Rapport de projet – Cryptographie
Le protocole SSL

1. Sommaire

2. Introduction

Le besoin de sécurisation des échanges sur Internet n'est pas le même qu'il y a dix ans de cela. Aujourd'hui, il est monnaie courante pour un consommateur de faire ses courses sur internet, de faire toute sorte de transactions bancaires, ou d'échanger des données sensibles professionnelles sur le réseau. Le protocole SSL, apparu en 1995, a pour but d'assurer une certaine sécurité dans les échanges de données sensibles sur internet. Dans le contexte actuel précité, l'utilisation du protocole SSL est indispensable, et il est bon, en tant qu'utilisateur final, de comprendre les buts, le fonctionnement, les limites de ce protocole. C'est ce que ce document essaye d'expliquer de façon simple.

3. Présentation de SSL

SSL signifie Socket Secure Layer (littéralement Couche de Sécurité des Sockets, où les sockets sont les structures de communications sur la couche transport du modèle OSI). La version 3.1 du protocole a été renommée en TLS pour Transport Layer Security (Sécurité de la Couche Transport).

4. Historique

SSL a été développé par la société Netscape en 1994. Il a été publié en 1995, directement en version 2. Les vulnérabilités de la version 2 ont été corrigées par la version 3 publiée en 1996. Suite au rachat du brevet de Netscape par l'IETF en 2001, la version 3.1 a été renommée en TLS. Les versions 1.1 et 1.2 ont succédé respectivement en 2006 et 2008 à TLS version 1.

5. Applications de SSL

5.1. Contextes d'utilisation

A l'heure actuelle, SSL est principalement mis en place sur des serveurs web proposant des services de transactions commerciales (achat/vente en ligne), ou simplement

des services d'identification d'utilisateurs de comptes en lignes (comptes de messageries, forums).

Cette utilisation de SSL se nomme https, pour « http secure », car le protocole http est utilisé au dessus de SSL, c'est-à-dire que SSL est inséré entre le bloc de protocoles TCP/IP et la couche application en http.

Mais SSL peut être utilisé pour d'autres protocoles applicatifs, tel que telnet ou ftp.

5.2. Objectifs sécuritaires

Les objectifs de sécurité que veut atteindre le protocole SSL sont les suivants : confidentialité, intégrité, authentification. Ci-après une explication de ces trois termes.

5.2.1. Confidentialité

Assurer la confidentialité des données consiste à assurer que les données ne sont accessibles que par les personnes ayant un droit moral d'accès à ces données. Dans le contexte de la communication entre deux parties, la donnée en question est l'information envoyée d'une partie à l'autre, et la personne ayant le droit d'accéder aux données est le destinataire du message envoyée.

La confidentialité peut être compromise par un pirate tentant d'intercepter un message en transit sur le réseau et en extraire des informations utiles.

5.2.2. Intégrité

Assurer l'intégrité d'une donnée signifie assurer que la donnée ne sera 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, sans altération durant son transfert.

Un message pourrait être altéré durant son transfert soit par un problème lié au support, soit délibérément par une personne mal intentionnée.

Il est à noter que SSL est sensé reposer sur un protocole fiable comme TCP, proposant un contrôle d'intégrité. Cependant, c'est un contrôle d'intégrité qui ne peut rien contre la modification volontaire des données, qui peut faire passer des données falsifiées pour des données sans erreur d'un point de vue traitement du signal. Là où TCP se limite aux CRC, numéros de segments, et acquittements, SSL se sert des signatures et des hashages de signatures. Voir la partie Fonctionnement pour les détails.

5.2.3. 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 2 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 sa partenaire.

Lors de l'accès à un site web de vente en ligne, le consommateur voudrait, au moment du paiement, être sur qu'il donne son argent à la boutique en ligne dont il connait la réputation, et non à un escroc qui encaissera l'argent sans suite. C'est pour cela qu'il a besoin d'authentification : pour être sur que le site web est bien la fameuse boutique en question, d'après son nom. On a donc souvent besoin d'identification du serveur web, mais plus rarement celle du client. SSL permet l'authentification des deux parties.

