

Chapitre III

Inspection du trafic SSL

Plan de chapitre

- **Introduction**
- **L'attaque Man In The Middle**
- **Sniffer Trusted Man In The Middle**
- **Conclusion**

3.1. Introduction

Une vulnérabilité dans la conception des protocoles SSL et TLS a été découverte au début de Novembre 2009 [7]. Cette vulnérabilité permet au Man in the middle d'injecter les communications chiffrées suite d'une renégociation. Dans ce chapitre on va représenter l'attaque MITM en général, ayant l'accent sur des techniques et des outils pour appliquer cette attaque. Ensuite, nous allons expliquer notre méthode qui est l'**homme au milieu confiant (trusted man in the middle)**. Afin de se profiter de différentes techniques de l'attaque en question pour donner aux firewalls un module efficace capable d'analyser le trafic SSL.

3.2. L'attaque Man In The Middle

L'*« homme au milieu »* est un programme d'attaque et comme son nom l'indique il est situé entre le client et le serveur pour intercepter toutes les communications entre eux par l'établissement d'une connexion SSL doublée aux deux côtés. Ce programme intercepte les clés légitimes qui sont échangées dans les deux sens, et les remplace par leur propres clés, et se fait passer pour le serveur auprès du client.

Les informations chiffrées échangées au commencement de la négociation SSL sont en réalité chiffrées avec la clef publique ou privée du programme parasite, au lieu de celles du client, ou du serveur. Le programme parasite finit par établir un premier ensemble de clés de session à utiliser avec le véritable serveur, puis un second ensemble à utiliser avec le client. Ceci permet au programme parasite non seulement de lire toutes les données transmises entre le client et le serveur, mais également de modifier ces données sans que cela ne soit détecté. Il est donc très important pour le client de vérifier que le nom de domaine du certificat serveur correspond effectivement au nom de domaine du serveur avec lequel il tente de communiquer, en plus de vérifier sa validité.

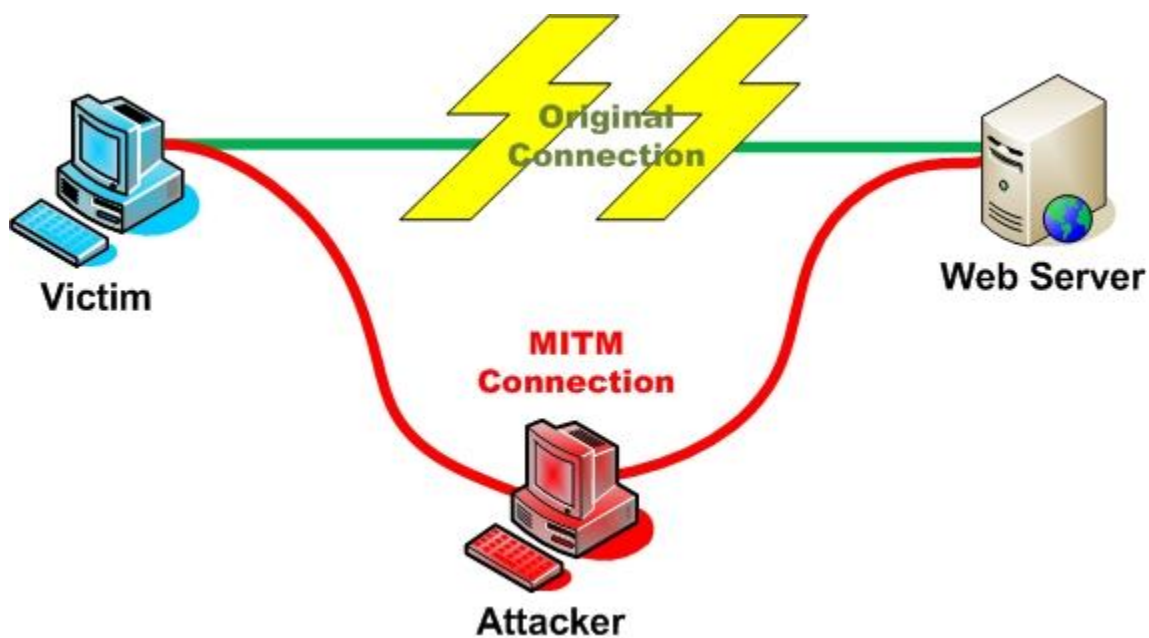


Figure 3.1architecture d'attaque man in the middle

3.2.1. Les techniques d'attaque Man in the Middle

Plusieurs techniques sont mises en œuvre pour réaliser cette attaque, les plus répandues sont les suivantes :

➤ **ARP Spoofing**

Le but de cette attaque est que l'attaquant veut intercepter les données circulant dans un réseau local (LAN) et les modifier, l'attaquant peut altérer le cache de ARP de postes connectés et prend finalement l'adresse IP du poste victime.

