dans le « Request for comments 2060 » **[16].**
Tout comme POP3, IMAP est un protocole de réception de message électronique. Il se pose donc comme une alternative à POP3. On peut se poser la question sur la pertinence de son existence au moment où il joue le même rôle, la réception des messages électroniques, que le POP3. La raison est que le protocole IMAP4 offre plus de services venant s'ajouter à ceux offerts par POP3. Un des principales nouveautés d'IMAP4 est la possibilité de pouvoir lire uniquement les objets des messages (sans le corps du message). Ainsi, on peut par exemple effacer des messages sans les avoir lus. L'autre avantage est qu'il permet le stockage des mails sur le serveur de réception. En effet, avec IMAP4 les messages restent stockés dans des dossiers sur le serveur. Ceci permet de proposer de nombreuses fonctionnalités telles que : Créer des dossiers sur le serveur, effacer, déplacer des mails sans les lire éventuellement avec des règles de tri automatique, lire des mails en les laissant sur le serveur de réception, recopier sur le serveur de réception des mails qui sont en local. IMAP4 offre aussi la possibilité à plusieurs personnes de consulter une même boîte aux lettres en même temps (cas des forums). Afin de bien comprendre tout cela, voyons comment fonctionne le protocole IMAP4.

Tout comme dans le cas de SMTP et POP3, IMAP fonctionne grâce à des commandes textuelles, la seule différence étant que les commandes IMAP4 doivent être indexées. C'est-à-dire qu'elles sont précédées d'un index unique. Ceci permet dans certaines conditions d'envoyer au serveur une nouvelle commande alors qu'on n'a pas encore reçu la réponse de la précédente. Il y a une interaction client/serveur consistant en une suite de commandes/réponses entre le client et le serveur. Toutes les données transmises entre le client et le serveur se présentent sous forme de lignes.

Les principales commandes d'IMAP sont :

Commande	Description
Capability	demande la liste des possibilités que le serveur supporte.
authenticate	indique au serveur un mécanisme d'authentification. Si le serveur supporte le mécanisme d'authentification demandé, il exécute un échange protocolaire d'authentification afin d'identifier le client. Il est aussi possible d'utiliser la commande `login`, plus rudimentaire car elle ne supporte que le nom d'utilisateur et son mot de passe en texte clair.
Namespace	Pour aller vite, cette commande retourne ce qu'il faut pour que l'utilisateur puisse connaître les dossiers auxquels il peut accéder et souscrire, ce qui peut inclure les dossiers partagés, prévus dans le protocole IMAP.
Lsub	permet d'obtenir à partir d'un point de référence la liste de dossiers auxquels le client a souscrit (qu'il désire voir dans son client).
List	permet d'obtenir à partir d'un point de référence la liste de dossiers existants
Select	sélectionne une boite aux lettres, ainsi les messages dans la boite aux lettres sont accessibles. Cette commande renvoie le nombre total de messages (EXISTS), et le nombre de nouveaux messages (RECENT)
getquotaroot	Comme son nom le laisse penser, cette commande renvoie l'état d'occupation de l'espace alloué, lorsque les quotas sont activés sur le serveur IMAP
Uid	s'utilise avec les commandes COPY, FETCH, STORE ou encore SEARCH. Son utilité est de renvoyer un index unique plutôt qu'un numéro de séquence, comme le feraient les commandes COPY, FETCH, STORE et SEARCH
Fetch	cette commande dispose d'une syntaxe assez complexe. Elle permet d'obtenir de nombreuses informations sur un message ou un lot de messages.
Idle	lorsque le serveur supporte cette commande, il permet au client d'être informé en temps réel de l'arrivée de nouveaux messages
Expunge et close	Expunge a pour mission de détruire physiquement tous les messages marqués \deleted dans le dossier courant. Close assez similaire à EXPUNGE, cette commande détruit de façon définitive tous les messages qui ont le drapeau `\deleted`. Elle ne renvoie pas de compte rendu, comme le fait EXPUNGE.
Logout	Logout met fin à la session IMAP.

Tableau 3 : Les commandes d'IMAP [20]

En plus de ces principales commandes s'ajoutent d'autres commandes appelées « flags » et ces flags sont les suivants :

\Seen	Le message a été lu
\Answered	Il a été répondu au message
\Flagged	Le message est marqué pour lui donner une attention particulière
\Deleted	Le message est supprimé pour que plus tard un EXPUNGE ou un CLOSE puisse le supprimer physiquement sur le serveur
\Draft	Le message n'a pas été entièrement composé. Il est marqué en tant que brouillon
\Recent	Le message est arrivé dans cette boite aux lettres après la dernière consultation de celle-ci. Les sessions ultérieures ne verront pas l'état \Recent pour ce message. Ce drapeau ne peut être modifié par le client. S'il n'est pas possible de déterminer si oui ou non, cette session est la première session à être notifiée du message, alors ce message DEVRA (SHOULD) être considéré comme récent

Tableau 4 : Les flags utilisés par IMAP [20].

La récupération des messages avec IMAP4 est décrite dans l'annexe.

III.3. Synthèse

Le schéma suivant montre les différentes étapes empruntées par un message électronique dès sa création jusqu'à son destinataire avec mention des différents cas possibles que nous avons décrits.

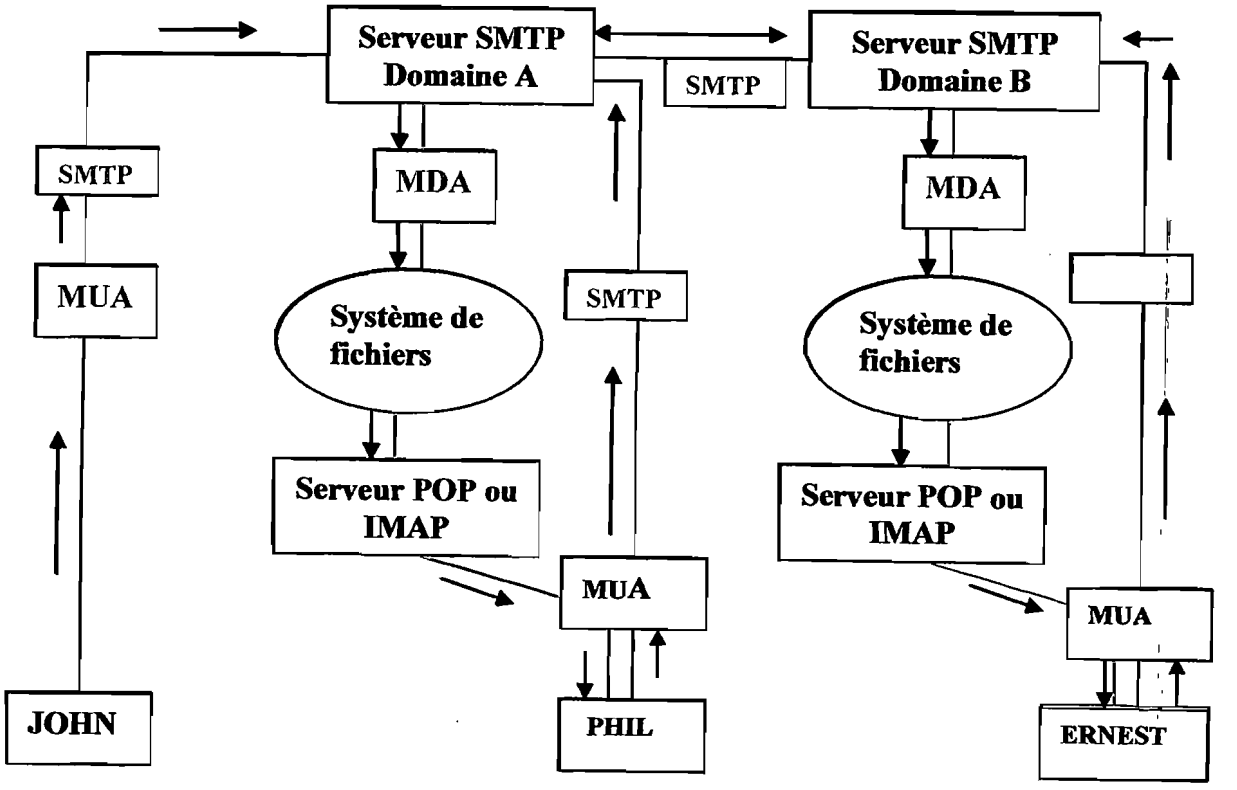


Figure III.1 : Parcours d'un message électronique [21]