5.2.4. Relations entre les trois concepts

Il est évident que ces trois concepts sont très liés. Une communication dont l'authentification est compromise est une communication où l'on risque de communiquer des informations à une personne qui n'est pas celle que l'on croit. Donc la confidentialité est compromise par définition, car cette personne malveillante n'a pas le droit de prendre connaissance de ces informations. De plus, l'intégrité est aussi compromise, car le véritable destinataire désiré n'aura pas reçu notre message, et à plus forte raison il n'aura pas reçu le message sans altération – dans le cas d'une compromission de l'authentification aboutissant à une configuration de l'homme du milieu, le message pourra être altéré par ce dernier, et on aura donc une compromission d'intégrité au sens usuel.

Nous avons donc que l'intégrité et la confidentialité dépendent de l'authentification, mais on peut ajouter que la confidentialité dépend aussi, dans une certaine mesure, de l'intégrité, comme on peut le constater dans des attaques par « copier-coller ».

6. Fonctionnement

SSL est un protocole de niveau Session. Plusieurs communications différentes peuvent simultanément avoir lieu sous la même session. Cela peut être utile lorsque deux parties communiquent dans plusieurs processus clients par exemple : les couts d'établissement de connexion multiples sont économisés.

6.1.Les deux couches de SSL

SSL est composé de 2 couches logiques superposées dans la pile OSI : une couche basse appelée « *Record Layer* » ou « *Record Protocol* » et une couche haute appelée « *Handshake Protocol* ».

La couche basse est en fait la couche qui fournit les services de bases qui seront nécessaires tout au long d'une communication. La couche haute est constituée de protocoles de gestions de la connexion, qui sont encapsulés dans la couche basse.

6.2.Couche basse : Record Layer

La couche basse de SSL le « *Record Layer* » ou « *Record Protocol* » est la couche encapsulant tous les types de données SSL : données brutes et données de gestion de la connexion.

Une trame de la couche *Record Layer* se compose des champs suivants :

