P. Lafourcade, M. More, M. Puys

Securité Web
2026/2027

TP2 Authentification

Avant de commencer. Un peu de préparation sera nécessaire avant de commencer.

1. **Extraction et Installation**: L'ensemble du TP sera dans VDN. Connectez-vous sur **vdn** sur la machine `debian-1` :

   (`Network > Secure > debian-1`), puis téléchargez l'archive suivante :

   ```
   wget https://sancy.iut.uca.fr/~lafourcade/SECU-3A/tp2.tar.gz
   ```

   Vous allez l'extraire avec `tar -zxvf tp2.tar.gz`, puis aller dans le dossier `websec25-tp2` et lancer les commandes suivantes :

   ```
   apt update
   apt install python3-requests python3-pycryptodome
   ```

2. **Lancement des serveurs**: Dans un *autre terminal*, lancer les serveurs avec la commande suivante, en prenant bien soin de laisser les serveurs tourner :

   ```
   docker load -i websec23-tp2.tar.xz
   docker run --rm -p 5001:5001 -p 5002:5002 -p 5003:5003 websec23-tp2
   ```

Exercise 1 (Sécurité SSH (5 points))
Clonez le dépôt https://github.com/jtesta/ssh-audit

1. Dans VDN, scannez votre serveur SSH local 127.0.0.1 avec ssh-audit. Décrivez les résultats.
2. Quelle version du serveur SSH tourne sur la machine ?
3. Modifiez le fichier `/etc/ssh/sshd_config` de votre machine VDN de façon à corriger certaines des faiblesses détectées par ssh-audit. Les mots clefs à utiliser sont: `Ciphers`, `HostKeyAlgorithms`, `MACs` and `KexAlgorithms`.

Il faut utiliser `systemctl restart sshd` pour relancer le service à chaque modification.

> **Attention :** Vérifiez si votre SSH est toujours fonctionnel après l'exercice. En cas de problème, vous pouvez réinitialiser la configuration SSH par défaut en supprimant l'image VDN (`vdn-clean`).

Exercise 2 (HTTPS en gérant la cryptographie avec openssl (10 points))
Exercice à réaliser dans **VDN**.

Objectif de l'exercice : Ajouter un certificat racine dans votre navigateur et placer une autorité intermédiaire dans votre site pour que l'accès à vos pages web soit considéré comme sûr.

1. (1 point) Générer avec `openssl` une clef RSA de 4096-bits qui servira de root CA (Certificate Authority). Choisir comme nom de fichier `cert/rootca.key` .

2. (1 point) Créer un certificat X.509 auto-signé pour root CA, en indiquant comme CN (Common Name) `www.rootdom.com`. Le résultat sera placé dans le fichier `cert/rootca.crt`. Pour cela, le début de la commande est :

   `openssl req -new -x509 -key cert/rootca.key` ... (à terminer)

3. (3 points) Créer un CA intermédiaire. Pour cela :

   - Commencer par créer une clef RSA de 4096-bits qui sera stockée dans le fichier :

     `cert/inter.key`

   - Générer un certificat pour ce CA intermédiaire dans le fichier `cert/inter.csr` en indiquant comme CN (Common Name) `www.mydom.com` en utilisant la clef `inter.key` avec la commande :

     `openssl req -new -key cert/inter.key -out cert/inter.csr` ... (à terminer)

   - Faire signer ce certificat par le root CA :

```
openssl x509 -req  -in cert/inter.csr -CA cert/rootca.crt -CAkey cert/rootca.key -set_serial 01 -out cert/inter.crt
```

   - Exporter les certificats PKCS12 dans `cert/inter.p12` :

```
openssl pkcs12 -export -out cert/inter.p12 -inkey cert/inter.key -in cert/inter.crt -CAfile cert/rootca.crt
```

   - Exporter à partir du certificat `inter.p12` le certificat et la clef au format PEM dans `cert/my.crt.pem` et `cert/my.key.pem` :

     `openssl pkcs12 -in cert/inter.p12 -out cert/my.key.pem -nocerts -nodes`

     `openssl pkcs12 -in cert/inter.p12 -out cert/my.crt.pem -clcerts -nokeys`

4. (1 point) Copier la clef au format `pem` dans le répertoire `/etc/ssl/private/` et le certificat dans le répertoire `/etc/ssl/certs/`

   Initialiser le fichier `default-ssl.conf` dans `etc/apache2/sites-enabled/` avec les commandes suivantes :

```
a2ensite default-ssl
a2enmod ssl
systemctl restart apache2
```

5. (2 points) Mettre à jour le fichier `default-ssl.conf` en changeant les noms des fichiers de la clef et du certificat, c'est-à-dire mettre les nouveaux à la place de `snakeoil`. Recharchez la configuration d'apache2.