Nous remarquons que les messages envoyés par JOHN à PHIL passent par un seul serveur (Domaine A) tandis que les messages envoyés par JOHN à ERNEST transitent par deux serveurs (Domaines A et B) au même titre que ceux entre PHIL et ERNEST. MUA (Mail User Agent) est un logiciel de relève/d'envoi du courrier électronique.

III.4. Sécurisation des protocoles de messagerie par SSL

III.4.0. Messagerie électronique sécurisée

Les protocoles SMTP, POP3 et IMAP permettent l'échange des messages en claire sur le réseau. Cela signifie que n'importe qui peut les intercepter et en prendre connaissance. Pire, une personne malveillante peut modifier un message qui ne lui est pas destiné, sans que le destinataire n'ait la possibilité de s'en rendre compte. Par exemple lorsque, nous recevons un message comportant Butoyi@yahoo.fr comme adresse de l'expéditeur, rien ne nous prouve que ce soit réellement l'expéditeur attendu qui l'a envoyé. Et même si c'est bien lui, rien ne l'empêchera ultérieurement de nier l'avoir fait. De tels cas de figures arrivent si la messagerie n'est pas sécurisée. Pour résoudre ces différents problèmes évoqués, le protocole SSL met à la disposition des systèmes de messagerie des solutions basées sur les méthodes cryptographiques.

Par message sécurisé, nous entendons tout message qui répond aux conditions suivantes :

- Confidentialité : Seul(s) le(s) destinataire(s) d'un message peut (peuvent) en prendre connaissance.
- Intégrité : Un message ne peut être modifié à l'insu du destinataire.
- Authentification : L'identité de l'expéditeur peut être vérifiée.
- Non-répudiation : l'expéditeur ne peut pas nier avoir émis un message une fois celui-ci reçu par son destinataire ou le destinataire de nier l'avoir reçu.

Le protocole SSL fait recours au chiffrement pour assurer la confidentialité d'un message et à la signature digitale pour assurer l'intégrité, l'authentification et la non répudiation. Comme il a été déjà explicité, le protocole SSL s'appuie sur l'utilisation d'un certificat pour assurer l'authenticité d'un message.

III.4.1. Signature et chiffrement d'un message

D'une façon générale, dès que l'on dispose d'un certificat, il est nécessaire de le diffuser à tous les correspondants réguliers de façon qu'ils puissent vérifier l'authenticité et l'intégrité des messages émis et en cas de besoin ces messages puissent être chiffrés. Pour chiffrer les messages, le protocole SSL possède un certain nombre d'algorithmes de chiffrement tels que le DES (56 bits), le 3 DES (168 bits) et, RC pour le chiffrement symétrique et RSA, DSA, ... pour le chiffrement asymétrique. Concernant la signature, des fonctions de hachage (SHA ou MD5) associées aux algorithmes de signature tels que RSA et DSA sont utilisées pour signer les messages.

Chacun de nous peut se demander comment se fait le chiffrement et la signature lorsque le message doit passer par plusieurs relais avant d'arriver au serveur de réception. En effet, lors de la transmission d'un message d'un relais à un autre, la négociation d'une session sécurisée se fait entre les deux relais. Ce qui veut dire que le relais d'envoi chiffre le message

que le relais de réception déchiffre. La procédure se répète jusqu'au dernier relais qui négocie la dernière session sécurisée avec le serveur de réception.

III.4.2. Implémentation de SSL dans les protocoles de messagerie

La mise en place d'une session sécurisée par SSL dans la messagerie électronique se décide tout au début du dialogue (entre le client et le serveur). Si la négociation de SSL réussie, le protocole SSL sera vu comme un tunnel où transitent les données échangées. Ce procédé est connu sous le nom de tunnelling. La sécurité du tunnelling réside dans le fait que l'on négocie le chiffrement avant tout échange d'information. Le chiffrement SSL s'obtient grâce à une requête lancée par le client au serveur avec la commande STARTTLS permettant de passer en mode chiffré. Cette commande une fois lancée indique au serveur que le client veut que ses messages soient chiffrés. Si la réponse du serveur est positive, la négociation TLS peut commencer. Cette négociation procède de la même façon que celle explicitée au niveau du sous-protocole SSLHandshake. Si la négociation réussie l'envoie/réception sécurisée via TLS peut commencer sinon l'envoi/réception échoue. Nous tenons à dire que pour le protocole IMAP, en sa quatrième version, le chiffrement est obligatoire. Cela veut dire que si la négociation TLS échoue, on a qu'à recourir aux anciennes versions pour initier des transactions non sécurisées. Mais pour les autres protocoles, POP3 et SMTP, les transactions non sécurisées sont possibles si la négociation SSL/TLS échoue.
Dans la section qui suit, nous revenons à nos dialogues (vus précédemment) mais avec l'introduction de SSL pour sécuriser les transactions s'effectuant entre le client et le serveur de messagerie.

III.4.2.1. Le protocole SMTP sécurisé par SSL

La négociation SSL dans le protocole de messagerie SMTP se passe tout au début du dialogue entre le logiciel client et le serveur SMTP. Nous présentons ci-dessous un dialogue SMTP entre un client et un serveur sécurisé par SSL.

S: < waits for connection >
C: < opens connection >
S: 220mail.superisp.fr ESMTP service ready
C: STARTTLS
S: 220 Go ahead
C: <Starts TLS negotiation >
C α S: <negotiate a TLS session >
C α S: <check results of negotiation >
C: EHLO mail. superisp.fr
S: 220mail.superisp.fr ESMTP Sendmail 8.8.8/8.8.8; Thu, 7 May2010 14:35:48 +0200(CEST)
C: EHLO john.superisp.fr
S: 250-mail.superisp.fr Hello john.superisp.fr [192.168.16.184], pleased to meet you.
250-EXP
250-VERB
250-8BITMIME
250-SIZE
250-DSN

250-ONEX
250-ETRN
250-XUSR
250-HELP
C: MAIL From: < john@superisp.fr > SIZE=64
S: 250 < john@superisp.fr >... Sender ok
C: RCPT TO: < jd@megaisp.fr >
S: 250 < jd@megaisp.fr >...recipient ok
C: DATA
S: 354 Enter mail, end with "." On line by itself
250MAAO1459 Message accepted for delively
C: QUIT
S: 221 mail.superisp.fr closing connection

Les trois premières lignes de notre dialogue nous montre que le client se connecte à son serveur. La réponse du serveur est favorable (service ready). La partie qui suit (en italique) montre le début de la négociation SSL en sa version 3.1 (TLS). Le client envoie la commande STARTTLS signifiant qu'il veut que ses données soient chiffrées. Le serveur SMTP lui répond favorablement (go ahead) et il commence la négociation SSL. Cette négociation est celle que nous avons expliquée pour le protocole SSLHandshake.
Si la négociation réussie, les données échangées seront chiffrées. Mais, si la négociation échoue, les échanges peuvent continuer (en mode clair) suivant la volonté du client.