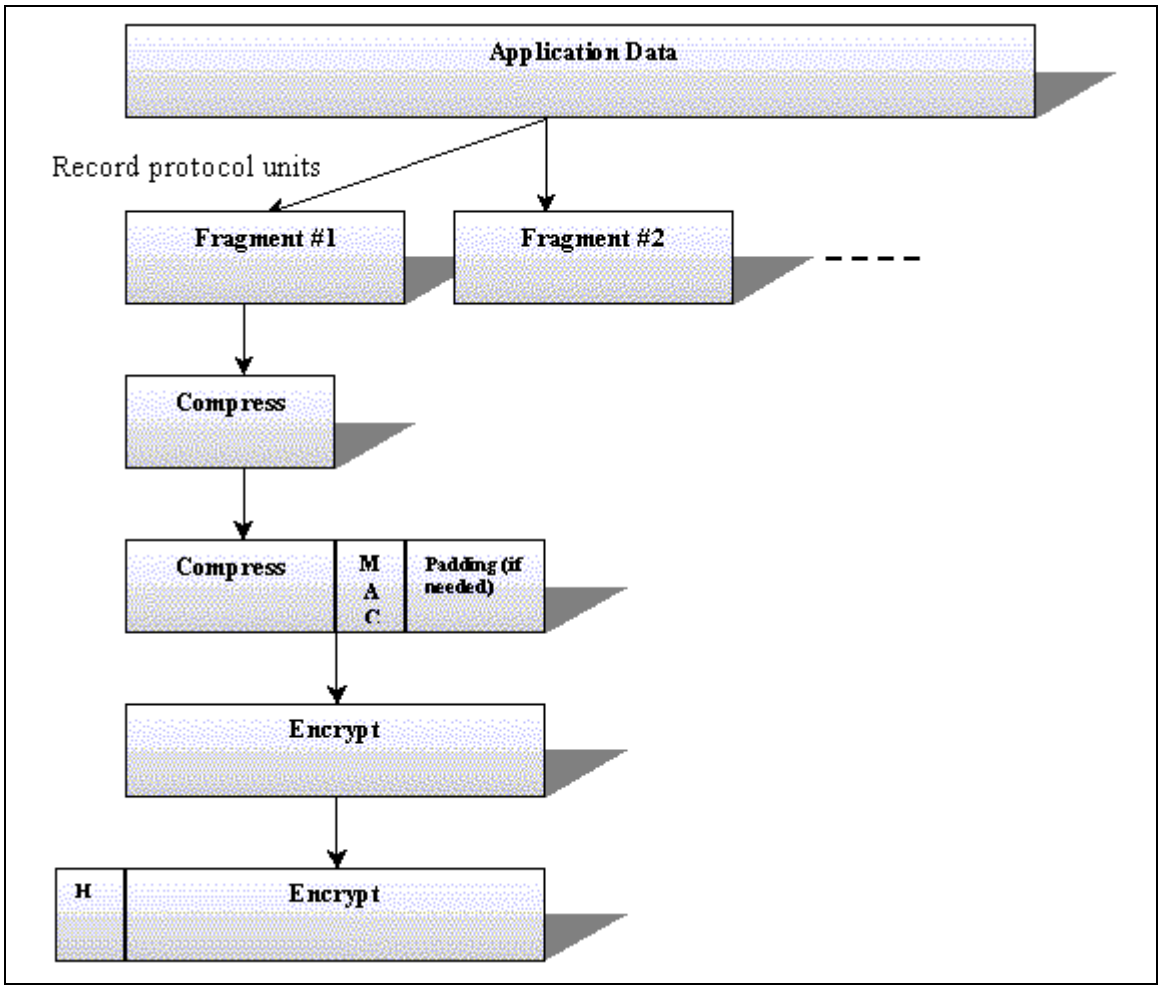
Type	Minor version	Major version	Record Length (< 2^{14}o)	Record Data
8 bits	8bits	8 bits	16 bits	< 2^{14} octets

Les champs trouvés ici sont classiques. On trouve le type du protocole encapsulé, (se référant au protocole de la couche haute de SSL, voir Couche haute : Handshake Protocol), les numéros de version de SSL utilisé, la taille des données et les données utiles après traitement.

Voici le traitement que subit un segment arrivant de la couche haute du SSL :

- Fragmentation
- Compression (Optionelle)
- Signature (MAC - Message Authentication Code)
- Chiffrement

Schéma montrant la succession de ces actions (1) :



Ces traitements constituent les services que fournit le *Record Layer* à la couche supérieure de SSL.

6.2.1. Fragmentation

Lorsqu'un segment de données provient de la couche haute, il est fragmenté en segments de taille inférieure à 2^{14}o (taille dépendant de l'implantation). Les séparations entre segments de la couche haute ne sont pas gardés (deux segments de couche haute séparés consécutifs peuvent être contigus dans un segment du *Record Layer*).

6.2.2. Compression

La compression est optionnelle. En fait, elle est toujours active, mais sans effet par défaut. Elle est inutile lorsque le taux de transfert est élevé, où elle crée un surcout de traitement.

6.2.3. MAC - Message Authentication Code

Le MAC est une signature de chaque segment de donnée qui passe dans la connexion. Le MAC sert à prouver qu'un message n'a pas été corrompu durant le transfert (il assure donc l'intégrité du message). Il est calculé de la manière suivante :

```
hash(MAC_write_secret + pad_2
                + hash (MAC_write_secret + pad_1 + seq_num + length + content));
```

Où :

- MAC_write_secret est une phrase secrète decidée au moment de la connexion
- pad_1 est le caractère 0x36 répété 48 fois pour MD5 ou 40 fois pour SHA
- pad_2 est le caractère 0x5c répété autant de fois que pad_1
- seq_num est le numéro de séquence du message
- hash est l'algorithme de hashage decidé au moment de la connexion

6.2.4. Chiffrement

Le chiffrement peut être de deux grands types : chiffrement par bloc, tel que AES, ou chiffrement de flux, tel que RS4. Le chiffrement par bloc est une méthode de chiffrement qui prend un bloc et qui le transforme en un bloc chiffré. Le chiffrement par flux est une autre méthode qui chiffre un flux au fil de l'eau, générant un flux chiffré en temps réel.

