

PSMM

Ajout d'un utilisateur au groupe sudo

Création de l'utilisateur monitor

adduser monitor

Installer sudo

apt install sudo -y

Ajout de monitor au groupe sudo

usermod -aG sudo monitor

```
root@letoval-ftp:# adduser monitor
Ajout de l'utilisateur « monitor » ...
Ajout du nouveau groupe « monitor » ...
Ajout du nouvel utilisateur « monitor » (1001) avec le groupe « monitor » (1001) ...
Copie des fichiers personnels à /home/monitor ...
Nouveau mot de passe :
Répétez le nouveau mot de passe :
password : mot de passe mis à jour avec succès
Modifier les informations associées à un utilisateur pour monitor
Entrer la nouvelle valeur, ou appuyer sur ENTER pour la valeur par défaut
  NOM []: monitor
    Numéro de chambre []:
    Téléphone professionnel []:
    Téléphone personnel []:
    Autre []:
Cette information est-elle correcte ? [o/n]
Ajout du nouvel utilisateur « monitor » aux groupes supplémentaires « users » ...
Ajout de l'utilisateur « monitor » à un groupe « users » ...
root@letoval-ftp:# usermod -aG sudo monitor
```

```
PS C:\Users\valen> ssh-keygen -t rsa -b 4096 -C "monitor@letoval-ftp"
Generating public/private rsa key pair.
Enter file in which to save the key (C:/Users/valen/.ssh/id_rsa):
C:/Users/valen/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:/Users/valen/.ssh/id_rsa.
Your public key has been saved in C:/Users/valen/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:NTTGrQcAn162a/K14iB198Dk00MC0u3RuqlhdQTMM monitor@letoval-ftp
The key's randomart image is:
+---[RSA 4096]---+
| .+.
| Eo.....
| o.+oooo
| ..+o*oX.
| ...oSo+=.
| ...oo oo...
| .+o...o =..
| .+o ... + o.
| .o. oo .
+---[SHA256]-----+
```

```
PS C:\Users\valen> dir C:/Users/valen/.ssh
 Répertoire : C:/Users/valen/.ssh

 Mode          LastWriteTime    Length Name
-- --          -- --          -- -- 
-a----  14/05/2025  13:33      4444 id_ed25519
-a----  14/05/2025  13:33       94 id_ed25519.pub
-a----  15/09/2025  12:35      3389 id_rsa
-a----  15/09/2025  12:35      746 id_rsa.pub
-a----  09/09/2025  09:31     22664 known_hosts
-a----  09/09/2025  09:31     21924 known_hosts.old
```

```
PS C:\Users\valen> type C:/Users/valen/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAACACQjHqqe8XWshmSWHMQjWGgBQhOMjqcjO2VsAPLYnA7t2/4vkwWcwq+eCZ9/y7owGGyzuebu9Xi0tZWR4YJ/...
dTCOpGh1PxRV2FN0lF9oMnzkZ8mg1kgUtYO4X/j0rvS4XnTiTjSb4qCFxGBHpBY8vm5l4E4x8J5NLH7b1vn6DcnPC3au02GMc2SD/YUw1KTk3U...
izLpIrbYafiu63QPSEUATKx6u4ocXZ5qolUIWea5+lNodr8hrrqrbf4a4fqA+sAmkv4EZM63oJP/AVq/fM+CMgvSyDXs1AcGkUq...
Q/Y8a62LbbyNcXT2a3BiKSyQEumrKmGSu4ltT/0TjzHwdWU+yGn3RvBRKKpeSV6bpRHk3g6ktScfGAVSZ/C2GHt4vSuU8FKH2urc...
JaE/2rRiwkUchOWWqYL7wupq0dJWG+ZtBk+6uaovBVxDzhXTt4VAKKp/AX5n/gCSpteZdP7sANhrwlaA6tvAB/8uVnSjo0Qo8tQa...
Mw8IDSiFF7cri4PZ4EnNWc/vdW058403VrNQE1xm8z6CcWKuo1GHxpIgQ== monitor@letoval-ftp
```

Sur la VM créer le dossier .ssh et donner les permissions 700

mkdir -p ~/.ssh

chmod 700 ~/.ssh

Créer le fichier authorized_keys

touch ~/.ssh/authorized_keys

chmod 600 ~/.ssh/authorized_keys

Ajouter la clé publique au fichier authorized_keys

echo "ssh-rsa

```
AAAAB3NzaC1yc2EAAAQABAAACACQjHqqe8XWshmSWHMQjWGgBQhOMjqcjO2VsAPLYnA7t2/4vkwWcwq+eCZ9/y7owGGyzuebu9Xi0tZWR4YJ/...
X3MpZbVXzdTCoPGh1PxRV2FN0lF9oMnzkZ8mg1kgUtYO4X/j0rvS4XnTiTjSb4qCFxGBHpBY8vm5l4E4x8J5NLH7b1vn6DcnPC3au02GMc2SD/YUw1KTk3U...
wqSjP/0WNWKMSSX7A/43izLpIrbYafiu63QPSEUATKx6u4ocXZ5qolUIWea5+lNodr8hrrqrbf4a4fqA+sAmkv4EZM63oJP/AVq/fM+CMgvSyDXs1AcGkUq...
biZ44jB/YNd6KJe/SghKSe4FKLxZw4Q/Y8a62LbYNcXT2a3BiKSyQEumrKmGSu4ltT/0TjzHwdWU+yGn3RvBRKKpeSV6bpRHk3g6ktScfGAVSZ/C2GHt4vSuU8FKH2urc...
KH2urcLpjmrfrgxIA2DXGITBjjbD4UGd29DQzAJjaE/2rRiwkUchOWWqYL7wupq0dJWG+ZtBk+6uaovBVxDzhXTt4VAKKp/AX5n/gCSpteZdP7sANhrwlaA6tvAB/8uVnSjo0Qo8tQa...
B/8uVnSjo0Qo8tQa8t/toBLULk5p7IXzlryUyKZ90k8cj8EqMMw8IDSiFF7cri4PZ4EnNWc/vdW058403VrNQE1xm8z6CcWKuo1GHxpIgQ== monitor@letoval-...
ftp" >> ~/.ssh/authorized_keys
```

Maintenant la connexion ssh se fera sans mdp

Editer la configuration de SSH pour modifier les règles :

sudo nano /etc/ssh/sshd_config

```
PubkeyAuthentication yes
PermitRootLogin no
PasswordAuthentication no
```

Config serveur FTP

Installer ProFTPD

`sudo apt install proftpd-basic -y`

Créer un utilisateur

`sudo adduser ftpuser`

Editer la configuration ProFTPD

Nous devons configurer ProFTPD pour enregistrer toutes les tentatives de connexion (réussies ET échouées) dans des fichiers de logs structurés. Ces logs nous permettront plus tard de détecter les attaques par force brute ou les tentatives d'accès non autorisées.

Créer des fichiers de logs détaillés que nos scripts Python pourront analyser pour extraire les tentatives d'accès malveillantes.

Pour cela on ajoute à la fin du fichier `proftpd.conf`: `sudo nano /etc/proftpd/proftpd.conf`

1. Les LogFormat (formats de logs)

```
LogFormat default "%h %l %u %t \"%r\" %s %b"
LogFormat auth     "%v [%P] %h %t \"%r\" %s"
LogFormat write    "%h %l %u %t \"%r\" %s %b"
```

Que signifient ces codes ?