III.4.2.2. Le protocole POP sécurisé par SSL

Tout comme pour le SMTP, le protocole POP dispose d'une commande STARTTLS (STLS est son nom pour POP3) permettant d'identifier le client en remplaçant le couple Nom d'utilisateur/Mot de passe par un certificat mais aussi de passer en mode chiffré. Illustrons notre propos par un exemple pratique en revenant au dialogue de l'exemple III.4 mais avec insertion de SSL :

S: +OK QPOP (version 2.5) at mail.megaisp.fr starting
C: STLS
S: +OK STLS required
S α C: < Negotiation for TLS session >
C: Certificate jd
S: +OK jd has 1 message (742 OCTETS).
C: UIDL
S: +OK UIDL command accepted
S: 1 C26dc4f60 e958de47 ab8042618bb5
S: .
C: RETR 1
S: + OK 742 octets
S: < le serveur POP3 envoie le message 1 >
S: .
C: DELE 1
S: + OK Message 1 has been deleted
C: QUIT
S: + OK Pop server at mail.megaisp.fr signing off.

Le client et le serveur commencent leur dialogue par un message de bienvenue envoyé par le serveur au client. Le client envoie en réponse au serveur une commande STLS lui indiquant qu'il veut un dialogue sécurisé par TLS. Le serveur répond favorablement (+OK TLS required). C'est à partir de ce moment que la négociation de la session TLS commence. Si la négociation s'est bien déroulée, le client demande le certificat au serveur et une fois accepté, le processus de récupération du message peut commencer et il est sécurisé par TLS.

III.4.2.3. Les limites et attaques de SSL dans la messagerie électronique

a) Les attaques de SSL dans la messagerie électronique

Dans cette section, nous allons essayer de décrire les différentes failles dans le protocole SSL mais la liste est non exhaustive. Différentes versions de SSL ont été définies pour contrecarrer certaines de ces attaques comme nous allons le voir dans cette section.

- SSL est théoriquement vulnérable aux attaques par force brute en cas d'utilisation de clés dont le nombre de bits est faible. Généralement, la clé privée d'un client est cryptée et conservée dans un fichier dont l'accès est protégé. Si l'on parvient à récupérer ce fichier, il suffit de lancer un logiciel destiné aux attaques par force brute. Si l'accès au fichier est protégé par un mot de passe, cette attaque peut alors être menée sur le mot de passe puis sur la clé qui se trouve dans le fichier. L'attaque par force brute se base sur le fait que n'importe quel mot de passe (clé) est crackable. Ce n'est qu'une histoire de temps dans la mesure où la puissance des machines double tous les deux ans (théorie de moore). De plus, les crackers n'hésitent pas à fabriquer des cartes électroniques de cracking, ce qui améliore en conséquence la rapidité de la machine et donc les chances de trouver un mot de passe valide. Il est donc conseillé d'utiliser des clés de 128 bits. En effet, si l'espace de la clé correspond à l'ensemble des mots de k bits, le nombre moyen d'appels à la fonction de déchiffrement requis dans cette attaque est égal à 2^{k-1}. Une telle attaque devient donc hors de portée dès que l'espace des clés est suffisamment grand. Le temps de calculs nécessaire à cette attaque est évidement exponentiel en la taille de la clé secrète. Il est 2^{64} fois plus dur de casser un système possédant une clé de 128 bits que de casser un système avec une clé de 64 bits.

- SSL V2 est vulnérable à une attaque dite du Roll back où l'opposant cherche à modifier le choix des algorithmes d'échange de clés de façon à ce que les deux entités n'utilisent pas les mêmes suites cryptographiques. Ceci est facilité par le fait que dans les messages client hello et serveur hello, les deux parties en communication s'envoient en clair les suites cryptographiques qu'ils utilisent. SSL V3 parvient à contrecarrer cette attaque étant donné que dans cette version au niveau du sous protocole changecipherspec (ccs), le client et le serveur s'envoient le message finished qui contient le MAC (en utilisant le master-secret comme clé) de tous les messages échangés dans le protocole Handshake. S'il advient que leurs suites cryptographiques soient changés, ils vont s'en rendre compte et décider s'il faut abandonner la session ou pas.

- L'homme au milieu: cette attaque peut avoir lieu lorsque trois personnes sont dans une session de communication: le client, le serveur et le malfrat. Ce dernier se situe sur le réseau entre le client et le serveur et intercepte le trafic provenant à la fois du client et du serveur. L'homme du milieu opère en faisant semblant d'être le vrai serveur envers le client et vice versa. Avec SSL, cette attaque est impossible grâce à l'utilisation du certificat serveur signé par une autorité. La clé publique du serveur, son nom ainsi que le nom du donneur du

certificat sont contenus dans le certificat du serveur. Le client vérifie le certificat en regardant la signature et en vérifiant si le nom du donneur du certificat est quelqu'un que le client reconnaît. De plus, le serveur doit chiffrer quelque chose avec la clé privée qui s'associe avec la clé publique mentionnée dans le certificat. Seul le serveur qui possède à la fois le certificat et la clé privée peut répondre convenablement au message.
Si l'homme du milieu fournit un faux certificat, la vérification de la signature le détectera. Si la signature est une signature légitime pour le malfrat mais pas pour le serveur, la signature passera mais alors c'est la vérification du nom du serveur qui détectera l'effraction. Finalement, si le malfrat fournit un certificat légitime pour le serveur en question, la signature et le nom passeront. Cependant, le malfrat ne possédant pas la clé privée du serveur, ne pourra pas encoder convenablement sa réponse au challenge et sera détecté lors de cette vérification. Dans le cas très peu probable où le malfrat a deviné la clé privée, il pourra retrouver le pré secret maître que le client envoie chiffré avec cette clé. L'homme au milieu sera détecté si au niveau de la dernière phase dans l'identification du serveur, le client se rendra compte que l'adresse de la machine à laquelle le certificat a été délivré ne correspond pas à celle du serveur auquel il est connecté.
- Attaque sur le CBC (Cipher Bloc Chaining): cette attaque est possible si l'opposant parvient à avoir une collision de deux blocs chiffrés y_i et y_j correspondants respectivement aux blocs clairs x_i et x_j. Nous avons:

$$y_{i-1} \times y_{j-1} = x_i \times x_j$$

Par analyse du xor des deux chiffrés, on enlève l'aléatoire introduit par la clé et on peut analyser statistiquement cette somme binaire pour en extraire les deux messages clairs x_i et x_j. D'après le paradoxe des anniversaires, la probabilité de produire une telle collision est de

$$P = 1 - e^{\frac{-1}{2}\left(N^2 \times W^{-b}\right)}$$

Où N est le nombre total de blocs et w le nombre total de mots que l'on peut former et b le nombre de bits sur lesquels on code. Pour plus de détails voir annexe C.

Si on analyse la valeur de P, on se rend compte qu'elle s'approche plus de zéro quand N croît. La probabilité de réussite d'une telle attaque est très faible pour N grand. Aussi, il faut nous rendre compte que cette attaque n'est possible que si la clé de chiffrement reste inchangée pour toutes les sessions; ce qui n'est pas le cas pour SSL car la clé de chiffrement est changée pour chaque connexion au cours du protocole handshake.

Exemple: Avec b=8, on peut avoir 2^8 soit 256 mots et par la théorie du paradoxe des anniversaires, il faut un total de 2^{35} bytes de la clé pour avoir une probabilité de réussite de 39%.

