Chapitre 4

Le protocole sécurisé SSL

Les trois systèmes de sécurisation SSL, SSH et IPSec présentés dans un chapitre précédent reposent toutes sur le même principe théorique : cryptage des données et transmission de la clé à distance. Nous allons détailler dans ce chapitre les principes théoriques sur lesquels ces solutions reposent puis plus en détail un de ces protocoles, à savoir SSL.

4.1 Les principes théoriques d'un protocole sécurisé

4.1.1 Le chiffrement pour assurer la confidentialité

Dans les protocoles réseau que nous avons vus jusqu'à maintenant, le texte est transmis en clair et peut donc être récupéré par une personne curieuse ou mal intentionnée (qu'il est devenu traditionnel d'appeler **attaquant** ; *attacker* en anglais). Comme nous l'avons vu, ce problème est connu depuis la plus haute Antiquité, époque à laquelle on a commencé à essayer de brouiller le message, par exemple en remplaçant une lettre par la troisième lettre qui suit dans l'alphabet.

Le premier problème à régler est donc celui de la **confidentialité** (*confidentiality* en anglais) : *les données brutes ne doivent pas pouvoir être lues par une personne autre que l'expéditeur et les destinataires.*

Ce problème est résolu, de façon relativement satisfaisante, par le chiffrement des données, dont nous avons déjà parlé.

4.1.2 Le problème de la transmission de la clé

Nous avons crypté un texte, en utilisant telle méthode et telle clé. Très bien. On peut maintenant transmettre le texte crypté à travers le réseau. Le destinataire a besoin de connaître la méthode de cryptage et la clé pour décrypter.

On considère en général que la méthode de cryptage n'est pas un renseignement sensible, de toute façon il n'en existe qu'un nombre restreint. Par conséquent les protocoles soit reposent sur une seule méthode de cryptage, soit l'un des champs de son en-tête précise en clair la méthode de cryptage choisie.

Il en est tout autrement de la clé. Si Alice et Bob[1] ont la possibilité de se rencontrer physiquement, ils peuvent s'échanger la clé à cette occasion. Ceci ne sera pas le cas cependant si vous faites des achats sur Internet. De toute façon on a intérêt à changer la clé de temps en temps et même souvent. On est donc confronter au **problème de transmission de la clé** (*the key management problem* en anglais).

Le problème a été résolu en 1976 à Stanford par Whitfield Diffie et Martin Hellman dans [D-F–76]. Ils y ont suggéré ce qui est maintenant appelé la **cryptographie à clé publique** (*public key cryptography* en anglais ; par opposition les anciennes méthodes s'appellent maintenant **cryptographie à clé secrète**). L'idée fondamentale est que les fonctions de cryptage et de décryptage utilisent des clés différentes. La clé de cryptage (la **clé publique**) peut être publiée mais la clé de décryptage (**clé privée**) est gardée secrète. Bien entendu les deux clés ont un lien (mathématique) mais il est très difficile (pour ne pas dire impossible en pratique) de déterminer la clé privée à partir de la clé privée.

La cryptographie à clé publique est aussi appelée **cryptographie assymétrique** et la cryptographie à clé secrète **cryptographie symétrique** pour des raisons évidentes.

4.1.3 Identification des extrémités et certification

Il est très important d'identifier l'expéditeur et le destinataire (*endpoint authentication* en anglais) ou tout au moins l'un des deux suivant le cas pour pouvoir contrer l'attaque dite du troisième homme. Avant d'envoyer son numéro de carte bancaire, on a intérêt à être sûr qu'on l'envoie au bon endroit.

[1] Il s'agit des noms devenus traditionnels, plus charmants que A et B.

Attaque du troisième homme.- En effet un type d'attaque possible lorsque les clés sont publiées électroniquement ou si elles sont échangées au début d'une communication est connu sous le nom d'**attaque du troisième homme** (*man-in-the middle attack* en anglais).