L'algorithme de chiffrement étant défini au début de la connexion, le chiffrement consiste à chiffrer le message accompagné du MAC, pour assurer la confidentialité du message.

6.2.5. Décapsulation

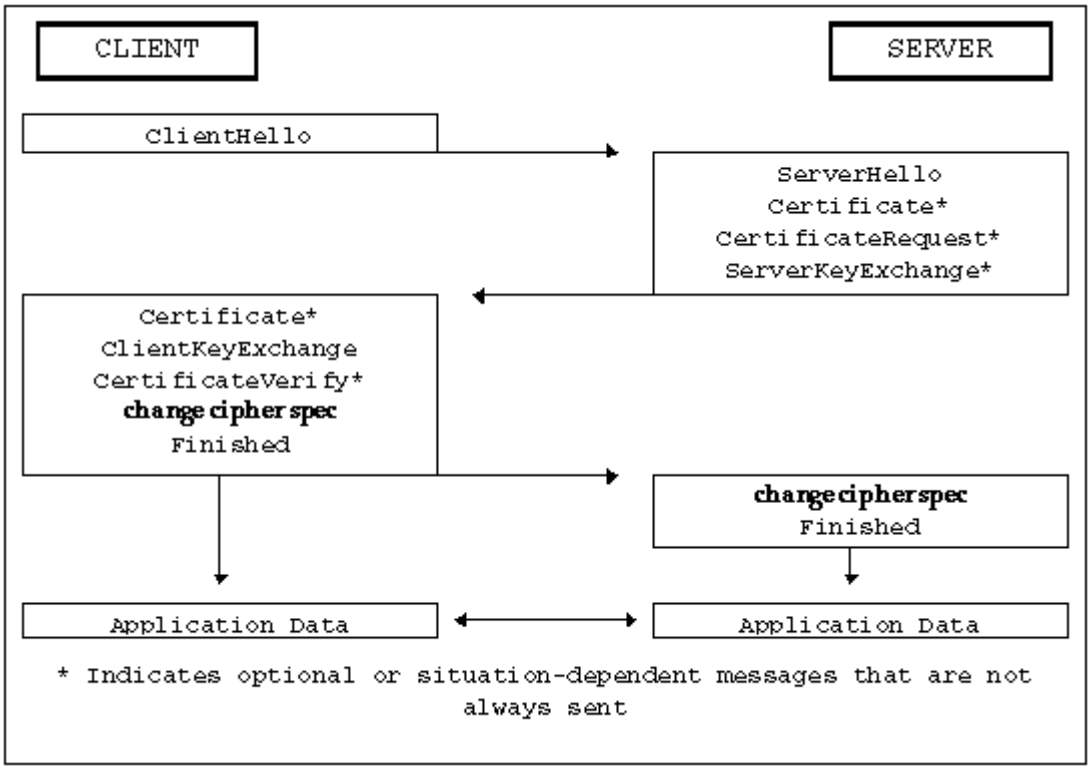
Bien sur, lors de la remontée des données dans la pile des protocoles, les opérations sont menées inversement : décryptage, contrôle d'intégrité, décompression, concaténation des messages, puis envoi à la couche haute SSL.

6.3.Couche haute : Handshake Protocol

Le *Handshake Protocol* sert à gérer une connexion SSL, c'est-à-dire décider des paramètres de sécurités d'une communication, et notamment assurer le point sensible qui est l'authentification initiale. A l'instar d'une connexion TCP comprenant de segments gérant

la fiabilité, le *Handshake Protocol* propose des segments dédiés à l'établissement d'une connexion et à la gestion du flux, mais tout ceci d'un point de vue sécurité.