- Attaque replay: Dans cette attaque une personne enregistre une session de communication entre un client et un serveur. Plus tard, cette personne se connecte à ce même serveur et rejoue les messages du client qu'il vient d'enregistrer. SSL élimine cette attaque en utilisant une "nonce" (connection-id) qui est "unique" pour chaque connexion. En théorie, le malfrat ne peut prédire à l'avance la nonce car elle est basée sur une série d'événements sur lesquels il n'a aucun contrôle. Le malfrat ne peut donc pas répondre aux demandes du serveur.
Une personne avec un bon enregistreur peut enregistrer beaucoup de session entre un client et un serveur et tenter de choisir la bonne session en se basant sur la nonce que le serveur envoie initialement dans son message SERVER-HELLO. Cependant, les nonces SSL ont une longueur de 128 bits. Le malfrat aurait donc besoin d'environ 264 nonces pour avoir 50% de chance de choisir la bonne session.

b) Les limites des protocoles SSL/TLS

Dans le monde informatique, aucun protocole utilisé pour sécuriser les transactions n'est infaillible. Tout protocole, SSL y compris, présente un niveau seuil dans la protection des données transitant sur l'Internet qu'il est censé protéger. Cela signifie que parmi les algorithmes cryptographiques utilisés pour sécuriser les données aucun n'est incassable. La sûreté d'un protocole dépend du temps nécessaire pour casser les algorithmes qu'il utilise. SSL présente lui aussi des limites dans la protection des transactions effectuées via la messagerie électronique.

- Le protocole SSL/TLS sert à sécuriser les courriels lors de leur transmission entre un client et un serveur et à obtenir une authentification mutuelle. Une session SSL/TLS correctement établie va donc protéger les échanges entre les parties, mais ne sera en aucun cas une garantie de sécurité pour les systèmes client ou serveur. Ainsi, si un pirate installe par exemple un enregistreur de frappe (ou keylogger) sur le poste client, il sera à mesure de récupérer les mots de passe ou toute information confidentielle, même si celle-ci a été émise lors d'une session SSL/TLS. De même, un serveur qui utilise des sessions SSL/TLS pour récupérer ses courriels peut être compromis et les données recueillies pourront être volées.
- Les transactions électroniques sont sécurisées au cours de leur transmission entre le serveur et le client. Cela veut dire que si un serveur mal intentionné parvient à se passer pour un relais il peut détourner ou modifier les données transmises. Il va utiliser l'attaque par l'IP Spoofing. (L'adresse IP d'un ordinateur est l'adresse qui est utilisée pour reconnaître un ordinateur sur internet.) Un des principaux problèmes est qu'en utilisant le routage source d'IP, l'ordinateur du hacker peut se faire passer pour un ordinateur connu. Le routage source d'IP est une option qui peut être utilisée pour spécifier une route directe à une destination et renvoyer le chemin de retour à l'expéditeur. La route peut inclure l'utilisation d'autres routeurs ou de serveurs qui n'auraient normalement pas été utilisés pour faire suivre les paquets à la destination finale. Voici un exemple qui montre comment ceci peut être utilisé de façon à ce que l'ordinateur de l'intrus apparaisse comme étant l'ordinateur certifié par le serveur :

  1. l'agresseur change l'adresse IP de son ordinateur pour faire croire qu'il est un client certifié par le serveur,
  2. il va ensuite forcer les communications à transiter par son ordinateur en se faisant passer pour un relais indispensable. Il lui est alors aisé de modifier les données échangées entre le serveur et le client,
  3. l'agresseur envoie une requête client au serveur en utilisant la route source et vice versa,
  4. le serveur accepte la requête du client comme si elle provenait directement du client certifié et retourne une réponse au client et vice versa.

- Malgré les mécanismes présentés ci haut pour empêcher l'attaque par l'homme du milieu, très récemment, le 31/7/2010, une méthode permettant d'intercepter les messages électroniques sécurisés par SSL en utilisant un certificat appelé '*null-termination*' (inventé par un chercheur en sécurité appelé 'Moxie Marlinspike') vient d'être démontrée. Ce certificat sert à remplacer le vrai certificat. Pour que cette attaque fonctionne, l'attaquant doit déjà réussir à se passer pour un relais. Une fois que c'est fait, il détecte le trafic SSL et se passe pour un client/serveur pour intercepter les communications entre le client et le serveur en présentant son certificat '*null-termination*'. L'attaquant crée des certificats pour son propre domaine Internet qui

inclue des caractères 'nuls' (souvent représentés par \0), certains programmes interprètent mal le certificat. Ainsi, un certificat issu par exemple de www.paypal.com\0.thoughtcrime.org pourrait être lu comme appartenant à www.paypal.com. Selon les chercheurs, ce problème est largement répandu, et affecte Internet Explorer, les logiciels VPN (virtual private network), les logiciels de messagerie instantanée et les clients e-mails, etc. **[4]**

Il est donc primordial d'utiliser les protocoles SSL/TLS en prenant certaines précautions, et ne pas leur prêter une trop grande garantie de sécurité sur l'ensemble de la chaîne d'information car nous venons de voir que ce protocole présente certaines limites.

III.5. Le système de messagerie de l'Université du Burundi

III.5.0. Introduction

Si une institution donnée se compose de plusieurs bureaux ou succursales, il y a nécessité de les relier par des connexions fiables, à bandes passantes élevées. Une telle interconnexion va permettre la mise en place d'un réseau local sur lequel différentes applications propres à l'institution peuvent être accessibles en interne. Entre autres applications, on peut trouver un service de messagerie, un service de téléchargement, un service de sauvegarde sur le réseau, etc. Dans cette section, nous allons focaliser notre attention sur les systèmes de messagerie institutionnelle et particulièrement sur le système de messagerie que l'université du Burundi a mis à la disposition de sa communauté.
L'Université du Burundi est une institution qui se compose de cinq campus (MUTANGA, KIRIRI, ROHERO, KAMENGE et ZEGE) lesquels campus se composent à leur tour de plusieurs Facultés et Départements. Pour mettre en communication ces différents campus, deux possibilités s'offrent : on peut utiliser l'Internet ou bien constituer un réseau local de communication et n'utiliser l'Internet que pour sortir du réseau local. Vu le coût élevé de la connexion Internet et sa faible fiabilité, la deuxième option s'avère être la plus efficace. C'est dans ce but que l'Université du Burundi s'est dotée d'un système de messagerie propre portant le nom ub.edu.bi.

III.5.1.Système de messagerie institutionnelle

Un système de messagerie institutionnel est un système de messagerie dont la gestion revient à l'institution propriétaire et c'est ce dernier qui donne accès aux utilisateurs. Il existe deux types de messagerie institutionnels : système de messagerie centralisé et système de messagerie distribué. Le choix de l'un ou l'autre dépend des besoins de l'institution. Mais avant d'arriver à ces types, voyons ce qu'est le nom de domaine parce que c'est une partie qui a une grande signification dans l'adresse d'un système de messagerie. Le nom de domaine, c'est ce que vous trouvez dans la première partie de l'adresse d'une page web. C'est ce qui définit l'adresse de la page web. Ainsi, nous reconnaissons dans l'adresse suivante : http:// www.ub.edu.bi.

- http qui est le protocole utilisé.
- www.ub.edu.bi, qui est le nom de domaine de l'université du Burundi.