- `%h` = Adresse IP du client (qui se connecte)
- `%l` = Nom d'utilisateur remote (souvent "-")
- `%u` = Nom d'utilisateur authentifié
- `%t` = Date et heure de la connexion
- `%r` = Première ligne de la requête (commande FTP)
- `%s` = Code de statut (succès/échec)
- `%b` = Nombre d'octets transférés
- `%v` = Nom du serveur virtuel
- `%P` = PID du processus

2. Les fichiers de logs :

```
TransferLog /var/log/proftpd/xferlog      # Transferts de fichiers
SystemLog   /var/log/proftpd/proftpd.log # Événements généraux du serveur
ExtendedLog /var/log/proftpd/auth.log AUTH auth # Tentatives d'authentification
```

Pourquoi ces 3 fichiers ?

- **xferlog** : Enregistre tous les transferts de fichiers
- **proftpd.log** : Enregistre les événements système (démarrage, arrêt, erreurs)
- **auth.log** : **LE PLUS IMPORTANT** - enregistre TOUTES les tentatives de connexion (mot de passe correct OU incorrect)

C'est le fichier **auth.log** qui nous intéressera le plus pour détecter les attaques !

Config serveur FTP

Créer le dossier de log

```
sudo mkdir -p /var/log/proftpd
```

Permettre au fichier proftpd d'écrire dans le dossier /var/log/proftpd

```
sudo chown proftpd:adm /var/log/proftpd
```

Redémarrer ftpd

```
sudo systemctl restart proftpd
```

Vérifier que les fichiers de logs sont créés

```
ls -la /var/log/proftpd/
```

```
monitor@letoval-ftp:/var/log$ ls -la /var/log/proftpd/
total 12
drwxr-xr-x 2 proftpd adm 4096 15 sept. 13:18 .
drwxr-xr-x 8 root root 4096 15 sept. 13:18 ..
-rw-r----- 1 root root 0 15 sept. 13:18 controls.log
-rw-r----- 1 root root 486 15 sept. 13:38 proftpd.log
```

Test de connexion FTP réussie depuis l'hôte bash : **ftp 192.168.175.140**

Nous allons tester une connexion réussie avec ftpuser et son mdp puis 2 connexions incorrectes avec des utilisateurs ftp qui n'existent pas.

```
monitor@letoval-ftp:/var/log$ ls -la /var/log/proftpd/
total 16
rwxr-xr-x 2 proftpd adm 4096 15 sept. 13:52 .
rwxr-xr-x 8 root root 4096 15 sept. 13:18 ..
rwxr--r-- 1 root root 1914 15 sept. 13:53 access.log
rwxr----- 1 root root 0 15 sept. 13:18 controls.log
rwxr----- 1 root root 1409 15 sept. 13:53 proftpd.log
rwxr--r-- 1 root root 0 15 sept. 13:52 xferlog
```

Les fichiers logs ont bien été créés et le fichier access.log qui va contenir les tentatives de connexion FTP

Lecture du fichier /var/log/proftpd/acces.log

access.log (Log d'accès)

- Contenu :** Toutes les commandes FTP exécutées par les clients
- Format :** Format structuré (IP, utilisateur, timestamp, commande, code de réponse)
- Exemple :** 192.168.175.1 UNKNOWN - [15/sept./2025:13:53:11 +0200] "USER hacker"
- 331**
- Utilité :** Analyse de trafic, statistiques d'utilisation, monitoring automatisé

```
monitor@letoval-ftp:/var/log$ sudo cat /var/log/proftpd/access.log
192.168.175.1 UNKNOWN - [15/sept./2025:13:48:04 +0200] "OPTS UTF8 ON" 200 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:48:04 +0200] "OPTS UTF8 ON" - - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:48:21 +0200] "USER ftpuser" 331 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:48:23 +0200] "PASS (hidden)" 530 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:49:06 +0200] "QUIT" 221 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:52:25 +0200] "OPTS UTF8 ON" 200 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:52:25 +0200] "USER ftpuser" 331 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:52:54 +0200] "USER ftpuser" 331 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:52:56 +0200] "PASS (hidden)" 230 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:08 +0200] "QUIT" 221 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:08 +0200] "OPTS UTF8 ON" - - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:11 +0200] "USER hacker" 331 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:14 +0200] "PASS (hidden)" 530 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:24 +0200] "QUIT" 221 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:31 +0200] "OPTS UTF8 ON" 200 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:31 +0200] "OPTS UTF8 ON" - - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:36 +0200] "USER tentativeinvasion" 331 - "Referer" "User-Agent"
192.168.175.1 UNKNOWN - [15/sept./2025:13:53:37 +0200] "PASS (hidden)" 530 - "Referer" "User-Agent"
```

-Connexion réussie de ftpuser avec le **code 230 = succès d'authentification**
-Tentatives échouées hacker et **tentativeinvasion** avec **code 530 = échec authentification**

-

Lecture du fichier /var/log/proftpd/proftpd.log

proftpd.log (Log système)

- Contenu :** Messages système du serveur ProFTPD (démarrage, arrêt, erreurs système)
- Format :** Messages détaillés en langage naturel avec contexte
- Exemple :** USER hacker: no such user found from 192.168.175.1
- Utilité :** Diagnostics système, erreurs détaillées, événements administratifs

```
monitor@letoval-ftp:/var/log$ sudo cat /var/log/proftpd/proftpd.log
2025-09-15 13:18:56.932 letoval-ftp proftpd[1888] letoval-ftp: ProFTPD 1.3.8 (stable) (built Sat Nov 30 2024 22:32:48 UTC) standalone mode STARTUP
2025-09-15 13:37:59.985 letoval-ftp proftpd[1888] letoval-ftp: ProFTPD killed (signal 15)
2025-09-15 13:37:59.986 letoval-ftp proftpd[1888] letoval-ftp: ProFTPD 1.3.8 standalone mode SHUTDOWN
2025-09-15 13:38:01.907 letoval-ftp proftpd[1951] letoval-ftp: ProFTPD 1.3.8 (stable) (built Sat Nov 30 2024 22:32:48 UTC) standalone mode STARTUP
2025-09-15 13:46:14.007 letoval-ftp proftpd[1951] letoval-ftp: ProFTPD killed (signal 15)
2025-09-15 13:46:14.008 letoval-ftp proftpd[1951] letoval-ftp: ProFTPD 1.3.8 standalone mode SHUTDOWN
2025-09-15 13:46:14.202 letoval-ftp proftpd[1889] letoval-ftp: ProFTPD 1.3.8 (stable) (built Sat Nov 30 2024 22:32:48 UTC) standalone mode STARTUP
2025-09-15 13:48:23.831 letoval-ftp proftpd[2080] letoval-ftp (192.168.175.1[192.168.175.1]): USER ftpuser: no such user found from 192.168.175.1 to ::ffff:192.168.175.140:21
2025-09-15 13:53:14.507 letoval-ftp proftpd[2080] letoval-ftp (192.168.175.1[192.168.175.1]): USER hacker: no such user found from 192.168.175.1 to ::ffff:192.168.175.140:21
2025-09-15 13:53:37.704 letoval-ftp proftpd[2051] letoval-ftp (192.168.175.1[192.168.175.1]): USER tentativeinvasion: no such user found from 192.168.175.1 to ::ffff:192.168.175.140:21
```

Sécuriser serveur sftp

Arréter et désactiver proftpd

sudo systemctl stop proftpd

sudo systemctl disable proftpd

désinstaller proftpd sudo apt remove proftpd-basic -y

Se connecter depuis windows :