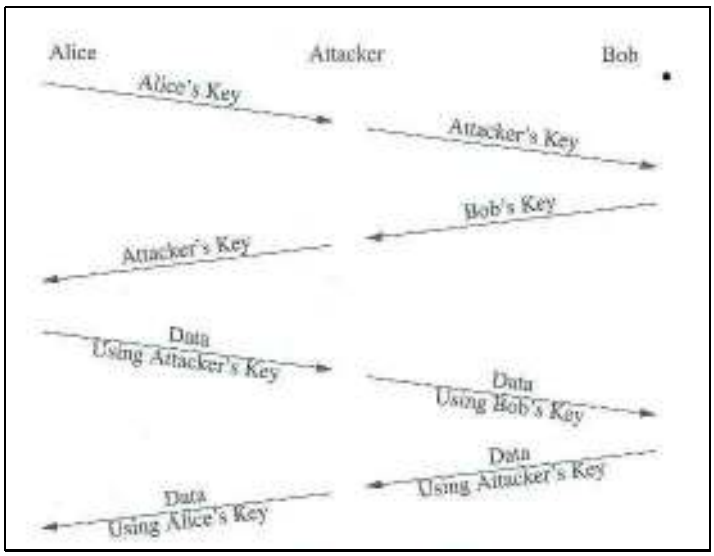

FIG. 4.1 – Attaque du troisième homme

L'attaquant se place entre Alice et Bob et intercepte leurs messages. Lorsqu'Alice envoie sa clé publique, il la garde pour lui et envoie sa propre clé à Bob à la place. Lorsque Bob répond en envoyant sa clé publique, l'attaquant la garde pour lui et envoie sa propre clé à Alice. Lorsque Alice envoie ensuite des données, l'attaquant peut les décrypter, les changer et les envoyer à Bob. Il en est de même lorsque Bob envoie des données (figure 4.1).

Certification.- La solution à ce problème est de faire appel à un tiers en qui on a toute confiance, appelé **autorité de certification** (**CA** pour l'anglais *Certificate Authority*). Celle-ci publie sous forme papier ou sur CD-ROM une liste de **certificats**, constitués sous sa forme la plus simple d'un nom et de sa clé publique.

4.1.4 Intégrité du message et fonction de hachage

Un autre problème concernant la sécurité est de s'assurer que le message que l'on reçoit est celui envoyé par l'expéditeur dans son intégralité, autrement dit qu'une partie de celui-ci n'a pas été altérée (volontairement).

Ceci est facile à vérifier dans le cas de la transmission d'un numéro de carte bancaire, qui comprend un nombre de chiffres fixe, mais cela n'est pas toujours le cas. La théorie montre que la résolution complète de ce problème n'est pas possible ; en pratique on utilise les *fonctions de hachage* conviennent.

Une **fonction de hachage** *(*hash function *en anglais) prend le message (de longueur arbitraire) comme entrée et renvoie un texte de longueur fixe, appelé le* **condensé du message** *(*message digest *en anglais).*

Bien entendu une telle fonction ne peut pas être injective. Le condensé du message doit être suffisamment caractéristique du message et on ne doit pas pouvoir reconstituer le message à

partir de son condensé. Elle doit également être le plus possible résistante aux collisions (*collision-resistance* en anglais), c'est-à-dire que deux messages différents ayant un sens doivent produire deux condensés différents.

L'étude des fonctions de hachage fait l'objet d'un autre cours. Retenons seulement ici que les deux méthodes les plus utilisées sont **MD5** (pour *Message Digest 5*), conçue en 1992, et **SHA-1** (pour *Secure Hash Algorithm 1*), conçue en 1994.

4.1.5 Un système simple d'envoi de message sécurisé

Bien entendu un condensé ne suffit pas puisqu'un attaquant peut lui-même utiliser la fonction de hachage. Nous allons donc montrer comment construire un protocole sécurisé. Commençons par le cas d'envoi de message (un e-mail par exemple). Seul l'expéditeur a besoin d'être identifié.