En pratique, le nom de domaine lui même est composé de plusieurs parties, séparées par un point. Voyons cela :
Le nom de domaine ub.edu.bi est en fait constitué de deux partie distinctes : le domaine de premier niveau ou top level domain (en anglais, souvent abrégé tld) .bi, .com, .net, .fr etc. et le nom du domaine lui même (ici : ub.edu). Lorsqu'une entreprise veut posséder un nom de domaine, elle achète un nom attaché au domaine de premier niveau, et le tarif dépend en fait du domaine de premier niveau. Une fois que l'on est titulaire d'un domaine, on peut encore créer des sous-domaines, c'est à dire des noms qui vont s'ajouter à gauche du nom de domaine, séparés par un point (dans ub.edu.bi ; ub est le sous-domaine). Cela éclairci l'identité du système de messagerie en question. Le propriétaire du nom de domaine peut créer (gratuitement) autant de sous-domaines qu'il le souhaite, et même des sous-sous-domaines. L'usage veut que pour les sites web on emploie le sous-domaine www, mais cela n'est rien de plus qu'une habitude. Chaque domaine complet, (avec son tld et ses sous-domaines éventuels) peut correspondre à un site web, à un serveur web, ou à un service particulier. Le nom de domaine permet de construire des adresses. Il s'agit en fait d'un centre de tri qui permet de joindre par tous les moyens possibles les différentes ressources ou services de notre système de messagerie (ub.edu.bi, par exemple). Ainsi, mon-nom@ ub.edu.bi prendra un chemin bien précis vers ma boîte mail, tandis que www.ub.edu.bi conduira par un autre chemin vers les pages du site web.

Les systèmes de messagerie institutionnels diffèrent de ceux à caractère commercial comme yahoo. En effet, dans les systèmes de messagerie institutionnels, le serveur (d'envoi ou de réception) se trouve dans l'institution tandis que pour les autres, dits ouverts, les serveurs sont éparpillés dans plusieurs pays. De plus, pour les systèmes de messagerie institutionnels les relais ne sont acceptés que sous l'autorisation de l'administrateur. En d'autres termes les relais sont habituellement interdits dans les systèmes de messagerie institutionnels.

III.5.1.1.Système de messagerie centralisé

Un système de messagerie centralisé se compose d'un important centre de données hébergeant toutes les ressources de serveurs, notamment les serveurs de catalogue global du service d'annuaire, les contrôleurs de domaine et les serveurs de messagerie. Le centre de données prend en charge tous les utilisateurs du système de messagerie, qu'ils se connectent localement ou à distance. Dans un système de messagerie centralisé les données sont hébergées et gérées dans un site central même si les utilisateurs se connectent à distance et les mises à niveau logicielles peuvent être transférées à partir d'un site central. Le centre des données comporte des périphériques de maintien de l'alimentation, tels que des onduleurs et des plans d'urgence. Pour envisager une conception centralisée, certaines conditions préalables doivent être remplies. Ces conditions sont principalement les suivantes :

- L'entreprise doit avoir les moyens suffisants pour supporter les coûts matériels du centre de données. Elle doit aussi maintenir un niveau de disponibilité et de redondance élevée dans le système pour pouvoir tolérer les pannes éventuelles. Cette organisation implique des coûts qui peuvent être largement compensés par les économies réalisées sur le plan du fonctionnement et de l'infrastructure, la réduction du temps d'indisponibilité et une meilleure évolutivité.
- L'entreprise doit concevoir des plans d'urgence. Lorsqu'on centralise les ressources de serveurs et de données à l'échelle de l'entreprise, on augmente le nombre de points de défaillance uniques. On doit concevoir des plans d'urgence au cas où le centre de données serait victime d'un sinistre.

- Le système de stockage des données doit être de grande capacité. Les volumes de données centralisées étant plus importants, on doit impérativement utiliser des systèmes de stockage fiables pour améliorer l'intégrité des données.
- L'entreprise doit garantir la sécurité physique des serveurs installés.

III.5.1.2. Système de messagerie distribué

Un système messagerie distribué est un système de messagerie dans lequel plusieurs succursales ou sites centralisés de petite taille sont généralement reliés à l'aide de connexions faibles à un concentrateur ou un centre de données d'entreprise. Les succursales comportent leurs propres serveurs d'envoi/réception, contrôleurs de domaine et serveurs de catalogue global. Un système de messagerie distribué est la solution généralement adoptée lorsque le réseau n'est pas à mesure de gérer le trafic vers un concentrateur central. Les serveurs de messagerie sont alors placés localement dans les succursales. Les besoins des utilisateurs peuvent également justifier cette option. En effet, si la connexion à un centre de données ne permet pas de satisfaire les utilisateurs en termes de besoins et de disponibilité, on n'a pas d'autres options que de placer les serveurs locaux dans des sites distants.

Les principales raisons de déploiement d'un système de messagerie distribué sont les suivantes :

- Les utilisateurs de l'entreprise sont dispersés sur plusieurs sites et des fois distants.
- L'infrastructure réseau de l'entreprise ne peut pas gérer le trafic vers un concentrateur central et cela pour tous les services de l'entreprise.
- Les besoins énormes des utilisateurs imposent la mise en place d'un serveur au niveau local pour leurs satisfaire.

Le système de messagerie centralisé facilite la gestion de la sécurité et donne par conséquent un degré de contrôle plus élevé. Le personnel chargé de la sécurité peut plus facilement tenir à jour les signatures de virus et prendre des mesures opportunes en cas d'incidents. Le système centralisé a également pour avantage de placer les serveurs dans un centre de données qu'on peut physiquement sécuriser. Dans un système de messagerie distribué, la sécurité physique des serveurs installés dans les succursales est d'une importance capitale. Dans une conception avec succursales, on doit assurer que les serveurs ne sont pas installés dans des endroits en libre accès et qu'ils sont physiquement sécurisés.

III.5.2. Caractéristiques et avantages

Le système de messagerie de l'Université du Burundi, comme tout autre système de messagerie institutionnelle, est un système à caractère professionnel. Ce qui signifie qu'il existe un administrateur qui contrôle tout et qui assure un suivi des mails échangés. Toute personne voulant une adresse électronique doit être reconnu par cet administrateur et ce dernier vérifie s'il remplit les conditions exigées pour en bénéficier. Cela permet d'empêcher qu'un utilisateur puisse créer des adresses électroniques comme il veut ; ce qui est le cas pour les systèmes de messagerie ouverts comme Yahoo, Gmail.

Chez Yahoo par exemple, le même nom GATOTO Placide peut être associé à plus d'une adresse électronique sans en justifier la raison. Mais chez ub.edu.bi, on ne peut pas faire cela sans avoir des raisons acceptables car l'utilisateur doit fournir une identification complète et

reconnue par l'administrateur. Il y a lieu de se demander le pourquoi de cette rigueur dans l'obtention d'une adresse électronique dans de tels systèmes de messagerie. A cela, il y a diverses raisons :

- En recevant un message provenant de kabura@ub.edu.bi par exemple avant même de lire le contenu par cette adresse, on reconnaît que le message provient de quelqu'un de l'Université du Burundi. C'est une étiquette professionnelle.
- Si par exemple un Département ou une Faculté entretient des relations avec une autre institution, les transactions effectuées entre ces deux utilisateurs sont fiables car chacun est informé de leur origine. S'il y a un problème, on sait à qui s'adresser.

Le système de messagerie professionnel présente des avantages multiples entre autres :

- la facilité des échanges entre différents partenaires ;
- il économise la bande passante Internet dans la mesure où il exploite l'Intranet. Même quand la connexion Internet n'est pas disponible, la connexion interne à l'institution reste fonctionnelle ;
- pour des messages devant sortir de l'intranet, il y a moyen de mettre en attente les mails, selon le temps configuré, en cas de coupure de connexion Internet ;
- la vitesse de téléchargement des données sur le serveur de ce système est suffisamment élevée (2×10^3 Mo/Sec au moment où c'est en terme de Ko/Sec pour les systèmes non institutionnels).
- il offre un suivi des mails, ce qui fait que s'il y a envoi d'un mail pouvant trahir l'institution, il y a moyen qu'il soit intercepté par l'administrateur ou bien si un message contient des virus, il peut être scanné.
- il offre une gestion centralisée des mails. En effet, si un mail rencontre un problème, l'administrateur peut l'identifier et le résoudre ou en informer l'expéditeur.