Sftp est installé en même temps que ssh

Créer dossier ssh, fichier authorized_keys

```
PS C:\Users\valen> sftp monitor@192.168.175.140
Connected to 192.168.175.140.
sftp> ls
sftp> pwd
Remote working directory: /home/monitor
sftp> quit
PS C:\Users\valen>
```

Config serveur Web

Télécharger paquets sudo, Créer l'utilisateur monitor, mettre dans sudo

su -

apt install sudo

adduser monitor

usermod -aG sudo monitor

Préparer SSH :

Créer dossier ssh, fichier authorized_keys

```
monitor@letoval-web:~$ mkdir -p ~/.ssh
chmod 700 ~/.ssh
touch ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQACjHqqe8XWshmSWHMQjWgBQh0Mjqcj02VsAPLYnA7t2/4vkWlcwq+eCZ9/y7owGGyzuewbu9Xi0tZWR4YJ/EgL
X3MpZbVXzdTC0PGhLPxRV2FN0lF9oMnzkZ8mg1kgUtY04X/j0rvS4XnTiTjSb4qCFfXGBHpBY8vm5l4E4x8J5NLH7blvn6DcWkuolGHxplgQ== monitor@letoval-ftp" > ~/.ssh/authorized_keys
```

Editer la configuration de SSH pour modifier les règles :

sudo nano /etc/ssh/sshd_config

```
PermitRootLogin no
PubkeyAuthentication yes
PasswordAuthentication no
```

Nous devons créer un site web protégé par authentification HTTP Basic pour générer des logs d'accès (réussis et échoués) que nos scripts Python analyseront pour détecter les tentatives d'intrusion web.

Apache est déjà installé car on a laissé la case serveur web cochée lors de l'installtion de la VM.

Créer le dossier pour notre site protégé :

sudo mkdir -p /var/www/html/secure

sudo chown www-data:www-data /var/www/html/secure

Créer une page web simple :

sudo nano /var/www/html/secure/index.html

Créer un fichier de mots de passe pour l'authentification :

sudo htpasswd -c /etc/apache2/.htpasswd webadmin

```
monitor@letoval-web:~$ sudo htpasswd -c /etc/apache2/.htpasswd webadmin
New password:
Re-type new password:
Adding password for user webadmin
```

Nous devons maintenant dire à Apache d'utiliser le fichier de mots de passe que nous venons de créer pour protéger le dossier /secure/. Dans le but d'ctiver l'authentification HTTP Basic et configurer les logs détaillés pour capturer toutes les tentatives d'accès.

Créer le fichier .htaccess :

sudo nano /var/www/html/secure/.htaccess

```
# Définit le type d'authentification utilisé (Basic = popup utilisateur/mot de passe)
AuthType Basic

# Message affiché dans la popup d'authentification du navigateur
AuthName "Zone Securisee - Acces Restreint"

# Chemin vers le fichier contenant les utilisateurs/mots de passe autorisés
AuthUserFile /etc/apache2/.htpasswd

# Exige qu'un utilisateur valide (présent dans le fichier .htpasswd) soit connecté
Require valid-user
```

Config serveur Web

Nous devons activer les modules Apache nécessaires à l'authentification et configurer Apache pour qu'il accepte les fichiers .htaccess.

Rendre l'authentification fonctionnelle et configurer les logs détaillés pour capturer les tentatives d'intrusion.

Activer les modules Apache nécessaires :

`sudo a2enmod rewrite`

`sudo a2enmod auth_basic`

`sudo a2enmod authz_user`

Ces commandes activent les modules pour l'authentification et la réécriture d'URL

Modifier la configuration du site par défaut :

`sudo nano /etc/apache2/sites-available/000-default.conf`

Ce fichier définit le comportement du site web par défaut d'Apache. Par défaut, Apache ne permet pas certaines fonctionnalités de sécurité que nous devons activer.

Pourquoi <Directory /var/www/html>

Sans cette section : Apache ignore complètement les fichiers .htaccess - notre authentification ne fonctionnerait pas du tout.

Avec cette section : Nous autorisons Apache à lire et appliquer les règles d'authentification que nous avons mises dans .htaccess.

Pourquoi AllowOverride All

Par défaut Apache : AllowOverride None - bloque tous les fichiers .htaccess pour des raisons de performance

Notre besoin : AllowOverride All - permet à notre .htaccess de définir l'authentification

Pourquoi Options Indexes FollowSymLinks

Indexes : Permet de lister les fichiers d'un dossier si pas de fichier index.html

FollowSymLinks : Permet de suivre les liens symboliques

Notre intérêt : Génère plus de logs quand les attaquants explorent notre serveur

Pourquoi Require all granted

Rôle : Autorise l'accès de base au dossier web

Subtilité importante : Cette directive s'applique au dossier parent, mais notre .htaccess dans /secure/ va override cette règle pour demander l'authentification

Logique : Accès libre au site principal, authentification obligatoire dans /secure/

Impact sur notre surveillance :

Les logs Apache captureront : tentatives d'accès à /secure/, codes de réponse HTTP (401 = échec auth, 200 = succès)

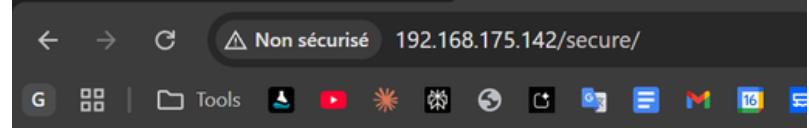
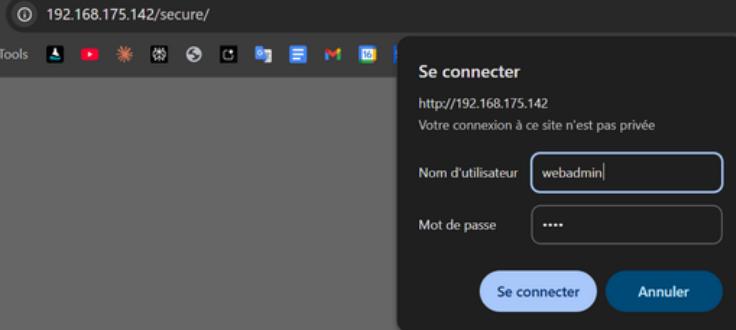
Les attaquants généreront : des erreurs 401/403 quand ils essaient des mauvais mots de passe

Nos scripts Python analyseront : ces logs pour détecter les tentatives d'intrusion

Sans ces modifications, Apache bloquerait notre système d'authentification et nous n'aurions aucun log d'intrusion à analyser - le piège à attaquants ne fonctionnerait pas.

