Le protocole SSL
Secure Socket Layer

S. LAZAAR
ENSA de Tanger

- Le protocole SSL (Secure Socket Layer) permet la transmission de données chiffrées sur le réseau Internet.

- C'est une création de Netscape, l'une des premières sociétés à avoir développé un navigateur grand public.

- La clé de cryptage utilisée est une clé publique **RSA**, se basant sur un ensemble d'opérations sur des nombres premiers.

La clé de cryptage:

- se compose dans les faits de deux clés :
- une pour le codage, et une pour le décodage.

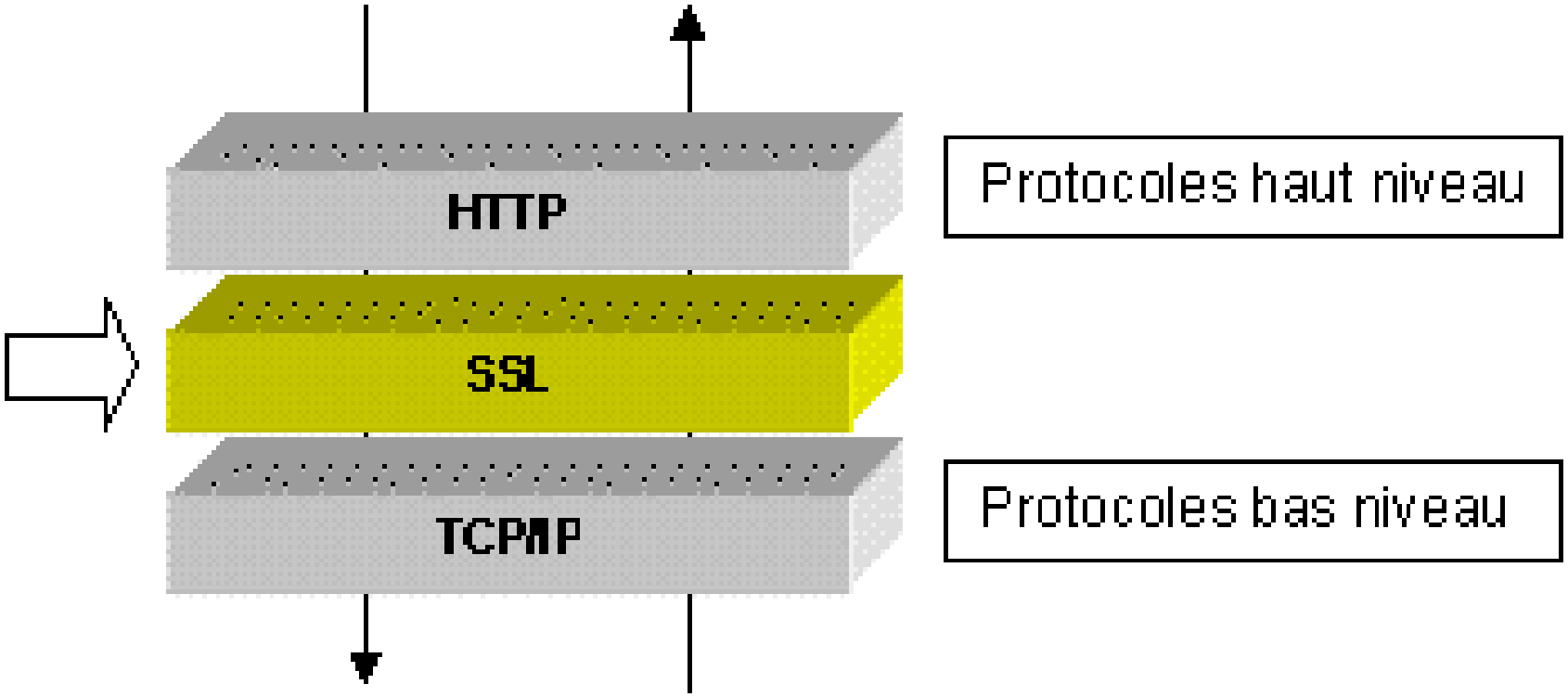
Cette technologie est acceptée par les navigateurs, mais aussi par les serveurs.

