TP : Sécuriser un serveur SSH avec Fail2Ban face à une attaque brute-force (Patator)

1. Objectif du TP

Dans ce TP, j'ai mis en place une protection contre les attaques par force brute sur un serveur SSH en utilisant **Fail2Ban**. Pour tester l'efficacité de cette protection, j'ai utilisé l'outil **Patator** pour simuler une attaque avec des identifiants aléatoires. L'objectif était de voir si l'IP attaquante était bien bannie automatiquement après plusieurs tentatives échouées.

2. Lexiques

Fail2Ban : outil de protection contre les attaques répétées (force brute) en analysant les logs et en bannissant les IP suspectes.

OpenSSH : suite logicielle permettant des connexions réseau sécurisées via SSH.

Patator : outil en ligne de commande permettant de lancer des attaques de type brute force à l'aide d'une wordlist (dictionnaire).

3. Prérequis

Avant de commencer, j'ai utilisé deux machines virtuelles sous Ubuntu :

- **Machine 1** : serveur SSH avec Fail2Ban.
- **Machine 2** : attaquant avec Patator.

J'ai aussi vérifié que la machine attaquante avait une **adresse IP différente** de la cible. (solution : clone VM)

4. Installation des outils :

4.1 Mettre à jour les paquets :

```
sudo apt-get update
```

4.2 Installer Fail2ban :

```
sudo apt install Fail2ban
```

4.3 Activer et vérifier le service :

```
systemctl start fail2ban
systemctl enable fail2ban
systemctl status fail2ban
```

4.4 Installer le serveur ssh (si nécessaire) :

```
sudo apt-get install openssh-server
sudo systemctl start ssh
sudo systemctl status ssh
```

5. Configuration de Fail2Ban

Fail2Ban fonctionne avec des " *jails* " (prisons) qui analysent les fichiers de log pour détecter les connexions suspectes.

5.1 Fichier de configuration

Copier le fichier de base :

```
sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Éditer le fichier :

```
sudo nano /etc/fail2ban/jail.local
```

Activer la surveillance du SSH :

```
[sshd]
enabled = true
port = ssh
logpath = /var/log/auth.log
maxretry = 3
bantime = 600
```

5.2 Redémarrage fail2ban :

```
sudo systemctl restart fail2ban
```

5.3 Vérifier la jail SSH est actives :

```
sudo fail2ban-client status
sudo fail2ban-client status sshd
```

6. Simulation d'une attaque avec Patator

6.1 Installation de Patator (machine attaquante)

```
sudo apt install patator
```

6.2 Préparation des fichiers d'attaque

Créer un fichier **user.txt** avec un identifiant :

```
echo "root" > user.txt
```

Créer un fichier **pass.txt** avec quelques mots de passe :

```
echo -e "1234\nadmin\npassword" > pass.txt
```

6.3 Lancer l'attaque

Lancer Patator avec la commande suivante :