Config serveur Web

Test connexion réussie :



Accès Autorisé

Vous êtes connecté à la zone sécurisée du serveur letoval-web

Surveillance active - Tous les accès sont loggés

Tests connexion échouées

utilisateurs testés : intrusion, hacker, pentesting, triche

sudo tail -f /var/log/apache2/access.log

```
monitor@letoval-web:~$ sudo tail -f /var/log/apache2/access.log
[sudo] Mot de passe de monitor :
192.168.175.1 - - [16/Sep/2025:11:06:27 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:06:38 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:06:49 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:06:49 +0200] "-" 408 0 "-" "-"
192.168.175.1 - - [16/Sep/2025:11:06:49 +0200] "-" 408 0 "-" "-"
192.168.175.1 - - [16/Sep/2025:11:07:00 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:07:06 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:07:13 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:07:26 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/140.0.0.0 Safari/537.36"
192.168.175.1 - - [16/Sep/2025:11:07:31 +0200] "GET /secure/ HTTP/1.1" 401 748 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/140.0.0.0 Safari/537.36"
```

Tentatives d'intrusion parfaitement capturées :

intrusion, hacker, pentesting, triche - **codes 401 (échec authentification)**

IP source : 192.168.175.1

Horodatage complet : avec date, heure précise

User-Agent détaillé : Informations sur le navigateur utilisé

Format structuré parfait pour l'analyse Python :

IP - utilisateur [timestamp] "GET /secure/" code taille User-Agent

sudo tail -f /var/log/apache2/error.log

```
monitor@letoval-web:~$ sudo tail -f /var/log/apache2/error.log
[Mon Sep 15 15:00:34.915761 2025] [mpm_event:notice] [pid 710:tid 710] AH00489: Apache/2.4.65 (Debian) configured -- resuming normal operations
[Mon Sep 15 15:00:34.916797 2025] [core:notice] [pid 710:tid 710] AH00094: Command line: '/usr/sbin/apache2'
[Tue Sep 16 10:19:01.634833 2025] [mpm_event:notice] [pid 710:tid 710] AH00492: caught SIGWINCH, shutting down gracefully
[Tue Sep 16 10:19:01.730193 2025] [mpm_event:notice] [pid 2278:tid 2278] AH00489: Apache/2.4.65 (Debian) configured -- resuming normal operations
[Tue Sep 16 10:19:01.730496 2025] [core:notice] [pid 2278:tid 2278] AH00094: Command line: '/usr/sbin/apache2'
[Tue Sep 16 11:07:00.896127 2025] [auth_basic:error] [pid 2279:tid 2324] [client 192.168.175.1:55970] AH01618: user intrusion not found: /secure/
[Tue Sep 16 11:07:06.358235 2025] [auth_basic:error] [pid 2279:tid 2327] [client 192.168.175.1:58133] AH01618: user hacker not found: /secure/
[Tue Sep 16 11:07:13.066749 2025] [auth_basic:error] [pid 2280:tid 2305] [client 192.168.175.1:62196] AH01618: user pentesting not found: /secure/
[Tue Sep 16 11:07:26.793517 2025] [auth_basic:error] [pid 2279:tid 2330] [client 192.168.175.1:62442] AH01618: user triche not found: /secure/
```

Timestamp précis : [Tue Sep 16 11:07:00.896127 2025]

Type d'erreur : [auth_basic:error] - échec d'authentification HTTP Basic

IP + port source : [client 192.168.175.1:55970] - adresse et port de l'attaquant

Code erreur Apache : AH01618 - utilisateur inexistant

Tentatives d'utilisateurs : intrusion, hacker, pentesting, triche

Config serveur SQL

Télécharger paquets sudo, Créer l'utilisateur monitor, mettre dans sudo

su -

apt install sudo

adduser monitor

usermod -aG sudo monitor

Préparer SSH :

Créer dossier ssh, fichier authorized_keys

```
root@letoval-sql:~# mkdir -p ~/.ssh
root@letoval-sql:~# chmod 700 ~/.ssh
root@letoval-sql:~# touch ~/.ssh/authorized_keys
root@letoval-sql:~# chmod 600 ~/.ssh/authorized_keys
root@letoval-sql:~# echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQjHqge8XWshmSWHMQjWGGBQhOMjqcj02VsAPLYnA7t2/4vkwWcwq+eCZ9/y7owGGyzuewbu9Xi0tzWR4Yj/EgLX3MpZbVXzdTCoPGhlPxRV2FN0LF9oMnzkZ8mg1kgUtY04X/j0rv5XnT1jSb4qCFFxGBhpBY8vm5l4E4x835NLH7blvn6DcnPC3au02GMc2SD/YUw1KTh3Ue+wq5jP/oWNWKMSXX7A/43izlpIrbYafUY63QPSEUATKx6u4ocXZ5qolIWea5+lNodr8hrrqrdf4a4fqA+sAmkv4EZM63oJPf/AVq/fM+CGMvSyDXs1ACgKUqX/biZ44jB/YNd6KJe/SghKSe4FKLXzWUQ/Y8a62LbYNcXT2a3BiKSyQEumrkMGSu4ltT/0TJzHwdWU+yGn3RvBRKkpeSV6bpRHk3g6ktsCfGAVSZ/C2Ght4vSUu8FKH2urcLpJmrfrGxLA2DXglTBjzbD4UGd29DQzAJJaE/2rRiwkUchOWWqYL7wupq0dJWG+ZtBk+6uaovBXvDzhXTt4VAKKp/AX5n/gCSpteZdP7sANhrw1aA6tvAB/8uVnSJ00Qo8tQa8t/toBLULk5fP7IXzlrUyKZ90k8cj8EqMMw8IDSifF7cri4PZ4EnNWc/vdW058403VrNQE1xm8z6CcWKuolGHxplgQ== monitor@letoval-ftp" > ~/.ssh/authorized_keys
```

Editer la configuration de SSH pour modifier les règles :

sudo nano /etc/ssh/sshd_config

Puis redémarrer sudo systemctl restart ssh

```
PermitRootLogin no
PubkeyAuthentication yes
PasswordAuthentication no
```

Installation mariadb-server -y:

sudo apt install mariadb-server -y

Sécuriser l'installation MariaDB:

sudo mysql_secure_installation qui est un script de sécurisation automatique fourni avec MariaDB/MySQL pour renforcer la sécurité d'une installation fraîche.

Et répondre :

- Root password : Créer un mot de passe fort
- Remove anonymous users : Y
- Disallow root login remotely : N (nous configurerons plus tard)
- Remove test database : Y - Reload privilege tables : Y

Configuration MariaDB pour le système de surveillance

Nous devons créer une base de données dédiée avec des tables structurées pour stocker les tentatives d'intrusion détectées sur les serveurs FTP et Web. Cette base centralisera tous les logs pour faciliter l'analyse.

Créer la structure de base de données et configurer les accès distants pour que les scripts Python puissent s'y connecter depuis l'hôte windows.

Se connecter à MariaDB en tant que root :

sudo mysql -u root -p

Créer la base de données de surveillance :

CREATE DATABASE surveillance_logs;

USE surveillance_logs;

