T.P. Sécurisation des services - SFTP

NOM : RENARD **PRÉNOM : Julien**

DATE : 08/02/24

Prérequis : machine virtuelle Debian déposée sur le FTP (BLOC1/3-DEB12.4.0)

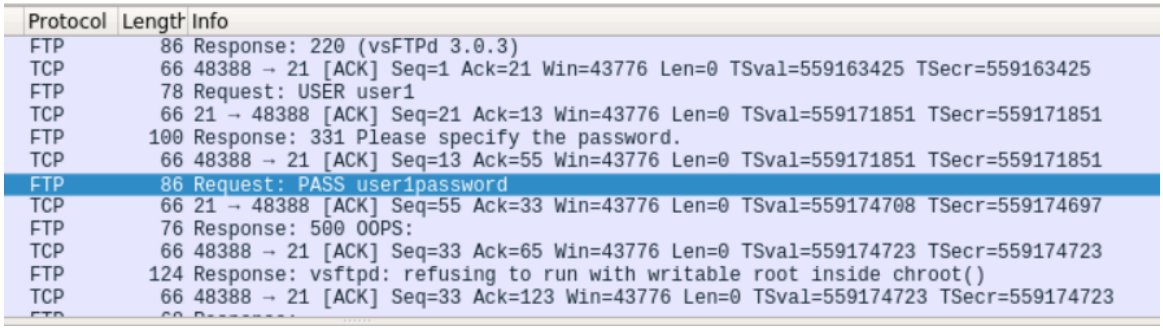
CONFIGURATION DU SERVEUR VSFTPD

- Installez les paquets **vsftpd, openssl, filezilla, wireshark**
- Rendez-vous dans le répertoire /etc et ouvrez le fichier **vsftpd.conf** pour le modifier
- Modifiez les lignes suivantes afin de configurer le service de téléchargement :
    - Pour autoriser la connexion par le compte anonymous : **anonymous_enable=YES**
    - Pour autoriser la connexion par les utilisateurs non privilégiés du système et ne les autoriser que de travailler dans leur répertoire sous /home : **local_enable=YES** **chroot_local_user=YES**
- Modifiez les lignes suivantes afin de configurer le service de dépôt anonyme :
    - Pour autoriser l'écriture dans les répertoires par défaut : **write_enable=YES**
    - Pour autoriser le dépôt (Upload) par l'utilisateur anonymous : **anon_upload_enable=YES**

- Sauvegardez le fichier
- Créez un répertoire sous **ftp** dans le répertoire **/var**, il sera réservé pour l'upload. Assignez lui les permissions nécessaires pour qu'il soit accessible pour l'utilisateur anonymous
- Relancez le service **vsftpd**

TEST DU SERVEUR VSFTPD NON SÉCURISÉ

- Créez l'utilisateur "**user1**" et assignez-lui le mot de passe "**user1password**".
- Lancez WireShark avec son interface graphique.
- Paramétrez la capture sur l'interface loopback (loop)
- Lancez la capture
- Installez le service **ftp**
- Connectez-vous à votre serveur en utilisant la commande **ftp localhost**
- Analysez le trafic capturé par WireShark, localisez le login et le mot de passé précédemment utilisé et insérez une capture d'écran dans votre compte rendu



MISE EN PLACE DE VSFTPD SÉCURISÉ

L'objectif de cette partie est de **sécuriser le trafic** entre client Filezilla et le serveur vsftpd par le biais du protocole **TLS**. On utilisera l'outil openssl pour générer le certificat garantissant, auprès du client, l'authenticité de la clé publique du serveur.

- Créez un certificat et une clé privée pour votre serveur avec la commande :
  sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vsftpd.pem -out /etc/ssl/private/vsftpd.pem

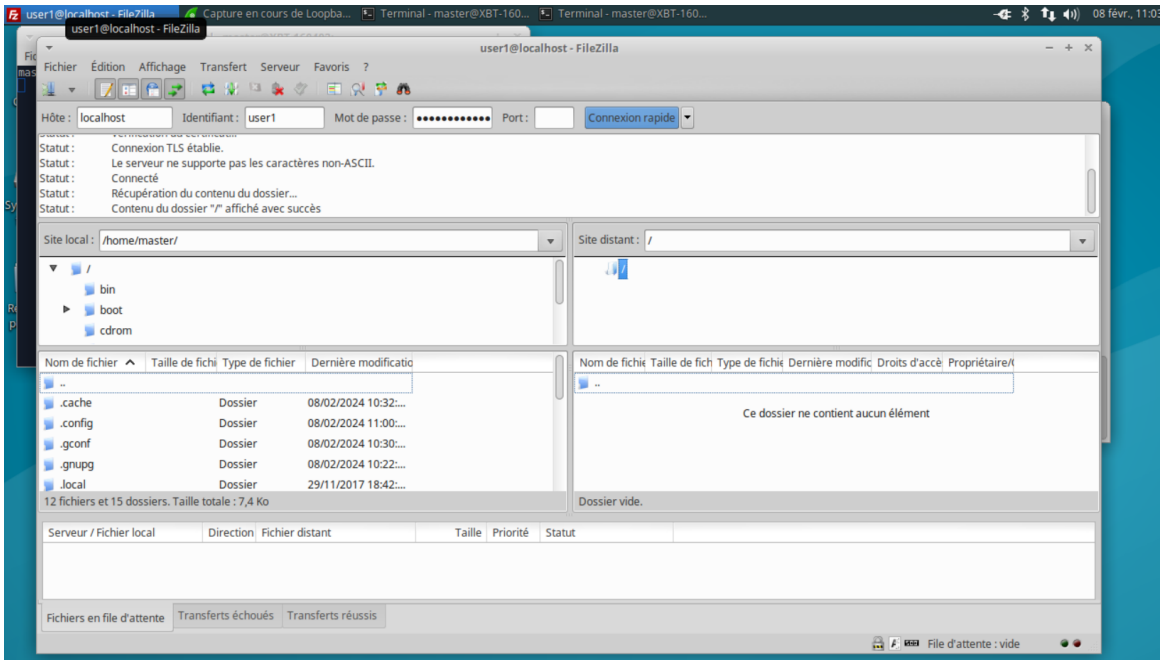
- Modifiez le fichier vsftpd.conf et ajoutez-y les lignes suivantes :