Schéma représentant les échanges de messages de type *Handshake* pour une connexion SSL :



La dernière ligne représente le début du transfert de données couche application, c'est-à-dire les données utiles, ne faisant plus partie du processus de poignée de main.

Toutes les données qui sont apparemment implicites dans la couche inférieures, c'est-à-dire les informations dont on a dit qu'elles ont été établies au début de la connexion, sont effectivement établies par le *Handshake,* et mémorisées en ce qui est appelé l'état de la session (*session state*) et les états des connexions (*connection states*).

6.3.1. Etat de la session (*session state*)

La structure de données *session state* est composée des champs suivants :

- session id – Octet arbitraire identifiant une session.
- peer certificate – Un certificat X509.v3 appartenant au poste.
- compression method – La méthode de compression.

- cipher spec – Description des algorithmes de chiffrement et des paramètres pour le MAC.
- master secret – Phrase secrète de 48 octets partagée entre client et serveur.
- is resumable – Drapeau indiquant la réutilisabilité d'une session pour plusieurs connexions.

6.3.2. Etat de la connexion (*connection state*)

La structure de données *connection state* est composée des champs suivants :

- server and client random – Suite d'octets aléatoire choisie par le client et le serveur à chaque connexion.
- server write MAC secret – Le secret utilisée pour calculer le MAC sur des données écrites par le serveur.
- client write MAC secret – Idem pour les données client.
- server write key – La clé de chiffrement utilisée pour les données cryptées par le serveur et décryptées par le client.
- client write key – Idem, inversement.
- initialization vectors – Données d'initialisation jouant le rôle de grain initial pour le chiffrement par bloc.
- sequence numbers – Chaque partie maintient de son coté les numéros de séquence des messages envoyés et reçus pour chaque connexion.

6.3.3. Messages possibles

Ci-dessous une description de la communication de poignée de main représentée dans le schéma précédent.

- ClientHello : Initialisation de l'état de la session par le client.
- ServerHello : Réponse au client_hello.
- ServerCertificate : Le serveur s'authentifie avec son certificat x509.
- CertificateRequest : Optionnellement, le serveur demande au client de s'authentifier.

- ServerKeyExchange: Si le serveur n'a pas de certificate ou un certificat ne contenant que la signature de sa clé publique, il définit ici quel algorithme d'échange de clés sera utilisé. Pour SSL 3, les algorithmes disponibles sont RSA, Diffie-Hellman fixe , Diffie-Hellman éphémère, Diffie-Hellman anonyme, et Fortezza).

- ClientCertificate : Le client s'authentifie en réponse au message CertificateRequest. Si le client n'a pas de certificat, il envoie un message d'erreur (voir Protocole d'Alerte).

- ClientKeyExchange : En réponse au message server_key_exchange, le client envoie les paramètres de l'algorithme d'authentification choisi. Par exemple, pour RSA, il envoie son *pre-master secret* chiffré avec la clé publique du serveur.

- CertificateVerify: This message is sent in order to provide definite verification of a client certificate. This message is sent following all client certificates except those including Fixed Diffie-Hellman parameters.

A ce stade, client et serveur calculent leur secret maître comme suit :

```
master_secret = MD5 ( pre_master_secret + SHA ( 'A' +
pre_master_secret + ClientHello.random +
ServerHello.random ) ) + MD5 ( pre_master_secret + SHA (
'BB' + pre_master_secret + ClientHello.random +
ServerHello.random ) ) + MD5 ( pre_master_secret + SHA (
'CCC' + pre_master_secret + ClientHello.random +
ServerHello.random ) ) ;
```

Le secret maître est utilisé comme grain pour la génération d'une clé symétrique pour la communication.

- ChangeCipherSpec : Le client et le serveur indiquent qu'ils ont calculé la clé symétrique et que tous les messages suivants seront chiffré avec.

- Finished: Ce message suit le message ChangeCipherSpec pour indiquer que les paramètres de cryptographie ont été bien pris en compte et il est chiffré avec la clé calculée juste avant.