6. (2 points) Importer `rootca.crt` dans votre navigateur comme certificat d'autorité. La procédure dépendra de votre navigateur :

   - Firefox : `Settings > Privacy & Security > View Certificates > Authorities > Import`
   - Pour un navigateur en console comme `links`, faire les commandes suivantes :

```
export SSL_CERT_DIR=$PWD/cert
export SSL_CERT_FILE=$PWD/cert/rootca.crt
```

   Quelque soit votre navigateur, ajouter dans `/etc/hosts` la ligne suivante :

```
10.0.2.15 debian-1  www.mydom.com
```

   Pour finir, vérifier que cela fonctionne en allant dans un navigateur sur `https://www.mydom.com:443`

   Pourquoi cela fonctionne-t-il avec `http://www.mydom.com/`

Exercise 3 (EFAIL (5 points))
Aller dans le dossier `efail`. Votre objectif sera de modifier le mail chiffré (`mail.txt`), et de l'envoyer sur le serveur SMTP (`http://localhost:5003`) pour en révéler son contenu. Durant cet exercice, **seul** le fichier `efail_exercice.py` devra être modifié, aucune autre modification n'est nécessaire. L'objectif est de compléter le fichier `efail_exercice.py` pour monter l'attaque EFAIL vue en cours.

1. Désactiver les proxy `http_proxy` et `https_proxy`.
2. Consulter vos mails en allant sur la page `http://127.0.0.1:5003`, utilisez un navigateur classique.
3. C'est à cette étape que votre travail commence. Le mail que dont vous souhaitez voir le contenu se trouve dans `mail.txt`. Dans cet exercice, vous allez modifier UNIQUEMENT le fichier `efail_exercice.py`, qui contient la fonction

   `def efail( iv, ciphertext, known_plaintext , begin_replace, end_replace )`

   où `iv` est le vecteur d'initialisation, `ciphertext` est le chiffré, `known_plaintext` est le texte clair connu, tandis que `begin_replace` et `end_replace` sont respectivement le message à remplacer au début et à la fin du message.

   La fonction doit retourner un couple `(iv', ct')` où `iv'` sera le nouveau vecteur d'initialisation et `ct'` le nouveau chiffré. La fonction est partiellement codée, il ne vous faudra que remplacer les parties entre chevrons (`< .. >`).

   Pour vous éviter de perdre du temps, nous vous fournissons `efail.py`, qui se chargera de lire le fichier `mail.txt`, de parser le chiffré, et d'appeler la fonction `efail` définie plus haut avec les bons paramètres. Pour exécuter le fichier `efail.py`, entrer la commande `python3 efail.py --mail mail.txt` Si votre fonction retourne le nécessaire, `efail.py` se chargera alors de générer le nouveau mail et de l'envoyer au serveur SMTP. Le message sera alors visible dans les logs du serveur. Le voyez-vous ?

Bonus

Exercise 4 (Mettre en place TLS 1.3 (3 points))
Exercice à réaliser dans **VDN**.

1. Tester ces commandes afin de vérifier que ces sites soient bien en TLS1.2 et TLS1.3.

   `openssl s_client -connect www.google.com:443 -tls1_2`

   `openssl s_client -connect www.google.com:443 -tls1_3`

   `openssl s_client -connect ayesh.me:443 -tls1_3`

Cloner le git:

`git clone https://github.com/drwetter/testssl.sh.git`

2. Tester le script `testssl.sh` sur `https://uca.fr`.
3. Essayez de vous connecter à `localhost` ou `127.0.0.1`. Si cela ne fonctionne pas, résolvez le problème.
4. Essayez de vous connecter à `https://127.0.0.1`. Si cela ne fonctionne pas, résolvez le problème en utilisant les commandes suivantes puis expliquez ce qu'elles font :

```
a2ensite default-ssl
a2enmod ssl
systemctl restart apache2
```

Penser à `unset` les variables `http_proxy` et `https_proxy`. A présent tenter à nouveau de vous connecter à `https://127.0.0.1`, cela devrait fonctionner.

5. Ouvrir ce fichier : `/etc/apache2/mods-available/ssl.conf` et enlever la version de TLS 1.3. Vérifier avec `testssl.sh` que les changements sont effectifs (attention à relancer le service `apache2`).

6. Maintenant mettre `TLS 1.3` est par défaut `SSLProtocol -all +TLSv1.3 -TLSv1.2` Vérifier que cela fonctionne.

7. Jouer avec les paramétres cryptographiques:

   - Authoriser uniquement les `Ciphersuites` suivants `TLS_CHACHA20_POLY1305_SHA256` et `TLS_AES_128_GCM_SHA256`.
   - Ajouter `TLS_AES_256_GCM_SHA384`

   Vérifier chaque fois que les bons options sont activées.

   Faire de même pour les courbes elliptiques(`Curves`):

   - `SSLOpenSSLConfCmd Curves X25519:sect571r1:secp521r1:secp384r1`
   - `SSLOpenSSLConfCmd Curves X25519:secp521r1:secp384r1:prime256v1`

ATTENTION. Faire un `service apache2 restart` entre chaque manipulation.