III.5.3. Protocoles d'envoi et de réception

Le protocole utilisé par le système de messagerie de l'Université du Burundi pour l'envoi des mails est le SMTP. Ce protocole est utilisé pour les mails circulant dans le réseau local au même titre que ceux destinés à sortir du réseau local.

La réception des mails dans le système ub.edu.bi s'effectue de deux manières et par des protocoles différents à savoir le POP3 et IMAP. Il y a lieu de se demander la raison de l'utilisation de ces deux protocoles et non l'un des deux. L'utilisation de l'un ou l'autre présente des avantages et des inconvénients. L'utilisation du POP3 pour la réception des mails présente un avantage dans l'économie de l'espace de stockage sur le serveur. Comme il a été déjà démontré, lorsqu'on reçoit un message avec le POP3, le message est retiré du serveur (effacé) c'est-à-dire qu'après la lecture on ne peut plus le récupérer. Cela présente un avantage du côté serveur parce qu'il évite la saturation mais un inconvénient pour l'utilisateur car il consulte le message une fois pour toute à partir du serveur.

Le protocole IMAP, quant à lui, présente un avantage pour l'utilisateur tandis qu'il est vu comme un problème pour un système qui possède une capacité de stockage limitée.

En effet, les mails reçus avec IMAP sont susceptibles d'être consultés en ligne. L'utilisateur, après consultation de son mail, a la possibilité de les laisser sur son serveur et il ne l'efface que quand il veut. C'est un inconvénient pour les systèmes de messagerie institutionnels possédant des capacités de stockage limités. Mais c'est un avantage pour l'utilisateur qui garde la possibilité de consulter ces mails autant qu'il veut.
Le système de messagerie de l'Université du Burundi sécurise les données transitant sur son réseau en utilisant les méthodes suivantes :

- le chiffrement : les données sont chiffrées en utilisant le protocole de sécurité TLS (TLSencryption) qui est une évolution de SSL appelée souvent SSL3. Il utilise l'algorithme de chiffrement 3DES.
- L'authentification : Le moyen utilisé pour identifier l'utilisateur est le mot de passe. Cette identification se fait à chaque nouvelle connexion. Mais cette identification n'est pas fiable dans la mesure où il peut être attaque par force brute.
- La dernière méthode utilisée pour assurer la sécurité est le fait que le serveur ub.edu.bi a été configuré pour ne pas accepter le relayage avec d'autres serveurs sauf en cas d'autorisation spéciale suite à une demande adressée à l'administrateur du système.

Le service informatique de l'Université du Burundi nous a révélé que leur système de messagerie compte aujourd'hui environ 150 utilisateurs et ces utilisateurs sont soit les professeurs de l'Université du Burundi soit les chefs de service de cette même institution. A voir l'effectif du personnel et des étudiants de cette institution, dépassant 10.000, on voit que les 150 utilisateurs représentent un effectif négligeable. Cela s'explique par le fait que ce système n'a pas de capacité suffisante en termes de stockage pour les accueillir tous. Même le peu d'espace qu'il y a doit être géré d'où l'utilisation du POP3 pour la réception des mails. Pour faire face à une demande accrue, il est prévu d'acheter d'autres disques durs hot plug pour le serveur de messagerie.

Nous avons aussi voulu connaître si les utilisateurs sont intéressés par ce système. Le service nous a dit que leurs clients sont satisfaits pour plusieurs raisons entre autres :

- La rapidité : les transactions se font à temps raisonnable, pas d'encombrements.
- Fiabilité : ils ont confiance en ce système.
- Redondance : pour tolérer les pannes, le service informatique de l'Université du Burundi utilise deux serveurs. Ainsi, si l'un tombe en panne, l'autre prend la relève.

Nous avons voulu connaître si les utilisateurs ne peuvent pas craindre que leurs messages soient révélés par l'administrateur mais on nous a dit que la déontologie professionnelle est là pour éviter ça.

III.6. Conclusion

Les systèmes de messagerie électronique sont un moyen de communication empreinté par les internautes pour échanger des messages. Il y a deux types de systèmes de messagerie : ceux centralisés et ceux distribués. Il faut remarquer aussi qu'il y a des systèmes de messagerie électronique à caractère professionnel dits propriétaires et ceux à caractère commercial dits ouverts. L'UB possède un système de messagerie électronique professionnel ub.edu.bi.
Les protocoles de messagerie les plus utilisés sont SMTP pour l'envoi et POP et IMAP pour la réception des messages électroniques. Ces protocoles transmettent généralement les messages en clair. Ce qui est un danger pour la sécurité des données échangées.

SSL intervient dans ces protocoles dans le but de les sécuriser. Il crée un tunnel où transitent les données échangées. Dans la sécurisation des échanges électroniques, le protocole SSL est soumis à un certain nombre d'attaques comme attaque par l'homme du milieu, attaque sur le CBC, etc. Ces attaques viennent limiter SSL et lui occasionne des faiblesses.

CHAPITRE IV. CONCLUSION GENERALE

Au début de notre travail, nous nous étions fixé l'objectif d'étudier le protocole SSL généralement utilisé pour sécuriser les transactions sur Internet. Dans le premier chapitre, nous avons exploité la notion de sécurité informatique et les concepts de base de la sécurité informatique à savoir la confidentialité, l'intégrité, l'authentification et la non répudiation des données. Dans ce même chapitre, nous avons abordé la notion de cryptographie qui est un outil important utilisé par le protocole SSL pour assurer la sécurité informatique. Au deuxième chapitre, nous avons exploré le protocole SSL et nous avons montré comment ce protocole assure les fonctions de base de la sécurité informatique. De plus, nous avons montré comment SSL intervient dans les systèmes de messagerie électronique pour les sécuriser. Dans ce même chapitre nous avons vu les différents protocoles de messagerie comme SMTP, POP et IMAP. Nous avons aussi montré les limites et les faiblesses de SSL en matière de sécurité. Pour concrétiser notre travail, nous avons étudié le système de messagerie de l'université du Burundi.

Nous ne saurons pas terminer sans attirer l'attention des utilisateurs effectuant des transactions électroniques avec l'Internet. Qu'ils sachent que quand ils effectuent des transactions avec ce réseau, ils ne sont pas isolés du monde extérieur. Cela pour dire qu'ils peuvent être exposés aux attaquants qui peuvent intercepter, voler ou modifier les données échangées. Les utilisateurs doivent aussi savoir que les attaquants travaillent jour et nuit pour améliorer leurs techniques d'attaques. Ils sont toujours en compétition avec le développement de l'informatique.