Créer les tables pour stocker les logs d'intrusion :

```
MariaDB [surveillance_logs]> -- Table pour les tentatives d'intrusion FTP
MariaDB [surveillance_logs]> CREATE TABLE ftp_intrusions (
->     id INT AUTO_INCREMENT PRIMARY KEY,
->     ip_source VARCHAR(45) NOT NULL,
->     username_attempted VARCHAR(255),
->     timestamp_attack DATETIME NOT NULL,
->     server_target VARCHAR(100) NOT NULL,
->     attack_type VARCHAR(50) DEFAULT 'FTP_AUTH_FAILED',
->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
-> );
MariaDB [surveillance_logs]> -- Table pour les tentatives d'intrusion Web
MariaDB [surveillance_logs]> CREATE TABLE web_intrusions (
->     id INT AUTO_INCREMENT PRIMARY KEY,
->     ip_source VARCHAR(45) NOT NULL,
->     username_attempted VARCHAR(255),
->     timestamp_attack DATETIME NOT NULL,
->     server_target VARCHAR(100) NOT NULL,
->     url_targeted VARCHAR(500) NOT NULL,
->     http_code INT,
->     user_agent TEXT,
->     attack_type VARCHAR(50) DEFAULT 'HTTP_AUTH_FAILED',
->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
-> );
Query OK, 0 rows affected (0,025 sec)

MariaDB [surveillance_logs]>
MariaDB [surveillance_logs]> -- Table pour les tentatives d'intrusion Web
MariaDB [surveillance_logs]> CREATE TABLE web_intrusions (
->     id INT AUTO_INCREMENT PRIMARY KEY,
->     ip_source VARCHAR(45) NOT NULL,
->     username_attempted VARCHAR(255),
->     timestamp_attack DATETIME NOT NULL,
->     server_target VARCHAR(100) NOT NULL,
->     url_targeted VARCHAR(500) NOT NULL,
->     http_code INT,
->     user_agent TEXT,
->     attack_type VARCHAR(50) DEFAULT 'HTTP_AUTH_FAILED',
->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
-> );
Query OK, 0 rows affected (0,013 sec)
```

Config serveur Web

Configuration de l'accès distant à MariaDB

- **Modifier la configuration MariaDB pour autoriser les connexions distantes**

Par défaut, MariaDB n'écoute que les connexions locales (localhost). Pour permettre à un client distant de se connecter, il faut configurer MariaDB pour écouter sur toutes les interfaces réseau ou au moins sur l'IP du serveur.

sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

Modifier bind-address = 1.1.1.1 en bind-address

= 0.0.0.0 ce qui permet à

mariaDB d'écouter sur toutes les interfaces réseau

- **Créer un utilisateur MariDB limité à l'ip windows**

sudo mysql -u root -p

CREATE USER 'valdeb'@'localost' IDENTIFIED BY 'poaaaaa';

GRANT ALL PRIVILEGES ON surveillance_logs.* TO 'monuser'@'10.10.71.243';

FLUSH PRIVILEGES;

- **Vérifier le pare-feu sur le serveur MariaDB**
- **Tester la connexion distante depuis Windows**

Télécharger HeidiSQL qui est un client graphique MariaDB



HeidiSQL 12.11.0.7065 - Gestionnaire de sessions: SurveillanceLogMariaDB

SurveillanceLogMariaDB - 15

Paramètres Tunnel SSH Avancé SSL Statistiques

Type de réseau : MariaDB or MySQL (TCP/IP)

Bibliothèque : libmariadb.dll

Nom ou IP de l'hôte : 192.168.175.139

Utilisateur : valdeb

Mot de passe : ****

Port : 3306

Bases de données : Séparation par point-virgule

Commentaire :

SurveillanceLogMariaDB\surveillance_logs\web_intrusions - HeidiSQL 12.11.0.7065

SurveillanceLogMariaDB 240.0 kB

information_schema 208.0 kB

surveillance_logs 32.0 kB

ftp_intrusions 16.0 kB

web_intrusions 16.0 kB

web_intrusions

Colonnes : Ajouter Supprimer Monter Desmonter

#	Nom	Type de données	Taille/Ensemble	Non sig.	NULL aut.	ZERO..	Par défaut	Commentaire	Collation
1	id	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	AUTO_INCREMENT	
2	ip_source	VARCHAR	45	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pas de défaut	utf8mb4_general_ci
3	username_att...	VARCHAR	255	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL	utf8mb4_general_ci
4	timestamp_att...	DATETIME		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pas de défaut	
5	server_target	VARCHAR	100	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pas de défaut	utf8mb4_general_ci
6	url_targeted	VARCHAR	500	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pas de défaut	utf8mb4_general_ci
7	http_code	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL	utf8mb4_general_ci
8	user_agent	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL	utf8mb4_general_ci
9	attack_type	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'HTTP_AUTH_FAIL...	utf8mb4_general_ci
10	created_at	TIMESTAMP		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	current_timestamp	utf8mb4_general_ci

Développement des scripts

Script ssh_login.py

Depuis la machine host se connecter sur un le serveur ftp et lancer une commande shell ls -la.

- Vérifier que python est installé sur l'hôte W: `python --version`
- Installer la bibliothèque SSH / `pip install paramiko`

Paramiko implémente le protocole SSH version 2 en Python pur, permet d'établir des connexions SSH depuis Python vers les serveurs Linux, gère l'authentification par clés SSH (fichiers id_rsa), Exécute des commandes à distance et récupère les résultats

- Installer les clients MariaDB : `pip install mysql-connector-python`

C'est une bibliothèque Python qui permet de se connecter à une base de données MySQL/MariaDB

- Créer le répertoire du projet :
`mkdir D:\PSMM_Scripts`
`cd D:\PSMM_Scripts`

ssh_login_sudo.py

Ce script étend le précédent en ajoutant la capacité d'exécuter des commandes avec priviléges administrateur.

C'est essentiel car les fichiers de logs système (/var/log/apache2/access.log, /var/log/proftpd/proftpd.log) ne sont lisibles qu'avec des droits sudo.

Nous devons avant **modifier le fichier sudoer sur chaque serveur** avec `sudo visudo`

Ajouter à la fin du fichier :

```
monitor ALL=(ALL) NOPASSWD: /bin/systemctl, /bin/ls, /bin/cat, /usr/bin/tail, /bin/journalctl
```

Ce script permet d'automatiser des connexions SSH vers plusieurs serveurs Linux en utilisant une clé privée et d'exécuter des commandes nécessitant des priviléges sudo sans mot de passe, grâce à la bonne configuration sudoers (NOPASSWD) sur les serveurs.

Cette méthode est pratique pour la supervision et l'administration automatique de serveurs distants.

ssh_mysql.py

Nous devons avant **modifier le fichier sudoer sur le serveur sql** avec `sudo visudo` pour ne pas avoir besoin de sudo pour mysql

Ajouter à la fin du fichier :

```
@includedir /etc/sudoers.d
monitor ALL=(ALL) NOPASSWD: /bin/systemctl, /bin/ls, /bin/cat, /usr/bin/tail, /bin/journalctl, /usr/bin/mysql
```

Ce script vérifie que le serveur MariaDB est accessible en SSH depuis votre station de monitoring Windows et que le service de base de données fonctionne correctement, préparant ainsi l'infrastructure pour stocker centralement tous les logs d'intrusion analysés.

ssh_mysql_error.py job5

On se connecte à l'utilisateur valdeb sur mysql **mysql -u valdeb -p**

On va sur la base de donnée **USE surveillance_logs**

On crée la table mysql_intrusions **CREATE TABLE mysql_intrusions;**

Pour voir la structure de la table :

DESCRIBE mysql_intrusions;