4.1.5.1 Envoi du message

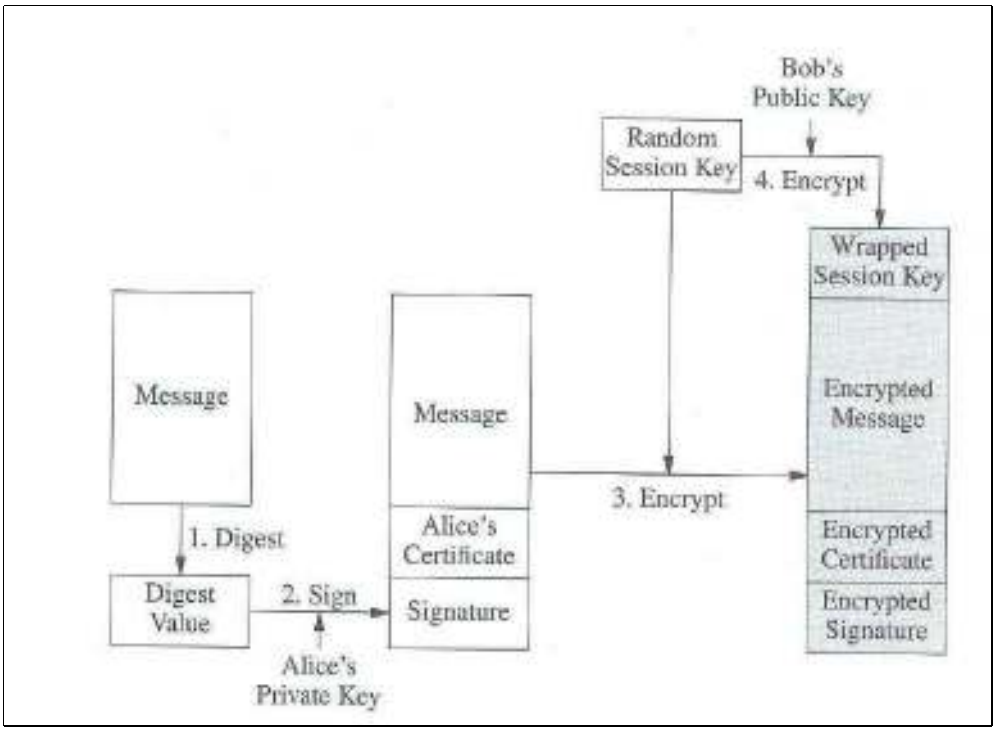

FIG. 4.2 – Envoi d'un message

Alice veut envoyer un message à Bob :

1. Elle calcule le condensé du message, de longueur connue.
2. Elle calcule la **signature numérique** du message en cryptant le condensé avec sa clé privée, signature qui aura également une longueur connue. Elle obtient alors un **message signé**, constitué du message initial, du certificat d'Alice (de longueur connue) et de la signature numérique du message.
3. Elle produit une **clé de session** de façon aléatoire et crypte le message signé avec cette clé.

4. Elle crypte la clé de session avec la clé publique de Bob et l'attache au message signé crypté, en tant que champ de longueur connue.

Elle envoie ce message final à Bob.