La sécurisation des systèmes de messagerie électronique est un domaine complexe et qui électronique est un domaine complexe et qui évolue du jour le jour. Cela s'explique par le fait que l'Internet est toujours en évolution. Pour cela, nous voudrions encourager tout un chacun à se lancer à l'étude de ce protocole qui est actuellement préféré par nombre d'utilisateurs. Que ceux qui vont nous suivre aient le courage et l'ambition de conquérir le monde informatique et en particulier la Sécurité Informatique afin de découvrir les faits que nous n'avons pas pu exploiter au cours de notre travail. Actuellement, l'Université du Burundi s'est dotée d'un système de messagerie propre, ub.edu.bi, pour permettre aux professeurs, aux étudiants et autres ayants droit à ce système de pouvoir communiquer facilement et avec plus de sécurité pour des raisons que nous avons évoquées dans notre travail. Pour ce, nous tenons à leur rassurer pour qu'ils ne doutent pas de la sécurité de ce système de messagerie électronique qui utilise, pour effectuer ses transactions, les protocoles que nous avons exploités dans notre travail à savoir SMTP, POP et IMAP. Cela veut dire que la sécurité qu'ils trouvent dans les systèmes de messagerie électronique ouverts comme Yahoo est celle-là même qui se trouve dans le système de messagerie électronique de l'Université du Burundi. A cela s'ajoutent les avantages que nous avons énoncés dans notre travail. Nous voudrions terminer par lancer un appel à l'Université du Burundi et en particulier au département de physique de mettre en place tous les moyens nécessaires pour que les étudiants puissent arriver là où nous ne sommes pas arrivé car l'Informatique est un domaine qui évolue du jour le jour. Et de plus, il est évident que le développement de toute personne dépendra, demain, de ses connaissances en informatique. Merci !

ANNEXES

ANNEXE A : RECUPERATION DES MESSAGES ELECTRONIQUES AVEC IMAP

Cette section montre le dialogue entre le serveur et le client lors de la récupération d'un message avec le protocole IMAP [8].

C: 10 capability

S: * CAPABILITY IMAP4REV1 LITERAL+ IDLE UIDPLUS NAMESPACE CHILDREN MAILBOX-REFERRALS BINARY
UNSELECT ESEARCH WITHIN SCAN SORT THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND
SASL-IR LOGIN-REFERRALS STARTTLS LOGINDISABLED

C: 10 OK CAPABILITY completed.

10 login chris@maison.mrs epikoi
10 OK Logged in.
20 list "*" "*"
S: * LIST (\HasNoChildren) "." "Trash"
* LIST (\HasNoChildren) "." "INBOX"
C: 20 OK List completed.
30 create Dossier_1
30 OK Create completed.
40 create Dossier_1.SousDossier_1
40 OK Create completed.
50 create INBOX.Rangement
50 OK Create completed.
60 subscribe INBOX.Rangement
60 OK Subscribe completed.
70 subscribe Dossier_1
70 OK Subscribe completed.
80 subscribe Dossier_1.SousDossier_1
80 OK Subscribe completed.
90 select INBOX
S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* OK [PERMANENTFLAGS (\Answered \Flagged \Deleted \Seen \Draft *)] Flags permitted.
* 3 EXISTS
* 0 RECENT
* OK [UNSEEN 1] First unseen.
* OK [UIDVALIDITY 1216295789] UIDs valid
* OK [UIDNEXT 10] Predicted next UID
C: 90 OK [READ-WRITE] Select completed.
100 rename Dossier_1 Archive
100 OK Rename completed.
110 list "*" "*"
S: * LIST (\HasNoChildren) "." "Trash"
* LIST (\HasChildren) "." "INBOX"

```
* LIST (\HasChildren) "." "Archive"
  * LIST (\HasNoChildren) "." "INBOX.Rangement"
   * LIST (\HasNoChildren) "." "Archive.SousDossier_1"
C: 110 OK List completed.
   120 rename Archive.SousDossier_1 Archive.2007-2008
   120 OK Rename completed.
   130 uid fetch 1:* (FLAGS)
S: * 1 FETCH (FLAGS () UID 7)
   * 2 FETCH (FLAGS () UID 8)
   * 3 FETCH (FLAGS () UID 9)
C: 130 OK Fetch completed.
   140 copy 1:2 Archive.2007-2008
   140 OK Copy completed.
   150 copy 3 INBOX.Rangement
   150 OK Copy completed.
   160 store 1:3 +FLAGS (\Deleted)
S: * 1 FETCH (FLAGS (\Deleted))
   * 2 FETCH (FLAGS (\Deleted))
   * 3 FETCH (FLAGS (\Deleted))
C:  160 OK Store completed.
    170 expunge
 S:* 3 EXPUNGE
    * 2 EXPUNGE
    * 1 EXPUNGE
 C: 170 OK Expunge completed.
    180 logout
 S: * BYE Logging out
C: 180 OK Logout completed.
   Connection closed by foreign host.
```

Exemple.6: Récupération d'un message avec IMAP [21]

La commande CAPABILITY ouvre le dialogue entre le client et le serveur.
La première ligne est la demande que le client adresse au serveur pour que ce dernier lui donne la liste des possibilités qu'il possède. C'est la liste qui est donnée à la deuxième ligne. Remarquons que la liste des possibilités supportées par le serveur varie d'un serveur à un autre.
La ligne suivante montre que le client est satisfait. Le client va alors s'identifier (login chris@maison.mrs epikoi) et demande au serveur de lui montre la liste des répertoires souscrits et existants. Ce que le serveur fait (Trash et INBOX).
Le client montre qu'il est satisfait (List completed) et demande la création d'un dossier (Dossier_1, Dossier_1.SousDossier_1, INBOX.Rangement) de plus il demande que ces dossiers soient souscrits (il demande d'être abonné à ces dossiers).

La commande SELECT permet de sélectionner un dossier et d'avoir un compte rendu de son contenu. Voyons ce qui se passe avec la commande SELECT:
La réponse à la requête lancée par le client nous donne la liste des Flags. Le serveur lui indique aussi que ces flags sont permanents et permis. Cela veut dire que le client a accès à ces flags et peut les utiliser en permanence.
Le serveur annonce au client qu'il a trois messages. Parmi eux aucun n'est récent, ils étaient déjà présents lors de la dernière consultation, mais le message1 n'a pas été lu ;

Le client est informé aussi que les UIDs (adresses des messages) sont valides et que le prochain sera le dixième qu'il reçoit. Cela veut dire qu'il a déjà reçu 9 messages dans sa boîte depuis qu'il a son adresse électronique.

Le client demande pour le moment de renommer Dossier_1 Archive (rename Dossier_1 Archive) et indique que sa requête est terminée.
Le client lance de nouveau la commande LIST et cela lui donne le rapport de toutes les actions qu'il a accomplies. Après avoir vu ce rapport, il demande que le dossier Archive soit renommé (Archive.2007-2008).
Cette action terminée, il lance la commande uid et fetch. Le uid fetch signifie que le client veut voir l'état de ses messages. Le serveur lui informe que les trois messages sont, respectivement le septième,huitième et neuvième (UID 7, UID 8, UID 9).
Le client est satisfait (Fetch completed) et demande que les messages (7) et (8) soient copiés dans le dossier Archive.2007-2008 et le (9) dans INBOX. Cela fait, le client demande que les messages 7, 8, et 9 soient marqués comme devra être effacés (store 1:3 +FLAGS (\Deleted)). Il lance ensuite la commande EXPUNGE servant à effacer les messages comportant les flags (\Deleted). Le serveur réplique en lui montra que les trois messages ont été effacés (3EXPUNGE, 2EXPUNGE, 1EXPUNGE) et le client montre qu'il est satisfait. Le client demande enfin la fermeture de la connexion. La commande "OK" montre que la demande s'est bien déroulée. Sinon, on verrait apparaître "NO" à la place de "OK". Et s'il y a eu erreur de protocole c'est "BAD" qui est utilisé. BYE indique que le serveur va fermer la session.