Field	Type	Null	Key	Default	Extra
<code>id</code>	<code>int(11)</code>	<code>NO</code>	<code>PRI</code>	<code>NULL</code>	<code>auto_increment</code>
<code>username</code>	<code>varchar(50)</code>	<code>NO</code>		<code>NULL</code>	
<code>source_host</code>	<code>varchar(100)</code>	<code>NO</code>		<code>NULL</code>	
<code>attempt_time</code>	<code>timestamp</code>	<code>YES</code>		<code>current_timestamp()</code>	
<code>log_timestamp</code>	<code>varchar(50)</code>	<code>YES</code>		<code>NULL</code>	
<code>connection_id</code>	<code>int(11)</code>	<code>YES</code>		<code>NULL</code>	
<code>using_password</code>	<code>enum('YES','NO')</code>	<code>YES</code>		<code>YES</code>	
<code>raw_log_line</code>	<code>text</code>	<code>YES</code>		<code>NULL</code>	
<code>processed_time</code>	<code>timestamp</code>	<code>YES</code>		<code>current_timestamp()</code>	

```
CREATE TABLE mysql_intrusions (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    source_host VARCHAR(100) NOT NULL,
    attempt_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    log_timestamp VARCHAR(50),
    connection_id INT,
    using_password ENUM('YES', 'NO') DEFAULT 'YES',
    raw_log_line TEXT,
    processed_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Le script va se connecter au serveur Mariadb puis récupérer les logs d'erreur d'authentification MariaDB dans journalctl, trier avec regex les infos utiles et les stocker dans la base de donnée **surveillance_logs**.

Dans le script la commande :

```
command = f"sudo journalctl -u mariadb --since='{hours_back} hours ago' --no-pager | grep -i 'Access denied'"
```

journalctl → extrait les logs bruts de MariaDB

grep → filtre pour garder seulement les erreurs d'authentification

regex → organise chaque ligne en données structurées

mysql.connector → insère en base de données

```
PS D:\PSMM_Scripts\job 6> PY .\ssh_mysql_error.py
== Script ssh_mysql_error.py - Job 06 ==
==

Serveur cible : 192.168.175.139
Base de données : surveillance_logs

[SSH] Connexion réussie vers 192.168.175.139
[CMD] sudo journalctl -u mariadb --since='24 hours ago' --no-pager | grep -i 'Access denied'
[INFO] 22 lignes de logs d'erreur récupérées

== Parsing des logs ==
[PARSE 1/22] Analyse...
  + utilisateur@localhost - 2025-09-18 11:40:20
[PARSE 2/22] Analyse...
  + root@localhost - 2025-09-18 11:40:38
[PARSE 3/22] Analyse...
  + localhost@localhost - 2025-09-18 11:47:54
[PARSE 4/22] Analyse...
  + utilisateur@localhost - 2025-09-18 11:48:07
[PARSE 5/22] Analyse...
  + utilisateur@localhost - 2025-09-18 11:48:11
[PARSE 6/22] Analyse...
  + utilisateur@localhost - 2025-09-18 11:48:16
[PARSE 7/22] Analyse...
  + valdeb@localhost - 2025-09-18 11:49:14
[PARSE 8/22] Analyse...
  + valdeb@localhost - 2025-09-18 11:49:18
[PARSE 9/22] Analyse...
  + valdeb@localhost - 2025-09-18 11:49:30
[PARSE 10/22] Analyse...
  + valdeb@localhost - 2025-09-18 11:49:38
[PARSE 11/22] Analyse...
  + valdeb@localhost - 2025-09-18 11:49:56
[PARSE 12/22] Analyse...
  + root@localhost - 2025-09-18 11:51:00
[PARSE 13/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:40:36
[PARSE 14/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:40:42
[PARSE 15/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:51:41
[PARSE 16/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:52:01
[PARSE 17/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:52:41
[PARSE 18/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:53:18
[PARSE 19/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:54:17
[PARSE 20/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:54:20
[PARSE 21/22] Analyse...
  + valdeb@localhost - 2025-09-18 14:54:51
[PARSE 22/22] Analyse...
  + valdeb@localhost - 2025-09-18 15:20:13

[RESUME] 22 intrusions parsées sur 22 lignes

== Insertion en base de données ==
[DB] Connexion à la base surveillance_logs réussie
[INSERT] utilisateur@localhost - 2025-09-18 11:40:20
```

```
monitor@letoval-sql:/var/log$ mysql -u valdeb -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 92
Server version: 10.11.14-MariaDB-0+deb12u2 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input s

MariaDB [(none)]> USE surveillance_logs;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [surveillance_logs]> SELECT username, source_host, log_timestamp
    ->     FROM mysql_intrusions
    ->     ORDER BY log_timestamp DESC
    ->     LIMIT 10;
+-----+-----+-----+
| username | source_host | log_timestamp |
+-----+-----+-----+
| têteenlair | localhost | 2025-09-18 16:01:18 |
| hacker4   | localhost | 2025-09-18 16:00:58 |
| intru4    | localhost | 2025-09-18 16:00:48 |
| intrul    | localhost | 2025-09-18 16:00:43 |
| valdeb    | localhost | 2025-09-18 15:58:33 |
| valdeb    | localhost | 2025-09-18 15:20:13 |
| valdeb    | localhost | 2025-09-18 15:20:13 |
| valdeb    | localhost | 2025-09-18 14:54:51 |
| valdeb    | localhost | 2025-09-18 14:54:51 |
| valdeb    | localhost | 2025-09-18 14:54:20 |
+-----+-----+-----+
```

```
MariaDB [surveillance_logs]> SELECT username, COUNT(*) as tentatives
    ->     FROM mysql_intrusions
    ->     GROUP BY username
    ->     ORDER BY tentatives DESC;
+-----+-----+
| username | tentatives |
+-----+-----+
| valdeb   | 31      |
| utilisateur | 8      |
| root     | 4      |
| localhost | 2      |
| têteenlair | 1      |
| hacker4  | 1      |
| intru4   | 1      |
| intrul   | 1      |
+-----+-----+
8 rows in set (0,004 sec)
```

password_manager.py

Ce script est un gestionnaire de mot de passes . Il évite que les mdp soient en clair dans les scripts que l'on crée ensuite. Les scripts, importeront la fonction **get_password** depuis **password_manager.py** pour récupérer les mots de passe déchiffrés en mémoire, sans jamais les stocker en clair dans le code source.

password_manager.py va chercher dans le **dossier .ssh**, les mdp chiffré par Fernet dans le **fichier .psmm_passwords.json** et la clé pour les déchiffrés dans le **fichier .psmm_key**

Il n'y a donc pas de mdp dans les scripts les mdp sont chiffrés dans un fichier json et la clé pour déchiffrer dans un autre dossier json.

ssh_mysql_error.py job6

Le script va se connecter au serveur Mariadb puis récupérer les logs d'erreur d'authentification MariaDB dans journalctl, trier avec regex les infos utiles et les stocker dans la base de données **surveillance_logs**.

Le premier script créé contenait le mdp de la base de donnée en clair ce qui constitue une grosse faille de sécurité, voici les étapes pour le régler avec une clé FERNET

- installation sur powershell de pip install cryptography
- créer le mdp chiffré avec :

```
python -c "from cryptography.fernet import Fernet; key = Fernet.generate_key(); f = Fernet(key); encrypted = f.encrypt(b'poi'); print('CLE:', key.decode()); print('MDP_CHIFFRE:', encrypted.decode())"
```

on obtient la clé et le mdp chiffré

```
PS D:\Scripts\PSMM_Scripts\job6\testf> python -c "from cryptography.fernet import Fernet; key = Fernet.generate_key(); f = Fernet(key); encrypted = f.encrypt(b'poi'); print('CLE:', key.decode()); print('MDP_CHIFFRE:', encrypted.decode())"
CLE: HObGoNXDeXxjTux2VHaqhR1LGE4TwgM3fRitqa4euEE=
MDP_CHIFFRE: gAAAAABo0-z1wH73-NgMuQhKZwjziiXLlHYX6qp32GZHvezERcJ0TMKHgbJhaD0LIITdHxyGPptpDhsV3THb686pWC8YxNUqA==
```

- Enregistrer la clé dans les variables d'environnement windows soit en graphique soit powershell : **setx FERNET_KEY "HOBGNXDeXxjTux2VHaqhR1LGE4TwgM3fRitqa4euEE=**

Sur le serveur web, ajouter /usr/bin/mysql à monitor dans sudo visudo permettre au script d'exécuter des commandes SQL via SSH en utilisant sudo mysql

On se connecte à l'utilisateur valdeb sur mysql mysql -u valdeb -p

On va sur la base de données USE surveillance_logs;

On vérifie le nombre total d'intrusions : DESCRIBE mysql_intrusinons;

Vérifier les intrusions des dernières 24h :