A ce moment a lieu une synchronisation avant le début des échanges de données utiles.

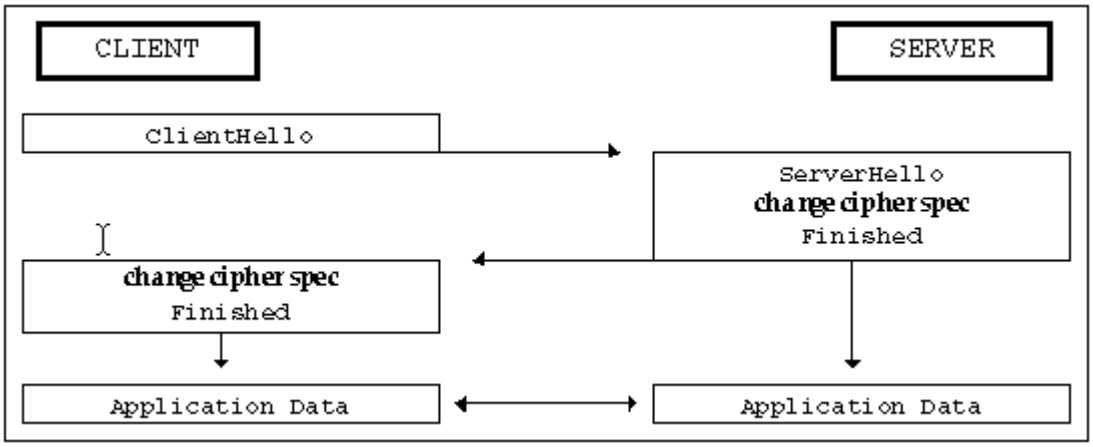
6.4.Couche Haute : Autres protocoles

La couche haute est appelée Handshake Protocol car elle est la plus importante en terme de nombre de messages utilisés pour gérer la connexion.

Les deux autres protocoles disponibles dans cette couche sont le protocole de changement des spécifications de chiffrement et le protocole d'alerte.

6.4.1. Changement des Spécifications de Chiffrement

Ce protocole sert à modifier la stratégie de chiffrement utilisée dans une session. Son fonctionnement est toujours le même que lorsqu'il apparait dans le processus de poignée de main :



Ainsi, après tout changement dans la stratégie de chiffrement, les deux parties se resynchronisent avec un message « Finished » (Terminé).

6.4.2. Protocole d'Alerte

Le protocole d'Alerte sert à signaler une erreur dans la communication, comme une erreur dans le calcul du MAC, ou une impossibilité de s'authentifier pour le client.

Les erreurs sont de deux types : fatales et non fatales. Les erreurs fatales mettent fin à la communication, tandis que les non fatales marquent seulement la session comme non utilisable pour les prochaines connexions. Le protocole est donc très simple : un octet pour l'erreur et un octet pour sa gravité.

7. Vulnérabilités Connues

SSL dans sa version 2 possédait des vulnérabilités qui ont fini par en faire un protocole obsolète. Il a été avantageusement remplacé par le protocole en version 3, qui règle la plupart de ces vulnérabilités.

SSL en version 3, d'après l'article de D.Wagner et B.Schneier, est relativement fiable, et une analyse de nombreux points de sécurités n'a pas révélé de faille importante dans SSL 3, bien que certains points puissent être améliorés.

Malgré la grande fiabilité de SSL 3, certaines vulnérabilités sont exploitables par un individu mal intentionné pour compromettre une ou plusieurs des dimensions de sécurité prises en charge par SSL. La cause principale qui permet d'exploiter ces failles est due, dans la majorité des cas, à une mauvaise configuration d'un utilisateur, de sa crédulité, ou de son ignorance de certains mécanismes de sécurité.

Ci-après une description d'attaques classiques en matière de sécurité, et une mise en correspondance rapide avec SSL.