- Un organisme est chargé de mettre en place des certificats de confiance pour tous les sites qui utilisent le protocole SSL

- But:

bien identifier un site et l'organisme qui le gère.

Architecture de réseau utilisant SSL



- Comment ça marche SSL ?

SSL est basé sur 2 protocoles:

- **SSL Handshake protocol**: avant de communiquer, les 2 programmes SSL négocient des clés et des protocoles de chiffrement communs.

- **SSL Record protocol**: Une fois négociés, ils chiffrent toutes les informations échangées et effectuent divers contrôles.

Le principe de fonctionnement est: serveur - client

- Le navigateur se connecte à un serveur sécurisé: **une clé de cryptage unique est mise en place tout au long de la transaction entre le serveur et le navigateur.**
-
- Le navigateur envoie des données cryptées à destination du serveur, le seul à déchiffrer les informations reçues, grâce à la mise en place d'une clé d'échange unique.

- Le serveur envoie un avis de bonne réception de l'information.

- Le client vérifie la validité du certificat (donc l'authenticité du marchand),
- Il crée une clé secrète aléatoire , chiffre cette clé à l'aide de la clé publique du serveur, puis lui envoie le résultat (**la clé de session**).
- Le serveur est en mesure de déchiffrer la clé de session avec sa clé privée.
- Les deux entités sont en possession d'une clé commune dont ils sont seuls connaisseurs.

  Le reste des transactions peut se faire à l'aide de clé de session, garantissant l'intégrité et la confidentialité des données échangées.

Utilité: e-commerce

- SSL une solution logicielle qui chiffre le numéro de carte bancaire lorsqu'il transite sur le réseau depuis le poste du client.

- Il est décrypté sur le serveur du marchand.

- Chaque transaction implique un coût fixe de 0,15 à 0,30 €, auquel il faut ajouter les pourcentages prélevés par la banque du commerçant, l'émetteur de la carte de crédit, le serveur de paiement Internet, etc...

- Ce coût fixe pose un vrai problème pour les paiements de faible montant

- SSL est actuellement le système le plus fiable puisque la totalité du numéro de la carte de crédit est cryptée.
- Le cryptage SSL est aujourd'hui utilisé par le plus grand nombre de commerçants en ligne à travers le monde et, de part sa notoriété, est amené à se développer à un niveau encore plus élevé.
- Le protocole SSL permet d'authentifier le commerçant, qui doit obligatoirement détenir une clé secrète correspondant à la clé publique de chiffrement de la connexion.

- Le protocole SSLne permet pas d'authentifier le client qui conserve la faculté de répudier la transaction.

Moralité : le commerçant supporte tous les risques liés à la fraude.

- Le système SSL est indépendant du protocole utilisé, ce qui signifie qu'il peut aussi bien sécuriser des transactions faites sur le Web par le protocole HTTP que des connexions via le protocole FTP, POP …

- SSL agit telle une couche supplémentaire, permettant d'assurer la sécurité des données, située entre la couche application et la couche transport

Avantages:

SSL est transparent pour l'utilisateur:

Un utilisateur du navigateur internet pour se connecter à un site de commerce électronique sécurisé par SSL enverra des données chiffrées sans aucune manipulation nécessaire de sa part

- La quasi intégralité des navigateurs supporte désormais le protocole SSL.

Netscape Navigator affiche un cadenas verrouillé pour indiquer la connexion à un site sécurisé par SSL et un cadenas ouvert dans le cas contraire

Microsoft Internet Explorer affiche un cadenas uniquement lors de la connexion à un site sécurisé par SSL.

Au milieu de l'année 2001, le brevet de SSL appartenant jusqu'alors à Netscape a été racheté par l'*IETF* (*Internet Engineering Task Force*)

Il a été rebaptisé pour l'occasion **TLS** (*Transport Layer Security*).

Fonctionnement de SSL 2.0

- ✓ La sécurisation des transactions par SSL 2.0 est basée sur un échange de clés entre client et serveur.
- ✓ La transaction sécurisée par SSL se fait selon le modèle suivant :

- Le client se connecte au site marchand sécurisé par SSL et lui demande de s'authentifier.
- Le client envoie également la liste des cryptosystèmes qu'il supporte, triée par ordre décroissant selon la longueur des clés.
- Le serveur a réception de la requête envoie un certificat au client, contenant la clé publique du serveur, signée par une autorité de certification (CA), et le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible

  La longueur de la clé de chiffrement - 40 bits ou 128 bits - sera celle du cryptosystème commun ayant la plus grande taille de clé.