```
SELECT username, source_host, log_timestamp
```

```
FROM mysql_intrusions
```

```
WHERE log_timestamp >= DATE_SUB(NOW(), INTERVAL 24 HOUR)
```

```
ORDER BY log_timestamp DESC:
```

MariaDB [surveillance_logs]> DESCRIBE mysql_intrusions;						
Field	Type	Null	Key	Default	Extra	
id	int(11)	NO	PRI	NULL		auto_increment
username	varchar(50)	NO		NULL		
source_host	varchar(100)	NO		NULL		
attempt_time	timestamp	YES		current_timestamp()		
log_timestamp	varchar(50)	YES		NULL		
connection_id	int(11)	YES		NULL		
using_password	enum('YES', 'NO')	YES		YES		
raw_log_line	text	YES		NULL		
processed_time	timestamp	YES		current_timestamp()		

ssh_ftp_error.py job7

Le script se connecte en SSH au serveur SFTP, extrait les logs d'erreur d'authentification via journalctl, parse les informations avec des regex pour identifier les tentatives d'intrusion, puis stocke ces données structurées dans la base MariaDB pour surveillance de sécurité.

Sur le serveur web, ajouter `/bin/journalctl` à monitor dans `sudo visudo` pour autoriser la connexion au journal d'erreur

Sur le serveur SQL on peut vérifier de plusieurs manières:

mysql -u valdeb -p

USE surveillance_logs;

Compter le total des tentatives d'intrusions FTP

```
SELECT COUNT(*) as total_intrusions FROM ftp_intrusions;
```

Voir les nouvelles tentatives d'intrusion (par ordre chronologique) :

```
SELECT username_attempted, ip_source, attack_type, timestamp_attack
```

FROM ftp intrusions

ORDER BY timestamp attack DESC

LIMIT 20:

Analyser les types d'attaques :

SELECT attack type, COUNT(*) as nombre tentatives

FROM ftp intrusions

GROUP BY attack type

ORDER BY nombre_tentativas DESC;

Identifier les sources d'attaque :

SELECT *in source COUNT(*) as tentatives*

```
SELECT ip_source, COUNT(*) as countattempts,  
GROUP_CONCAT(DISTINCT username_attempted) as usernames_tested
```

FROM ftp_intrusions GROUP BY ip_source ORDER BY tentatives DESC;

ssh_web_error.pyjob8

Le script se connecte en SSH au serveur SFTP, extrait les logs d'erreur d'authentification via journalctl, parse les informations avec des regex pour identifier les tentatives d'intrusion, puis stocke ces données structurées dans la base MariaDB pour surveillance de sécurité.

- Vérification dans les logs

```
sudo grep -i "auth\|401\|403" /var/log/apache2/access.log  
sudo grep -i "auth\|password" /var/log/apache2/error.log
```

- Sur le serveur web, ajouter bin/grep à monitor dans sudo visudo pour autoriser la commande grep sans mot de pass.

- Sur le serveur SQL on peut vérifier de plusieurs manières:

```
mysql -h 192.168.175.139 -u valdeb -p surveillance_logs
```

- Analyse des données récupérées :

```
DESCRIBE web_intrusions;
```

```
SELECT * FROM web_intrusions;
```

```
SELECT * FROM web_intrusions ORDER BY timestamp_attack DESC LIMIT 10\G
```

```
MariaDB [surveillance_logs]> SELECT username_attempted, COUNT(*) as nb_tentatives, MIN(timestamp_attack) as premiere_tentative
-> FROM web_intrusions
-> WHERE DATE(timestamp_attack) = '2025-09-22'
-> GROUP BY username_attempted, HOUR(timestamp_attack), MINUTE(timestamp_attack)
-> ORDER BY premiere_tentative;
+-----+-----+-----+
| username_attempted | nb_tentatives | premiere_tentative |
+-----+-----+-----+
| anonymous          |       4        | 2025-09-22 10:11:05 |
| webadmin            |      16        | 2025-09-22 10:11:10 |
| webadmintryfake    |       8        | 2025-09-22 10:11:16 |
| intru1              |       8        | 2025-09-22 10:11:20 |
| intru2              |       8        | 2025-09-22 10:11:25 |
| hakceur              |       8        | 2025-09-22 10:11:28 |
| anonymous            |      12        | 2025-09-22 10:18:32 |
+-----+-----+-----+
7 rows in set (0,005 sec)
```

```
[RESUME] 16 intrusions parsées sur 16 lignes
== Insertion en base de données ==
[DB] Connexion à la base surveillance_logs réussie
[INSERT] anonymous@192.168.175.1 → /secure/
[INSERT] webadmin@192.168.175.1 → /secure/
[INSERT] webadmintryfake@192.168.175.1 → /secure/
[INSERT] intru1@192.168.175.1 → /secure/
[INSERT] intru2@192.168.175.1 → /secure/
[INSERT] hakceur@192.168.175.1 → /secure/
[INSERT] webadmin@192.168.175.1 → /secure/
[INSERT] anonymous@192.168.175.142 → /secure/
[INSERT] anonymous@127.0.0.1 → /secure/
[INSERT] anonymous@192.168.175.142 → /secure/
[INSERT] webadmin@192.168.175.1 → /secure/
[INSERT] webadmintryfake@192.168.175.1 → /secure/
[INSERT] intru1@192.168.175.1 → /secure/
[INSERT] intru2@192.168.175.1 → /secure/
[INSERT] hakceur@192.168.175.1 → /secure/
[INSERT] webadmin@192.168.175.1 → /secure/
== RESUME FINAL ==
Logs récupérés : 16
Intrusions parsées : 16
Enregistrements insérés : 16
```

ATTENTION

A partir de ces jobs, les scripts sont réalisés sur les VM donc sont adaptés à linux dans le but de pouvoir automatiser avec crontab et de voir aussi les différences de script.

ssh serveur mail.py job9

Le script envoie un mail à l'administrateur avec les historiques des tentatives de connexion de la veille.

Il faut créer un MDP pour l'application dans google account qui permet d'avoir un mdp à utiliser dans le script pour utiliser gmail (différent du mdp gmail) .

Attention il faut garder le mdp car il disparait et impossible de le réaficher

The screenshot shows the Google Account security interface. The left sidebar has a red box around the 'Sécurité' (Security) option. The main search bar at the top contains the text 'mot de passe applications'. Below it, a search result titled 'Mot de passe des applications' is highlighted with a red box. The page content discusses application passwords and includes a link to 'En savoir plus'.

← Mots de passe des applications

Les mots de passe d'application vous permettent de vous connecter à votre compte Google sur des applis et des services plus anciens, non compatibles avec les normes de sécurité les plus récentes.

Les mots de passe d'application sont moins sécurisés que les applis et services à jour qui utilisent les normes de sécurité les plus récentes. Avant de créer un mot de passe d'application, vous devez vérifier si votre appli en a besoin pour établir la connexion.

[En savoir plus](#)