➤ **DNS Spoofing**

Dans ce cas-là L'attaquant commence l'interception de l'ID de chaque requête DNS, et puis faire répondre de cette requête avant que le serveur réel est répondu.

➤ **La redirection d'ICMP** (Internet Control Message Protocol)

Les attaques de redirection sont des autres formes d'attaques MITM. Quelques outils MITM permettent la création des messages de redirections ICMP pour annoncer qu'il y a une meilleur route pour faire passer le trafic réseau. Donc un attaquant peut utiliser ce type d'attaque pour obliger tous le trafic de réseau local de passer par son ordinateur. Une fois les données traversent l'ordinateur de l'attaquant ; il peut les capturer et les modifier selon le but d'attaque.

3.2.2 Les outils d'attaque MITM [7]

Ces outils sont développés pour simplifier l'attaque MITM ; ils peuvent contenir une ou plusieurs techniques d'attaque. Les outils les plus utilisés sont **Cain**, **ARPoison**, **Wireshark**, **The Middler**, **SSLStrip**, le package **Dsnif** de Linux qui contient des programmes similaires comme **arp spoof**, **dnsspoof**, **webmitm**, **dsniff**, et **webspy**.

3.2.2.1 Dsnif

Est une package pour **linux** qui contient des outils pour examiner le réseau et faire un test de pénétration. **Dsniff**, **filesnarf**, **mailsnarf**, **msgsnarf**, **urlsnarf**, et **webspy**, **webmitm** sont des moniteurs de réseau pour capturer les données (**passwords**, **e-mail**, **files**, etc.).

3.2.2.2 SSLStrip

Sslstrip est un outil développé par Moxie Marlinspike, vous pouvez le trouver sur son site web: thoughtcrime.org. Voici un schéma basique (figure 3.1) qui illustre son fonctionnement:

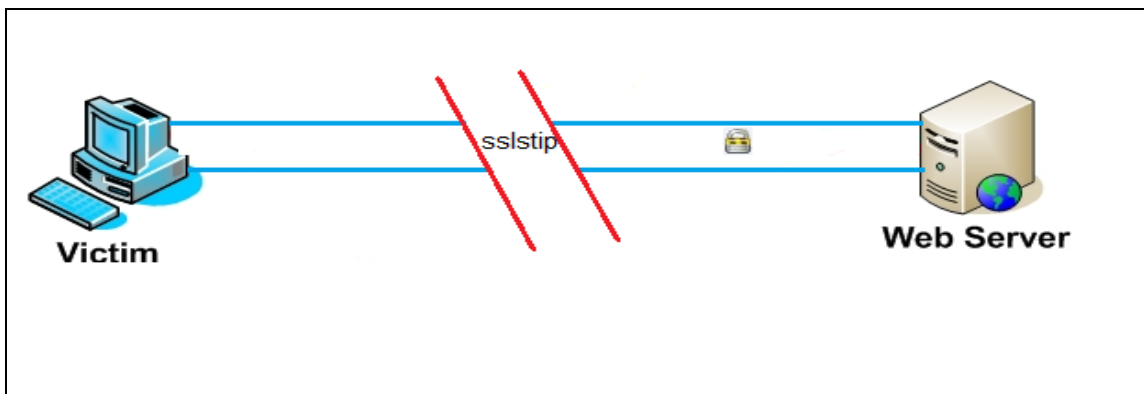


Figure 3.2 fonctionnement de SSLstrip

En effet, sur la plupart des sites proposant du contenu sécurisé en https, il faut dans un premier temps passer par une page http (accueil par exemple) avant d'être redirigé vers une page sécurisée. C'est le cas sur les sites bancaires, sur Google mail, etc... La grande majorité des sites web proposant du contenu sécurisé en https ont une page d'accueil en http.

Sslstrip n'attaque pas le https, il attaque le http. Il va transformer tous les liens https sur le poste victime en liens http.

