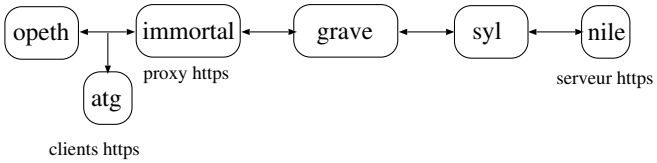
Université de Bordeaux — année 2024-2025

Master CSI/Master Info. — Sécurité des réseaux

TD - Interception HTTPS

Le but de ce TP est de mettre en place un proxy `HTTPS` afin d'écouter en clair le trafic `HTTPS` entre un client et un serveur. L'objectif du TP est de mettre en place un proxy `HTTPS` au niveau de la machine `immortal`. Ceci reviendra à implémenter un mécanisme similaire à `mitmproxy` dont la description est disponible à l'URL suivante : https://docs.mitmproxy.org/stable/. Ce type de proxy peut être utilisé par l'administrateur au sein d'un réseau informatique pour avoir accès au contenu du trafic `HTTPS`. Plus d'informations à ce sujet sont fournies dans les documents officiels suivants : https://cyber.gouv.fr/sites/default/files/2016/09/guide_tls_v1.1.pdf et https://cyber.gouv.fr/sites/default/files/IMG/pdf/NP_TLS_NoteTech.pdf.



La topologie réseau correspondante peut être obtenue en lançant le script de démarrage `/net/stockage/aguermou/SR/TP/9/qemunet.sh` en lui fournissant la description de la topologie réseau à l'aide de l'option `-t` ainsi que l'archive contenant la configuration initiale des machines à l'aide de l'option `-a`. Ceci revient à lancer les commandes suivantes :

```
cd /net/stockage/aguermou/SR/TP/9/; ./qemunet.sh -x -t topology -a archive_tp9.tgz
```

`nile` héberge un serveur `HTTPS` (accessible en consultant l'url `https://nile.metal.fr`) alors que `opeth` et `atg` vont jouer le rôle de client : Le certificat d'autorité de certification qui a été utilisé pour générer le couple certificat/clé privée de nile est connu de ces deux machines (grave étant la machine qui a été utilisée pour la génération des certificats). D'autre part, `immortal` connaît aussi ce même certificat. L'implémentation du proxy se fera en python3.

1. À partir des fichiers `client.py` et `server.py` qui implémentent une ouverture de connexion TCP à l'aide de socket de python, implémetez une version ssl de ce même échange. Pour ce faire, vous pouvez vous référer à la documentation disponible à l'URL suivante :

   https://docs.python.org/3/library/ssl.html#module-ssl

   Vous pourrez utiliser les fonction python `ssl.create_default_context` ainsi que `wrap_socket`. Un couple de certificat/clé privée (ainsi que le certificat d'autorité de certification correspondant) pour nile sont disponibles sur immortal dans le dossier `/root/certs`.
2. Il vous est demandé maintenant d'implémenter le proxy qui va permettre au niveau d'immortal d'écouter le trafic `HTTPS`. Un exemple de code python3 implémentant un proxy `HTTP` vous est fourni (`proxy.py`). Vous devrez mettre en place sur `immortal` :
   - un mécanisme d'interception du trafic `HTTPS` passant par immortal à direction du serveur hébergé par nile. Immortal jouera le rôle de client vis à vis de nile (le serveur `HTTPS`) et de serveur vis d'opeth (le client légitime). Elle pourra alors, sous réserve que tout est correctement implémenté, avoir accès au trafic `HTTPS` en clair. Ceci se fera à l'aide d'`iptables`.
   - générer un faux certificat pour nile sur immortal en créant une fausse autorité de certification. Attention pour que les certificats soient utilisables, il faut absolument respecter les contraintes suivantes :

    - Pour le certificat de l'autorité de certification seules les options qui permettent de signer d'autres certificats et des CRLs doivent être activées.
    - Pour le certificat de nile, il faut que le *Common Name* soit le nom DNS de nile (`nile.metal.fr` en l'occurrence). D'autre part, il faudra activer l'extension `TLS Web Server`.
  - Pour que le client (opeth ou atg) accepte le certificat frauduleux que vous avez généré il faut ajouter ce dernier au certificats de confiance au niveau des clients. Pour ce faire, il faut :
    - Renomer le certificat de l'autorité de certification pour qu'il soit au format `crt`.
    - copier le fichier `crt` sur les clients en le positionnant dans un sous dossier que vous aurez préalablement créé dans le dossier `/usr/local/share/ca-certificates`.
    - Mettre à jour, sur les clients, le coffre à certificats à l'aide de la commande `dpkg-reconfigure ca-certificates`.
3. Une amélioration du proxy `HTTPS` que vous avez implémenter à l'étape précédente serait de générer dynamiquement le faux certificat pour le serveur. Il vous est donc demandé d'implémenter ce mécanisme de génération dynamique. Cette étape est un bonus !