ANNEXE B : LE PROTOCOLE IMAP SECURISE PAR SSL

Le protocole IMAP en ses versions 1, 2, et 3 utilisé pour la récupération des courriels se fait généralement en claire. Ce qui est un danger pour la sécurité des données échangées car elles peuvent être interceptées par des personnes malveillantes. Tout comme pour le SMTP et POP, on peut sécuriser IMAP via SSL par création d'un tunnel SSL. La récupération sécurisée des courriels via IMAP est obtenue en utilisant la commande STARTTLS permettant de chiffrer automatiquement les données échangées.
Revenons au dialogue précèdent en ses 6 premières lignes, où le logiciel client demande au serveur IMPAP4 rev1 ses possibilités :

C: 10 capability

S: CAPABILITY IMAP4 rev1 UIDPLUS CHILDREN NAMESPACE

THREAD = ORDER SUBJECT THREAD = REEFERENCES

SORT QUOTA IDLE ACL ACL2 = UNION STARTTLS

LOGINDISABLED

C: 10 OK CAPABILITY completed

Nous remarquons la présence de STARTTLS et LOGINDISABLED. La commande STARTTLS permet de passer en mode chiffré et l'autre, LOGINDISABLED, indique que la commande LOG qui est utilisée pour les transactions en clair n'est pas acceptée par le serveur. Cela veut dire que lors de la négociation TLS, le serveur doit présenter un certificat (avec sa clé publique) et le client doit accepter ce certificat. Ce qu'il faut savoir c'est qu'avec la version actuelle du protocole IMAP4 (IMAP4 rev1) la sécurité des données est assurée grâce à cette commande STARTTLS qui n'est pas optionnelle dans IMAP4 rev1. Dans le cas des machines qui ne supportent pas TLS on fait recours aux versions antérieures du protocole IMAP pour recevoir leurs messages en l'absence du chiffrement.

ANNEXE C : PARADOXE DES ANNIVERSAIRES

Le paradoxe des anniversaires, dû à Richard von Mises, est à l'origine une estimation probabiliste du nombre de personnes que l'on doit réunir pour avoir une chance sur deux que deux personnes de ce groupe aient leur anniversaire le même jour de l'année.

La clé consiste à se demander quelles sont les chances qu'aucune paire de personnes ne soit née le même jour. Pour chaque personne ajoutée dans la pièce, le nombre de dates non déjà prises diminue. La première personne a donc 365 choix, la deuxième 364, la troisième 363, la quatrième 362, et ainsi de suite. Pour bien comprendre cela, nous allons partir d'un cas concret. Considérons un groupe de k personnes. La question qui se pose est de calculer la probabilité pour que, dans un groupe de k personnes, ces personnes aient toutes un jour d'anniversaire différent. Nous avons les cas suivants :

- Quand on a 2 personnes, la première peut avoir son anniversaire n'importe quand, la seconde n'importe quel autre jour. On a donc :

$$p_2 = \frac{364}{365} = 1 - \frac{1}{365}$$

- Quand on a 3 personnes, la troisième doit avoir son anniversaire un jour différent des 2 autres :

$$p_3 = \left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right)$$

- On peut réitérer le raisonnement. Pour un groupe de k personnes, on obtient :

$$p_k = \left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right)\ldots\left(1 - \frac{k-1}{365}\right)$$

Une petite application numérique donne :

Nombre de personnes	Probabilité pour que les anniversaires tombent tous un jour différent
1	1
2	0.99
5	0.97
10	0.88
20	0.58
22	0.52
23	0.49
30	0.29
50	0.03

Il ne faut donc que 23 personnes pour qu'il y ait plus d'une chance sur 2 pour que 2 personnes aient leur anniversaire le même jour, contrairement à ce que l'intuition laisse présumer. A partir de 50 personnes, il n'y a que 3% de chances que tous les anniversaires diffèrent.

Ce paradoxe a son importance en cryptographie, lorsqu'on étudie les fonctions de hachage. Une telle fonction calcule le résumé d'un texte, en vue de le signer. Pour que cette méthode soit fiable, il ne faut pas que quelqu'un puisse produire 2 textes aux sens très différents, mais donnant le même résumé (ex : l'attaquant produit 2 textes, et introduit de ci, de là, des espaces jusqu'à obtenir le même résumé).

Si le haché (= le résumé) est codé sur b bits, il y a 2^b résumés possibles. Si l'on prend k textes différents, la probabilité pour que 2 textes aient le même haché est donc :

$$1-\left(1-\frac{1}{2^b}\right)\left(1-\frac{2}{2^b}\right)...\left(1-\frac{k-1}{2^b}\right)$$

Combien l'attaquant doit-il essayer de textes avant de trouver le même résumé avec une probabilité d'au moins un demi? On va faire quelques approximations, et d'abord, en vue de l'égalité de Taylor :

$$\left(1-\frac{j}{2^b}\right) \square \exp\left(\frac{-j}{2^b}\right)$$

On a alors :

$$\left(1-\frac{1}{2^b}\right)\left(1-\frac{2}{2^b}\right)...\left(1-\frac{k-1}{2^b}\right) \square \exp\left(\frac{-k(k-1)}{2^b}\right) \qquad \textbf{[22]}$$

Il faut donc que :

$$\exp\left(\frac{-k(k-1)}{2^b}\right) \leq \frac{1}{2}$$

Une application numérique donne :

Taille du haché (en bits)	Nombre de textes à essayer
8	13
16	213
32	54562
64	$9{,}6\times10^{9}$
128	$1{,}5\times10^{19}$
160	$1{,}0\times10^{24}$
256	$2{,}8\times10^{38}$

REFERENCES BIBLIOGRAPHIQUES ET ELECTRONIQUES

1. http://fr.wikipedia.org/wiki/Histoire_d%27Internet
2. http://www.infoclick/crypto/crypto.htm
3. http://www.commentcamarche.net/contents/internet/smtp.php3
4. **Arnaud-F. Fausse** : La signature électronique : Transactions et confiance sur Internet ; Edition Dunod (8 janvier 2001) ; 5, rue Laromiguière Panthéon.
5. http://securite.reseaux-telecoms.net
6. http://www.bibmath.net/crypto/moderne/feistel.php3
7. http://www.bibmath.net/crypto/moderne/des.php3
8. http://www.bibmath.net/crypto/moderne/aes.php3
9. http://fr.wikipedia.org/wiki/Th%C3%A9or%C3%A8me_de_Bachet-B%C3%A9zout
10. **Ryan Russell**: Stratégies anti-hackers; Edition Eyrolles 2001; 61, Bld Saint-Germain 75240 Paris Cedex 05.
11. **Solange Ghernaouti-Hélie** : Sécurité Internet ; Edition Dunod ; Paris (3 Novembre 2000)
12. http://fr.wikipedia.org/wiki/infrastructure_%C3%AO_cl%C3%A9s_publiques
13. http://www.authsecu.com/ssl-tls/ssl-tls.php
14. **Guy Pujolle** ; Cours réseaux et télécoms ; Edition Eyrolles ; mars 2010(3ème édition).
15. **Joe Habraken et Matt Hayden :** Les réseaux ; 3ème Edition publiée par Pearson Education ; juillet 2007.
16. http://www.arobase.org/docs/rfc2060
17. **Olivier Pavie** : Exploiter votre messagerie électronique ; Edition Osmane Eyrolles Multimédia 2000 ; 1, Rue Thénard-7500.
18. http://www.commentcamarche.net/contents/internet/smtp.php3
19. **Eric Larcher** : L'Internet Sécurisé ; Editions Eyrolles2000 ; 61, Bld Saint-Germain 75240, Paris Cedex 05.
20. http://www.commentcamarche.net/contents/internet/smtp.php3#le-protocole-imap
21. http://www.hsc.fr/ressources/presentations/.../eurosec04-msg.pdf
22. http://fr.wikipedia.org/wiki/Paradoxe_des_anniversaires