Replay

La *Replay Attack* consiste à intercepter une communication et réenvoyer à l'identique un message déjà envoyé dans cette communication. Ceci peut servir à envoyer des informations d'authentifications copiées sur celles d'une communication passée.

SSL pare cette attaque grace au MAC qui contient le numéro de séquence du message et d'autres paramètres spécifiques à la connexion et au poste. Donc un message réenvoyé sera détecté comme n'étant pas à sa place. Soit mauvais numéro de séquence, soit mauvais numéro de connexion, etc. De plus, le MAC ne peut être modifié car il est hashé. Le hashage est sensé être à sens unique.

Analyse de Traffic

Cette attaque consiste à écouter une communication et essayer de deviner des information à partir de la longueur des champs protocolaires.

SSL ne propose pas de solution contre ces attaques car elles sont considérées peu dangereuses. Cependant dans de rares cas (cas des chiffrements de flux), il est possible de récupérer la longueur du champ GET de http (l'URL). D'après l'article de D.Wagner et B.Schneier, ce cas, bien que peu dangereux, pourrait faire l'objet d'une amélioration par un bourrage aléatoire.

Copier-Coller – Cut-And-Paste

L'attaque par copier-coller consiste à intercepter un message dans une communication SSL, puis de le rediffuser à un autre moment ou dans une autre session. Une variante plus subtile consiste à tenter d'insérer un bloc (couche basse) de message chiffré et l'introduire comme bloc d'un autre message qui est destiné à être restitué en clair. En effet,

certains champs d'une page web par exemple présentent un niveau faible de confidentialité et seront restitués en clair.

SSL pare ce danger en appliquant un MAC à chaque bloc. Le MAC dépend du poste, de la session et du numéro de message. Et l'attaquant ne peut pas générer un bon MAC, car le MAC dépend aussi de ma phrase secrète.

L'Homme du Milieu

Comme il a été vu auparavant, l'intégrité et la confidentialité sont tributaires de l'authentification. Cela veut dire qu'une compromission de cette dernière est la voie royale vers la compromission des deux autres.

L'attaque de l'Homme du milieu (ou MITM pour Man In The Middle) exploite ce phénomène en compromettant l'authenticité durant la phase d'authentification (*Handshake protocol*) par substitution à l'une des deux parties. En général, l'authentification du client n'est pas exigée par le serveur, donc coté serveur il n'y a pas de problème. Coté client, le pirate espère que le client sera assez ignorant pour accepter une connexion à un site dont la clé publique ne correspond pas à celle désignée par le jeton x509 associé à son nom de domaine chez l'Autorité de Certification.

Pour se substituer, d'un point de vue réseau, au serveur pour le client, l'attaquant corrompt le cache ARP et/ou DNS en diffusant en broadcast des réponses respectivement ARP et/ou DNS.

SSL implante l'authentification utilisant les jetons x509 permettant aux parties de se donner les moyens d'éviter cette attaque, mais ne peut probablement pas faire plus.

8. Conclusion

L'utilisation importante et croissante témoigne de l'efficacité de SSL. Ce protocole qui continue d'évoluer jusqu'à aujourd'hui par de petits ajouts n'est pas prêt de voir son utilisation diminuer.

Malheureusement, le facteur limitant de façon ultime la sécurité reste le facteur humain. Bien que SSL constitue un outil complet pour assurer une sécurité satisfaisante pour des échanges sensibles, il y a des efforts à faire pour sensibiliser les utilisateurs finaux aux mécanismes de cette sécurité, car ces derniers en ont de plus en plus besoin dans leur vie quotidienne. Car, sans surprise, les techniques actuelles de divulgation d'information et de vol d'argent sur le web reposent largement sur l'ingénierie sociale.

9. Bibliographie

(1) http://www2.rad.com/networks/2001/ssl/over.htm

Analysis of the SSL 3.0 protocol (David Wagner, Bruce Scheier), Avril 1997

SSL Man-in-the-Middle Attacks (Peter Burkholder), Février 2002 (v2.0)

SSL 3.0 Specification (Final Draft) (IETF) 1999.