1. 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.

2. 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.

3. 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.

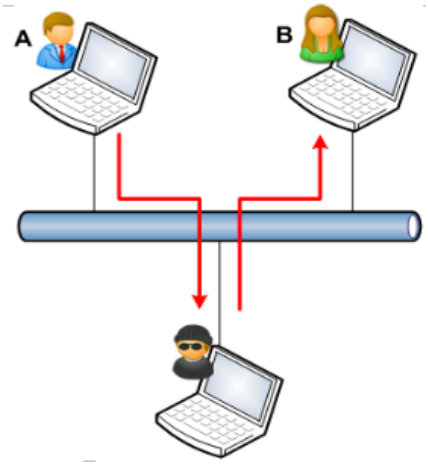

Figure 1 : Exemple d'une attaque MITM

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.

Quelques détails supplémentaires concernant l'attaque ARP spoofing :
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.

4. Comment fonctionne la mise en oeuvre 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.

Client SSH		Serveur SSH
1. Demande de connexion (Port 22/TCP).	→	
	←	2. Le serveur envoie les versions des algorithmes acceptés.
3. Choix parmi les algorithmes de celui qui convient le mieux.	→	
4. Échange de la clé de session (Diffie-Hellman) et authentification du serveur par l'envoi de l'empreinte de sa clé publique.	↔	4. Échange de la clé de session (Diffie-Hellman) et authentification du serveur par l'envoi de l'empreinte de sa clé publique.

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.

Alice choisit une couleur secrète (rouge) qui représente sa **clé privée**.

Alice mélange le jaune au rouge et obtient une couleur **publique** orange qu'elle envoie à Bob.

Alice reçoit sa couleur publique (bleu) de Bob.

Alice mélange la couleur reçue avec sa couleur secrète et obtient une couleur brune identique à celle obtenue par Bob, couleur qui représente la **clé secrète partagée**.

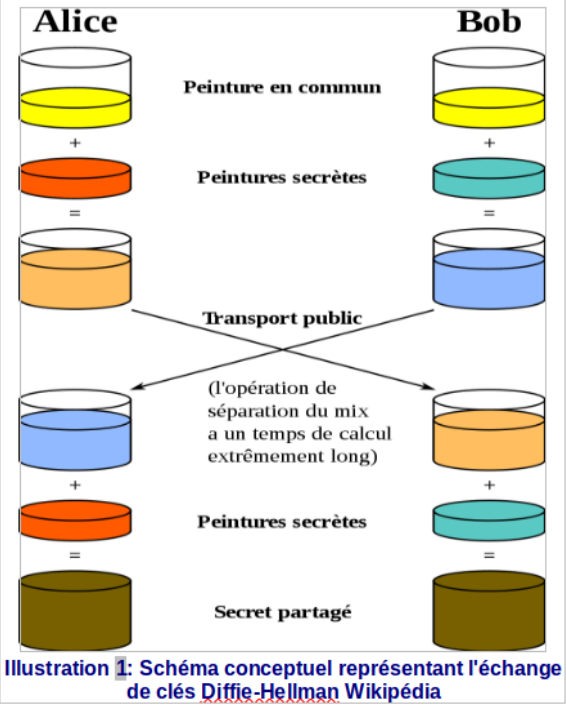

Illustration 1: Schéma conceptuel représentant l'échange de clés Diffie-Hellman Wikipédia

Bob choisit une couleur secrète (vert) qui représente sa **clé privée**.

Bob mélange le jaune au vert et obtient ainsi une couleur **publique** Bleu qu'il envoie à Alice.

Bob reçoit la couleur publique (orange) d'Alice.

Bob mélange la couleur reçue avec sa couleur secrète et obtient une couleur brune identique à celle obtenue par Alice, couleur qui représente la **clé secrète partagée**.