SSL 3.0

- SSL 3.0 vise à authentifier le serveur vis-à-vis du client

Note : Dans Firefox 2, le support de SSL 2.0 est désactivé par défaut, en faveur de SSL 3.0.

SSL permet d'assurer les services de sécurité suivants:

- **confidentialité** : obtenue par l'utilisation d'algorithmes de chiffrement symétriques de blocs comme DES, FORTEZZA, IDEA, 3DES ou RC2, ou par des algorithmes à chiffrement symétrique de flots comme RC4

- **intégrité** : assurée par l'utilisation de MACs (Message Authentication Code) basés sur les fonctions de hachage MD5 (16 octets) ou SHA-1 (20 octets)

- **authentification** : SSL permet l'authentification des 2 entités (authentification client facultative) basé sur des certificats X.509, et l'authentification des données grâce aux MACs.

Les sous-protocoles de SSL

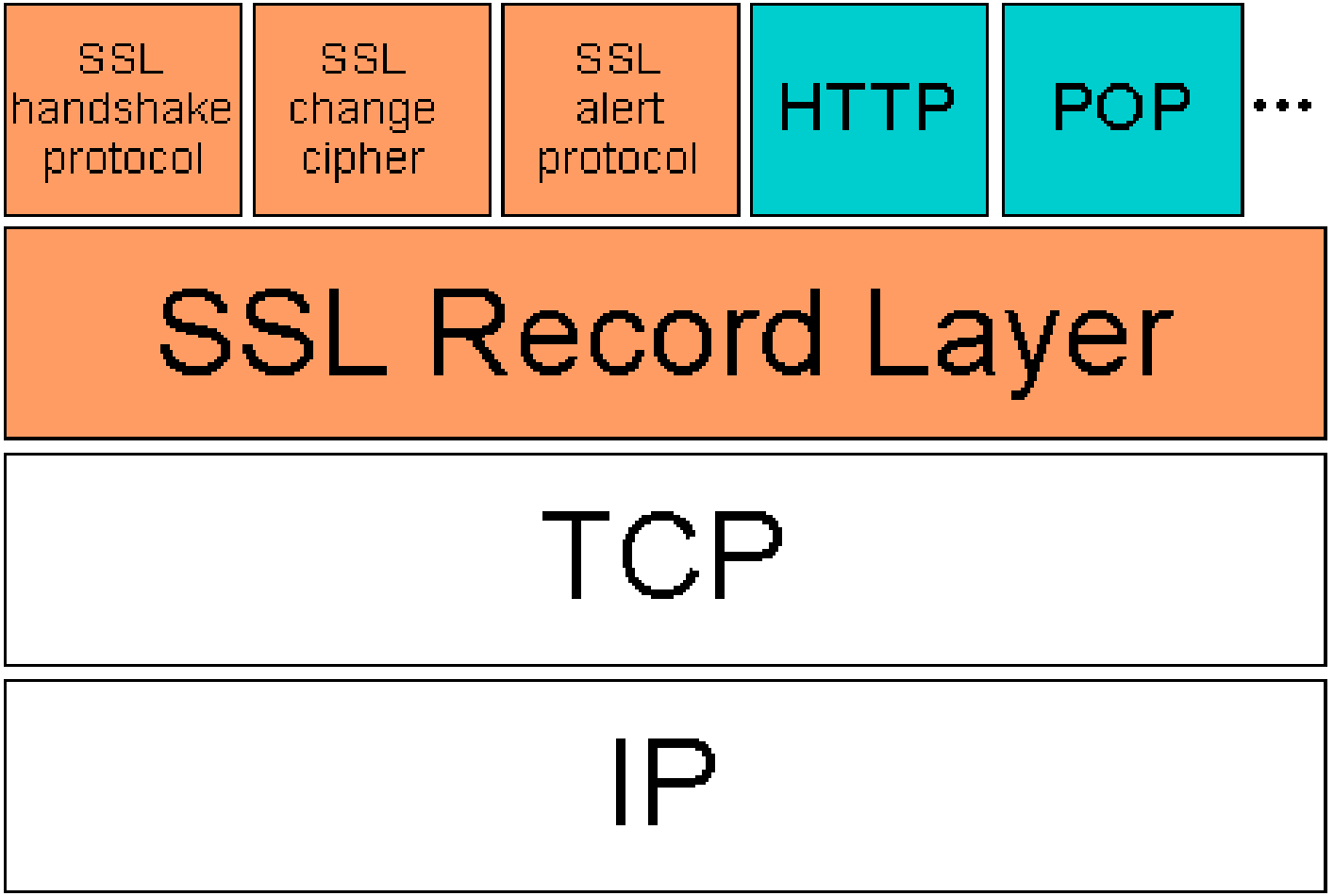
Le protocole SSL est constitué de quatre sous-protocoles:

1) **Handshake** qui permet l'authentification mutuelle du client et serveur, la négociation des algorithmes de chiffrement, de hachage, et l'échange des clés symétriques qui assurent le chiffrement.

2) le protocole **SSL Change Cipher Spec**

3) le protocole **SSL Alert**

4) le protocole **SSL Record**


SSL handshake protocol
SSL change cipher
SSL alert protocol
HTTP
POP
...
SSL Record Layer
TCP
IP

Le protocole Handshake:

- Ce protocole permet l'authentification obligatoire du serveur, du client est optionnelle,

- Il permet de négocier pour choisir les suites de chiffrement qui seront utilisées lors de la session.

Le protocole ChangeCipherSpec (CCS):

- Ce protocole comprend un seul et unique 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 Protocole SSLRecord:

Ce protocole intervient après l'émission du message ChangeCipherSpec. Il permet de garantir :

- la confidentialité à l'aide de chiffrement des données

- l'intégrité à l'aide de génération d'un condensat.

Le protocole SSL Alert:

- Ce protocole génère des messages d'alerte suite aux erreurs que peuvent s'envoyer le client et le serveur.

- Les messages sont composés de 20 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.

Liste des Messages du protocol Alert:

Les erreurs fatales sont:

- **bad_record_mac:** réception d'un MAC erroné
- **decompression_failure**: les données appliquées à la fonction de compression sont invalides
- **handshake_failure**: impossibilité de négocier les bons paramètres
- **illegal_parameter**: un paramètre échangé au cours du protocole Handshake ne correspond 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 précisés 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

- ssldump is an SSLv3/TLS network protocol analyzer.
- It identifies TCP connections on the chosen network interface and attempts to interpret them as SSLv3/TLS traffic.
- When it identifies SSLv3/TLS traffic, it decodes the records and displays them in a textual form to stdout.

Sample Output

- Here's an example trace generated by ssldump.

```
New TCP connection #3: localhost(3638) <-> localhost(4433)
3 1 0.0738 (0.0738) C>S Handshake ClientHello
3 2 0.0743 (0.0004) S>C Handshake ServerHello
3 3 0.0743 (0.0000) S>C Handshake Certificate
3 4 0.0743 (0.0000) S>C Handshake ServerHelloDone
3 5 0.0866 (0.0123) C>S Handshake ClientKeyExchange
3 6 0.0866 (0.0000) C>S ChangeCipherSpec
3 7 0.0866 (0.0000) C>S Handshake Finished
3 8 0.0909 (0.0043) S>C ChangeCipherSpec
3 9 0.0909 (0.0000) S>C Handshake Finished
3 10 1.8652 (1.7742) C>S application_data
3 11 2.7539 (0.8887) C>S application_data
3 12 5.1861 (2.4321) C>S Alert warning close_notify
3 5.1868 (0.0007) C>S TCP FIN 3 5.1893 (0.0024) S>C TCP
  FIN
```

Les variables d'état d'une session SSL

Une session SSL est définie par les variables suivantes:

- **Session ID** (identifiant de session): séquence arbitraire de 32 octets choisie par le serveur pour identifier une session.
- **Peer certificate** (certificat du pair): certificat X 509 du correspondant (soit pour un serveur ou un client).
- **Compression method** l'algorithme de compression utilisé, NULL pour l'instant (ce champ reste vide)
- **Cipher spec:** suite de chiffrement définit les algorithmes de chiffrement et de hachage
- **MasterSecret :** clé de 48 octets partagée entre le client et le serveur.
- **Is resumable** (drapeau): flag qui indique s'il est possible d'ouvrir de nouvelles connexions sur la session en question.