```
PS D:\Scripts\PSMM_Scripts\job9> python -c "import os; from cryptography.fernet import Fernet; key = os.getenv('FERNET_KEY'); f = Fernet(key.encode()); gmail_pwd = 'ixlaunqzufkykrz'; print('Nouveau GMAIL chiffré:', f.encrypt(gmail_pwd.encode()).decode())"
Nouveau GMAIL chiffré: AAAAMAA=2827RBRZQ-ZoTzL7fa-5d1717A1WmU?2hA-1w-GenUVw-1c5E5v-1udC96gat-gw-1w-1w-TuSMNC4-201Nwv-VT-0HylVgJPCid-y92cyzP1DyH-
```

```
PS D:\PSMM_Scripts\job09> py .\ssh_serv_mail_mdphide.py
== Script ssh_serveur_mail.py - Job 09 SÉCURISÉ ==
 
⚠ == SAISIS SÉCURISÉE - MOT DE PASSE MySQL ==
Le mot de passe ne s'affichera pas à l'écran
-----
Mot de passe MySQL (utilisateur valdebf):
 Mot de passe MySQL saisi avec succès

Base de données : surveillance_logs sur 192.168.175.139
Email destinataire : valentin.rossi@laplateforme.io
Période : 3 derniers jours

[ETAPÉ 1] Récupération des données depuis la base...
[DB] Connexion réussie à surveillance_logs
[INFO] Récupération des données depuis le 2025-09-19 00:00:00
[MYSQL] 5 groupes d'intrusions récupérées
[FTP] 14 groupes d'intrusions récupérées
[WEB] 8 groupes d'intrusions récupérées

[ETAPÉ 2] Génération du rapport HTML...
[HTML] Rapport généré (1541 caractères)

[ETAPÉ 3] Envoi du rapport par email...
[EMAIL] Préparation du message...
[EMAIL] De: valentin.rossi@laplateforme.io
[EMAIL] Vers: valentin.rossi@laplateforme.io
[SMTP] Connexion Gmail réussie
[EMAIL]  Email envoyé avec succès!

== RESUME FINAL ==
 Données récupérées avec succès
 Rapport HTML généré
 Email envoyé à valentin.rossi@laplateforme.io
 Job 09 terminé avec succès!
```

ssh_cron_backup.py job 10

Le script fait une sauvegarde locale des données de la base de données, horodatée, avec conservation des sept dernières sauvegardes, avec une planification toutes les 3h.
Sauvegarde toutes les 3 heures // Conservation de 7 sauvegardes rotatives // Compression automatique (90% de gain) // Logs complets et notifications d'erreur

- **Créer le répertoire pour le script** `mkdir -p /home/monitor/scripts`
- **Créer le script** `nano ssh_cron_backup.py`
- **Rendre le scscript exécutable** `chmod +x /home/monitor/scripts/ssh_cron_backup.py`
- **Créer le répertoire de sauvegarde**

`sudo mkdir -p /home/monitor/backups`

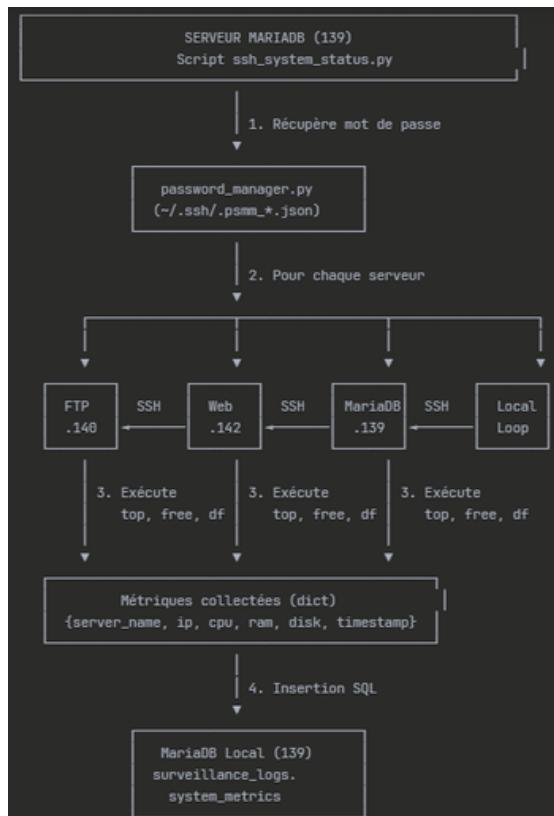
`sudo chown monitor:monitor /home/monitor/backups`

- **Configurer le crontab** `crontab -e`

`0 */3 * * * cd /home/monitor/scripts && python3 ssh_cron_backup>`

```
# Job 10 PSMM - Sauvegarde automatique toutes les 3 heures
0 */3 * * * cd /home/monitor/scripts && python3 ssh_cron_backup.py >> psmm_backup.log 2>&1
```

ssh_system_status.py job 11



On doit reférer la paire de clé SSH car on à changé d emachine monitor (passé de mon PC au serveur SQL)

Sur le serv SQL (monitor) `ssh-keygen -t ed25519 -f ~/ssh/id_ed25519 -N ""`
puis on copie la clé publique avec `cat ~/ssh/id_ed25519.pub` sur le serv ftp et web ainsi que SQL dans leur fichier `nano ~/ssh/authorized_keys`

- **Créer la table sur sql**

`sudo mysql -u valdeb -p surveillance_logs`

```

CREATE TABLE IF NOT EXISTS system_metrics (
  id INT AUTO_INCREMENT PRIMARY KEY,
  server_name VARCHAR(50) NOT NULL,
  server_ip VARCHAR(15) NOT NULL,
  cpu_usage FLOAT NOT NULL,
  ram_usage FLOAT NOT NULL,
  disk_usage FLOAT NOT NULL,
  timestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
  INDEX idx_timestamp (timestamp),
  INDEX idx_server (server_name)
);
  
```

- **Vérifier la table a bien été crée `DESCRIBE system_metrics;`**
- **Ensuite on exit pour créer le script `ssh_system_status.py` et donner les droits d'executions avec `chmod +x ssh_system_status.py` aa**
- **on lance le script qui fonctionne et on vérifie sur la base de donnée avec**
- **`SELECT * FROM system_metrics ORDER BY timestamp DESC LIMIT 10;`**

MariaDB [surveillance_logs]> SELECT * FROM system_metrics ORDER BY timestamp DESC LIMIT 10;						
id	server_name	server_ip	cpu_usage	ram_usage	disk_usage	timestamp
2	web	192.168.175.142	0	0	29	2025-10-03 11:01:03
3	mariadb	192.168.175.139	0	27.99	42	2025-10-03 11:01:03
1	ftp	192.168.175.140	0	0	28	2025-10-03 11:01:02

ssh_system_mail.py job 12

```
SELECT * FROM system_metrics ORDER BY timestamp DESC LIMIT 3;
```

```
crontab -e
```

```
*/5 * * * * /usr/bin/python3 /home/monitor/scripts/ssh_system_mail.py >>
/home/monitor/scripts/logs/ssh_system_mail.log 2>&1
```

Toutes les 5 minutes, exécute le script Python de monitoring système et écris tous les messages (résultats et erreurs) dans le fichier de log.

```
mkdir -p ~/scripts/logs
```