4.1.5.2 Réception du message

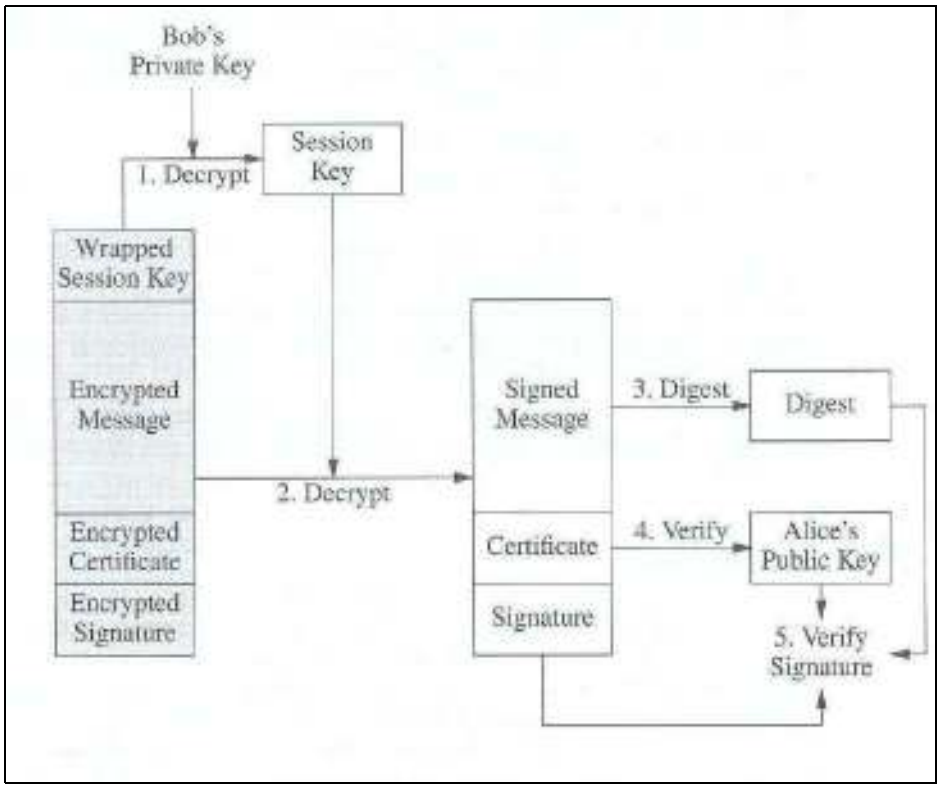

FIG. 4.3 – Réception d'un message

Lorsqu'il reçoit le message :

1. Bob utilise sa clé privée pour décrypter le champ clé de session crypté et obtenir la clé de session obtenue par Alice.
2. Il utilise alors cette clé de session pour décrypter le message signé et crypté et obtenir le message signé.
3. Il peut alors récupérer le message (champ de longueur connue) et en calculer lui-même le condensé.
4. Il vérifie le certificat d'Alice auprès du CA et en extrait la clé publique d'Alice.
5. Il utilise la clé publique d'Alice pour décrypter la signature numérique. Si le résultat obtenu est égal au condensé qu'il a calculé, il fera confiance au message.

4.1.6 Un canal de communication sécurisé

Le système que nous venons de décrire peut évidemment s'appliquer à un canal de communication (qui serait alors sécurisé) mais on perdrait alors beaucoup de temps à calculer à chaque fois une clé de session et les coûts en temps de calcul pour chaque paquet seraient prohibitifs. On préfére disposer d'un ensemble de clés secrètes disponibles pour la session dans son intégralité.

4.1.6.1 Protocole interactif

Dans un **protocole interactif**, on utilise un même ensemble de clés durant toute la session et les certificats sont envoyés (et non récupérés ou vérifiés auprès d'un tiers). Un tel protocole comprend quatre étapes :

1. Une phase de **négociation** (*handshake*, pour poignée de main, en anglais) permettent à Alice et Bob d'utiliser leurs certificats et leurs clés privées pour s'identifier et échanger un nombre appelé **secret partagé** (**MS** pour l'anglais *Master Secret*).
2. Alice et Bob utilisent ce secret commun MS pour **déterminer des clés** qui seront utilisées lors du transfert des données.
3. Les données à transmettre sont divisées en **enregistrements** (*record* en anglais), chacun d'eux étant protégé individuellement.
4. Des messages spéciaux protégés sont utilisés pour la **fermeture de la connexion**. L'existence de cette phase empêche un attaquant de forcer la fermeture et ainsi de tronquer le message.

4.1.6.2 Une négociation simple

Dans le cas où seul Bob a besoin de s'identifier, par exemple pour qu'Alice puisse lui envoyer son numéro de carte de crédit, la négociation peut s'effectuer de la manière suivante :

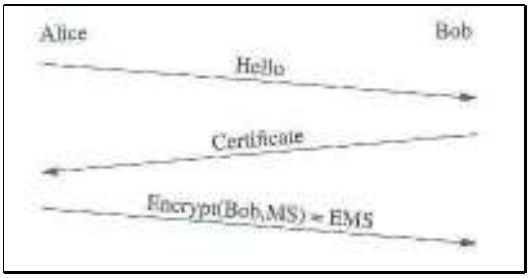

FIG. 4.4 – Identification dans un seul sens

1. Alice envoie un message à Bob pour lui dire qu'elle cherche à communiquer. On parle souvent d'étape d'« Hello ».
2. Bob répond en lui envoyant son certificat.
3. Alice envoie alors le secret à partager MS, bien entendu non en clair mais crypté avec la clé publique de Bob (**EMS** pour l'anglais *Encrypted Master Secret*).

Dans le cas où Bob et Alice doivent tous les deux s'identifier, la négociation peut s'effectuer de la manière suivante :

1. Alice envoie un « Hello » à Bob.

Ceci est un exemple, cliquez sur le lien de téléchargement pour obtenir le cours complet.