Les paramètres qui définissent une connexion SSL

- **Server_random et Client_random**: deux nombres aléatoires de 32 octets, générés par le client et le serveur lors de chaque connexion.
- **Server_MAC_write_secret**: clé secrète utilisée par le serveur pour calculer les MACs
- **Client_MAC_write_secret**: clé secrète utilisée par le client pour calculer les MACs
- **Server_write_key**: clé symétrique utilisée par le serveur pour le chiffrement des données.
- **Client_write_key**: clé symétrique utilisée par le client pour le chiffrement des données.
- **Initialization vectors**: vecteur d'initialisation pour le chiffrement par bloc en mode CBC (Cipher Bloc Chaining), l'un du coté serveur et l'autre du coté client.
- **Sequence number**: chaque message est numéroté, l'un pour le serveur, l'autre par le client, et chacun codé sur 8 octets.

Les attaques et faiblesses de SSL

- SSL est théoriquement vulnérable aux attaques par **force brute** en cas d'utilisation de clés 40

  (utiliser des clés de 128 bits).

- SSL est **très vulnérable** aux attaques par le milieu (**man in the middle**): l'attaquant intercepte (physiquement) la requête du client et se fait passer pour le serveur auprès de lui, tout en se faisant passer pour un client auprès du serveur légitime:

  Il reçoit donc la totalité du flux supposé protégé.

- SSL est faible dans le sens où il n'impose pas l'authentification client

- SSL est faible car il présente des souplesses dans son implémentation, notamment en ce qui concerne la **vérification des certificats** des serveurs.

  En effet, le client devrait vérifier la signature, la validité ainsi que le statut de révocation du certificat, s'assurer que le CA concerné appartient bien aux CAs auxquels il fait confiance.

- Il peut permettre à un attaquant de substituer sa propre clé publique à celle du serveur original puis de rediriger le traffic vers son faux site tout en faisant croire au client qu'il communique bien avec son serveur légitime.

- SSL est éventuellement vulnérable à des attaques plus poussées, basées sur des propriétés cryptographiques.

  Il existe des attaques par **brute-force** sur l'échange de la clé symétrique ou des attaques dites de **rollback:**

✓ l'attaquant cherche à modifier le choix des algorithmes d'échanges de clés de façon à ce que les 2 entités n'utilisent pas les mêmes.

✓ l'attaquant pourra déchiffrer le message car les paramètres fournit par le serveur dans le cas d'un algorithme n'offrent aucune sécurité si on les applique à un autre

Attaque Brute force:

Généralement les mots de passe de la plupart des logiciels sont stockés cryptés dans un fichier.

- Pour obtenir un mot de passe, il suffit de récupérer ce fichier et de lancer un logiciel de brute force cracking.

Ce procédé consiste à tester de façon exhaustive toutes les combinaisons possibles de caractères (alphanumériques + symboles), de manière à trouver au moins un mot de passe valide.

Cette attaque se base sur le fait que n'importe quel mot de passe est crackable. Ce n'est qu'une question de temps.

Implémentations:

Plusieurs offres commerciales du serveur SSL sont disponibles, par exemple:

- SSLeay (open source)
- Netscape Entreprise Server
- Apache
- Oracle Web Application Server
- Internet Information Server (IIS)
- Lotus Domino d'IBM
- Java Server de Sun Microsystems

Qu'est-ce qu'une fonction de hachage ?

- Une **fonction de hachage** (*fonction de condensation*) est une fonction permettant d'obtenir un condensé (*condensat* ou *haché* ou *message digest*) d'un texte:

  Une suite de caractères assez courte représente le texte qu'il condense.
- La fonction de hachage doit être telle qu'elle associe un et un seul haché à un texte en clair (la moindre modification du document entraîne la modification de son haché).
- C'est une fonction à sens unique **(*one-way function*)** , **impossible de retrouver le message original à partir du condensé.**
- S'il existe un moyen de retrouver le message en clair à partir du haché, la fonction de hachage est dite « à brèche secrète ».