Résultat:

- le serveur ne voit rien, pour lui la connexion est toujours encryptée ;
- le client ne voit aucun message d'alerte dans son navigateur ;
- l'attaquant peut sniffer toutes les données car elles transitent en clair ;

Grace à ce procédé, il est très difficile de différencier la véritable page https cryptée via ssl de la fausse page http renvoyée par sslstrip.

3.3. Sniffer trusted man in the middle (Sniffer TMITM)

Comme la majorité des entreprises souffrent du problème de trafic crypté malveillant, et au même temps le besoin d'utilisation des protocoles de cryptage par plusieurs sites pour sécuriser ses sessions avec leurs clients est omniprésent.

Donc pour résoudre ce problème en respectant les deux conditions (le trafic doit être crypté et au même temps la possibilité de l'analyser) ; on exploite l'avantage d'attaque **MITM** pour inspecter le trafic SSL. Par une technique nommée **Homme au milieu confiant** (trusted Man In The Middle) qui prend comme un principe l'ouverture de deux sessions séparées avant le commencement de chaque session SSL, car toutes les sessions entre un client et un serveur SSL sont commencées par l'échange des messages de **handshakes** comme on a expliqué dans le chapitre précédent. La plupart des sniffers utilisent la technique de « deux sessions séparées ».

3.3.1 Homme de milieu confiant avec deux sessions SSL/TLS séparées [05]

Dans cette approche, deux sessions SSL/TLS doivent être créées avant le commencement de la communication, une session SSL/TLS entre le client et notre sniffer et une autre entre le sniffer et le serveur, la **figure 3.3** ci-dessous décrit la communication entre le client, le serveur et le sniffer pendant le handshake.