  rsa_cert_file=/etc/ssl/private/vsftpd.pem
  rsa_private_key_file=/etc/ssl/private/vsftpd.pem

ssl_enable=YES

force SSL. This will restrict clients that can't deal with TLS
ssl_enable=YES
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
#configure the server to use TLS (more secure than SSL)
#explicitly allowing TLS and denying the use of SSL
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
#If set to yes, all SSL data connections are required to exhibit SSL session reuse (which proves that they know the same master secret as the control channel). Although this is a secure default, it may break many FTP clients, so you may want to disable it
require_ssl_reuse=NO
#select which SSL ciphers vsftpd will allow for encrypted SSL connections
ssl_ciphers=HIGH
allow writable chroot if chroot_local_user was set to YES
allow_writeable_chroot=YES

- Sauvegardez votre fichier et relancez le service
- Vérifiez que le service fonctionne correctement
- Lancez WireShark et commencez une capture sur l'interface loopack.
- Lancez FileZilla sur votre machine virtuelle et connectez-vous au serveur vsftpd sécurisé.



- Analysez le trafic sur WireShark, identifiez avec des captures d'écrans :
  - La phase de connexion

```
1 0.000000000   127.0.0.1   127.0.0.1   TCP   74 48438 → 21 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=559684688 TSecr=0 WS=128
2 0.000025869   127.0.0.1   127.0.0.1   TCP   74 21 → 48438 [SYN, ACK] Seq=0 Ack=1 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=559684688 TSecr=5596…
3 0.000049428   127.0.0.1   127.0.0.1   TCP   66 48438 → 21 [ACK] Seq=1 Ack=1 Win=43776 Len=0 TSval=559684688 TSecr=559684688
```

SYN → SYN, ACK → ACK

  - Les étapes d'établissement d'un tunnel TLS

```
3 0.000049428   127.0.0.1   127.0.0.1   TCP   66 48438 → 21 [ACK] Seq=1 Ack=1 Win=43776 Len=0 TSval=559684688 TSecr=559684688
4 0.004187633   127.0.0.1   127.0.0.1   FTP   86 Response: 220 (vsFTPd 3.0.3)
5 0.004270873   127.0.0.1   127.0.0.1   TCP   66 48438 → 21 [ACK] Seq=1 Ack=21 Win=43776 Len=0 TSval=559684688 TSecr=559684688
6 0.004497765   127.0.0.1   127.0.0.1   FTP   76 Request: AUTH TLS
7 0.004505683   127.0.0.1   127.0.0.1   TCP   66 21 → 48438 [ACK] Seq=21 Ack=11 Win=43776 Len=0 TSval=559684688 TSecr=559684688
8 0.004566171   127.0.0.1   127.0.0.1   FTP   97 Response: 234 Proceed with negotiation.
9 0.005563953   127.0.0.1   127.0.0.1   FTP   304 Request: \026\003\001\000\351\001\000\000\345\003\003e\304\251;Z5eg~\002n\0313\365L\020\017\022\35…
10 0.011157940  127.0.0.1   127.0.0.1   FTP   1401 Response: \026\003\003\000=\002\000\0009\003\003\337\f\322t!9\375a\207\243\307h?9{\241j~\230\230s/…
```

```
    Length: 10
    Timestamp value: 559684688
    Timestamp echo reply: 559684688
▼ [SEQ/ACK analysis]
    [iRTT: 0.000049428 seconds]
    [Bytes in flight: 10]
    [Bytes sent since last PSH flag: 10]
File Transfer Protocol (FTP)
▼ AUTH TLS\r\n
    Request command: AUTH
    Request arg: TLS
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 3e c5 92 40 00 40 06  77 25 7f 00 00 01 7f 00   .>..@.@. w%......
0020  00 01 bd 36 00 15 e4 b9  41 1b a2 21 ba 99 80 18   ...6.... A..!....
0030  01 56 fe 32 00 00 01 01  08 0a 21 5c 1c 50 21 5c   .V.2.... ..!\.P!\
0040  1c 50 41 55 54 48 20 54  4c 53 0d 0a              .PAUTH T LS..
```