Le haché représente en quelque sorte l'*empreinte digitale* (*finger print*) du document.

Algorithmes de hachage les plus utilisés:

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

- **SHA** (***Secure Hash Algorithm***), pouvant être traduit par *Algorithme de hachage sécurisé* crée des empreintes d'une longueur de **160 bits**

  SHA-1 est une version améliorée de SHA (1994), produit une empreinte de 160 bits à partir d'un message d'une longueur maximale de 2^{64} bits en le traitant par blocs de 512 bits.

Vérification d'intégrité

En expédiant un message accompagné de son haché, il est possible de garantir l'intégrité d'un message i.e le destinataire peut vérifier que le message n'a pas été altéré durant la communication:

Lors de la réception du message, il suffit au destinataire de calculer le haché du message reçu et de le comparer avec le haché accompagnant le document.

Si le message (ou le haché) a été falsifié durant la communication, les deux empreintes ne correspondront pas.

Le scellement des données:

L'utilisation d'une fonction de hachage permet de vérifier que l'empreinte correspond bien au message reçu, mais rien ne prouve que le message a bien été envoyé par celui que l'on croit être l'expéditeur.

Pour garantir l'authentification du message, il suffit à l'expéditeur de chiffrer (on dit *signer*) le condensé à l'aide de sa clé privée (**le *haché signé* est appelé sceau**) et d'envoyer le sceau au destinataire.

A réception du message, il suffit au destinataire de déchiffrer le sceau avec la clé publique de l'expéditeur, puis de comparer le haché obtenu avec la fonction de hachage au haché reçu en pièce jointe. Ce mécanisme de création de sceau est appelé *scellement*.

Conclusion

- ✓ Le protocole SSL est actuellement le seul protocole de sécurisation déployé et utilisé à grande échelle.
- ✓ Son grand avantage étant sa transparence
- ✓ Il garantit l'authentification, la confidentialité et l'intégrité des données.
- ✓ Avec son architecture modulaire, il ne se limite pas à des applications traditionnelles:

Il intègre les réseaux sans fil comme le WAP (Wirless Transport Layer) sous le nom WTLS

(Wireless Transport Layer Security).

ANNEXE

openSSL

openSSL est une boîte à outils cryptographiques

implémentant les protocoles SSL et TLS qui offre

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

- la création de clés RSA, DSA (signature)
- la création de certificats X509
- le calcul d'empreintes (MD5, SHA, RIPEMD160, ...)
- le chiffrement et déchiffrement (DES, IDEA, RC2, RC4, Blowfish, ...)
- la réalisation de tests de clients et serveurs SSL/TLS
- la signature et le chiffrement de courriers (S/MIME)

- **Blowfish** est un algorithme de chiffrement à clé symétrique par blocs conçu par Bruce Schneier en 1993.
- Il tire son nom du poisson-lune japonais (ou *fugu*), qui en est également l'emblème.

- Blowfish utilise une taille de bloc de 64 bits et la clé de longueur variable peut aller de 32 à 448 bits.
- Elle est basée sur l'idée qu'une bonne sécurité contre les attaques de cryptanalyse peut être obtenue en utilisant de très grandes clés pseudo-aléatoires.

- Blowfish présente une bonne rapidité d'exécution excepté lors d'un changement de clé, il est environ 5 fois plus rapide que TripleDES et deux fois plus rapide que IDEA.
- Malgré son âge, il demeure encore solide du point de vue cryptographique avec relativement peu d'attaques efficaces sur les versions avec moins de tours. La version complète avec 16 tours est à ce jour entièrement fiable et la recherche exhaustive reste le seul moyen pour l'attaquer.
- Il est utilisé dans de nombreux logiciels propriétaires et libres (dont GnuPG etOpenSSH).

- Références
- http://www.0faute.com/ssl.htm
- http://lococheh.net/protocole.php
- http://fr.wikipedia.org/wiki/Fonction_de_hachage
- http://www.commentcamarche.net/crypto/signature.php3

- www.securite.org/db/crypto/**ssl**

- Support de cours. Université d'été.INPT, Rabat 2006