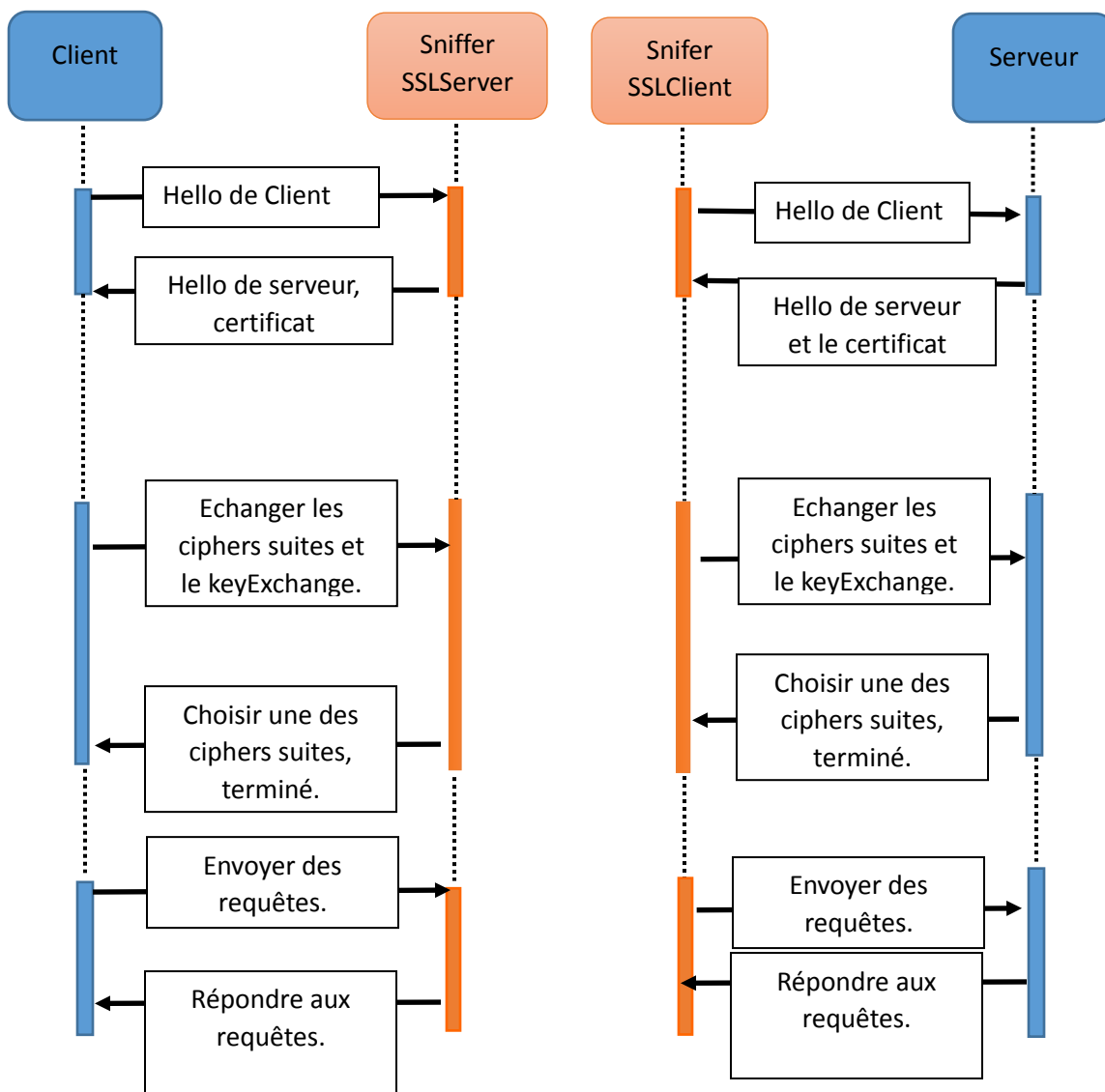


Figure 3.3 une sniffer avec deux sessions SSL séparés.

Comme le montre ce diagramme ; le pare-feu crée deux sessions indépendantes et stocke leurs données et exécute toutes les opérations de chiffrements pour les deux coté. Tous ces traitements augmentent la consommation des ressources (la mémoire et le débit...).

Pour cela une autre méthode moins couteuse a été inventée : « **homme au milieu confiant avec la modification de Handshake** ».

3.3.2 Homme au milieu confiant avec la modification de Handshake

Cette méthode fonctionne sur une seule session, notre sniffer intervient lorsque le client envoie le message de handshake, le sniffer intercepte les communications, modifie le handshake et l'envoie au serveur et vice versa (figure **3.4**) :

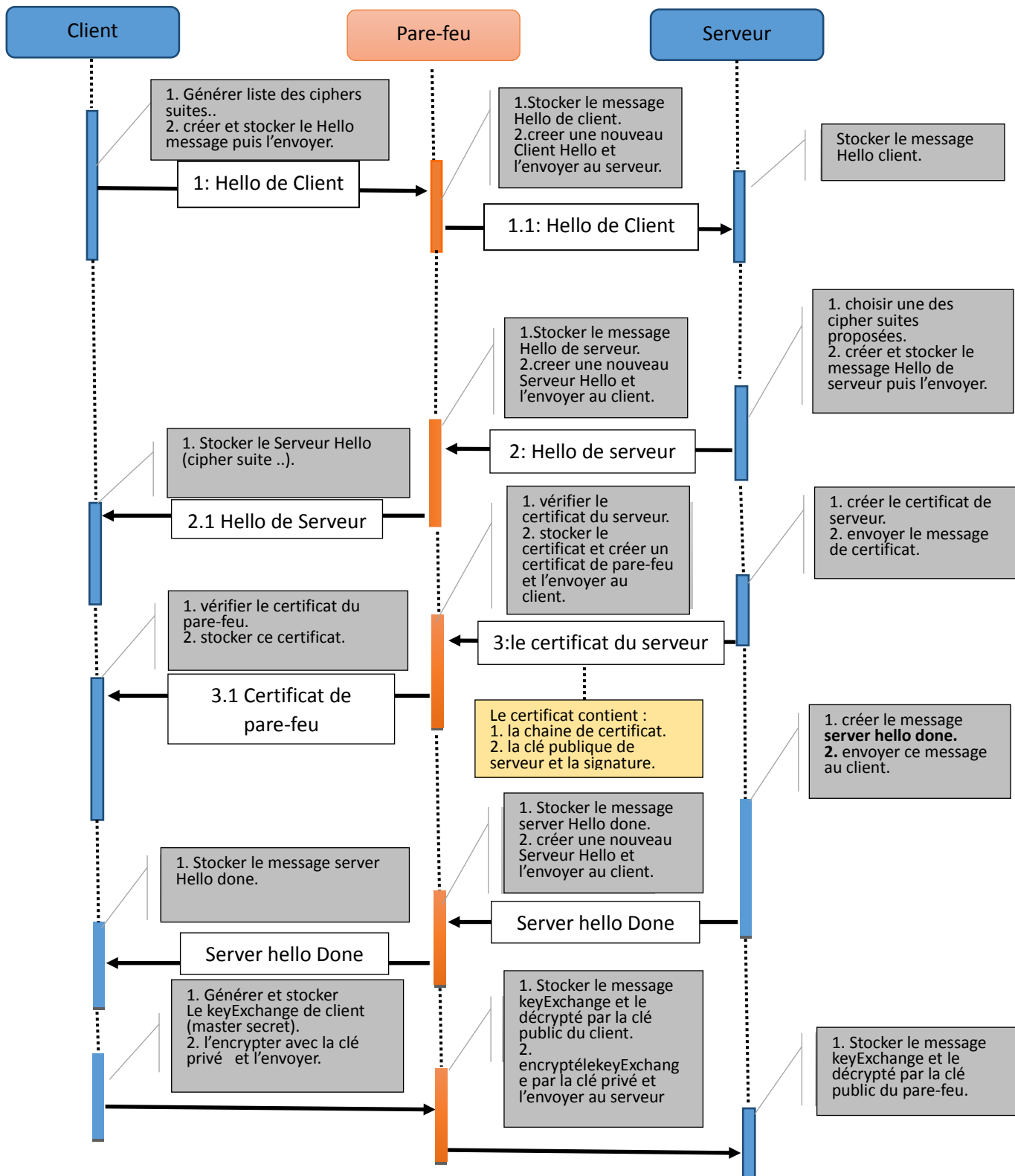


Figure 3.4 une sniffer MITM avec la modification de Handshake.

Maintenant, on va expliquer précisément le déroulement de cette technique, nous aurons vue comment le handshake va exécutera, dans notre exemple le pare-feu fait l'acte de MITM.

Les communications entre le client, le serveur, et le pare-feu pendant le handshake sont décrit ci-dessous :

1. Premièrement, le client génère la liste des ciphers suites supportés, génère la liste des méthodes de compressions, crée et stocke le message **Hello client** et l'envoyer au serveur.
2. le pare-feu intercepte la communication. Il stocke le message Hello de client, créer et stocker un autre message **Hello Client** et l'envoie au serveur.
3. le Serveur stocke le message **Hello de client**, puis choisi le plus puissant cipher suite supporté, crée et stocke le message **Hello du serveur et l'envoie au client**.
4. le pare-feu fait les mêmes opérations avec ce message.
5. le client stocke les informations de message **Hello de serveur**.
6. Le serveur crée son certificat et le transmet au client.
7. Le pare-feu intercepte et stocke le message du certificat de serveur (avec la clé public), et envoie son certificat au client (avec sa clé publique).
8. Le client vérifie le certificat du pare-feu puis stocke ce certificat.
9. le serveur crée un message de **serverHelloDone** et le transmet au client pour l'informer que leur rôle (serveur) est terminé.
10. le pare-feu intercepte la communication, et crée un message **serverHelloDone** et l'envoie au client.
11. le client génère le **ClientkeyExchange** (la clé de session) qui contient le **Master Secret** encrypté par la clé privée du pare-feu et le transmet au serveur.

12. le pare-feu intercepte le **ClientKeyExchange**le décrypte par la clé publique du client.
Puis l'encrypte par sa clé privée, puis le transmet au serveur.

13. le serveur reçoit le message de **ClientKeyExchange**puis le décrypte par sa clé publique du pare-feu. Après les messages seront chiffrés et déchiffrés avec la clé de session.

Après cette étape le pare-feu aura la clé de session. Donc il peut facilement intercepter tous les messages transmis.

La différence entre les deux approches est que cette dernier utilise une seule clé de session, par contre l'approche de deux sessions séparées utilise deux clés de session séparées et donc augmente le coût des opérations de cryptages.

3.4. Conclusion

Dans ce chapitre on a présenté l'attaque **Homme au milieu** (man in the middle) ; leurs techniques, leurs outils... et comment on a l'exploité dans l'inspection du trafic SSL par une technique nommée **Homme au milieu confiant** avec la modification des messages de **handshake**, qui aide le pare-feu pour intercepter tous le trafic crypté.

Dans le chapitre suivant on va présenter notre module d'inspection et les outils utilisés pour l'implémenter.