```
patator ssh_login host=192.168.60.128 user=FILE0 password=FILE1 0=user.txt
1=pass.txt -x ignore:mesg='Authentication failed.'
```

Remplacer l'adresse IP par celle de la machine cible

7. Résultat de l'attaque

Côté attaquant

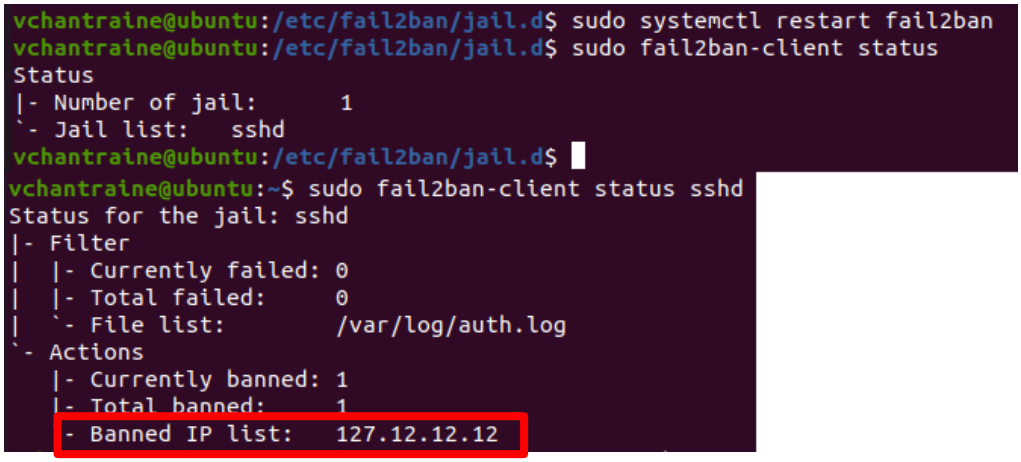
Après plusieurs tentatives, j'obtiens des **authentication timeout**, ce qui signifie que l'IP a été bannie.

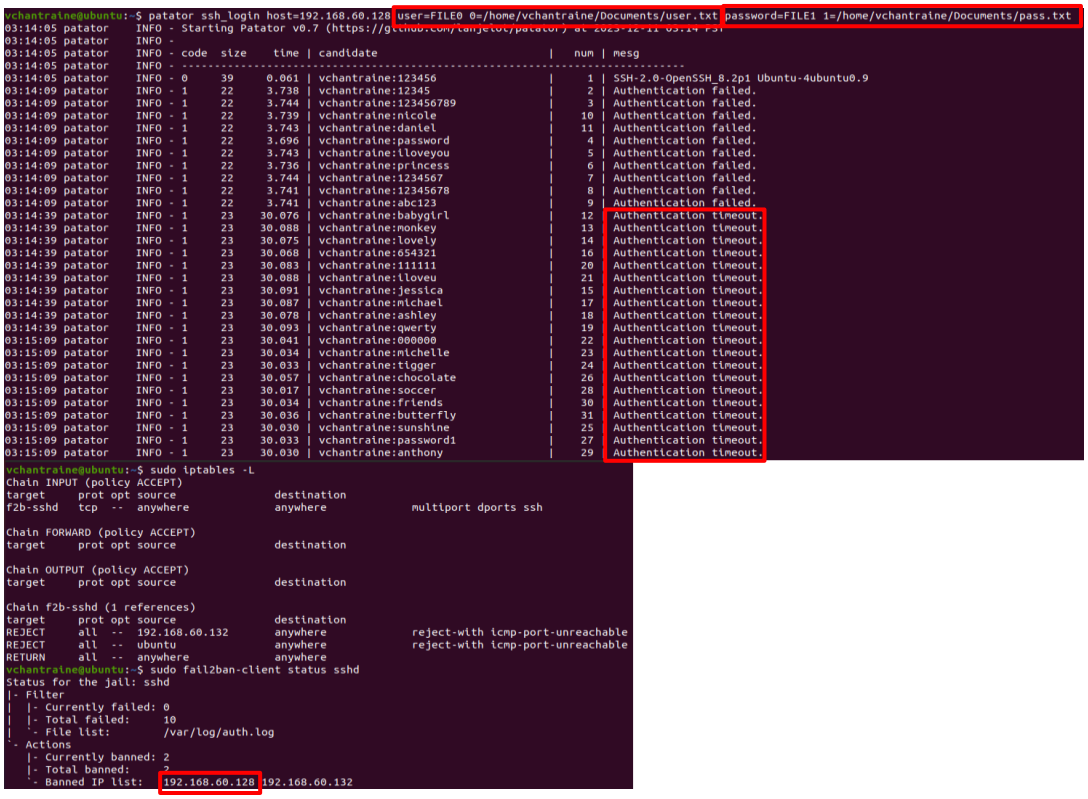
Côté serveur (défenseur)

J'ai vérifié avec :

```
sudo fail2ban-client status sshd
```

On peut voir l'IP de l'attaquant bannie dans la liste : 127.12.12.12





Est le résultat du côté défenseur où l'on peut observer l'adresse IP de l'attaquant banni.

8. Conclusion

Ce TP m'a permis de tester un mécanisme de sécurité simple mais efficace pour limiter les attaques SSH par force brute. Fail2Ban détecte automatiquement les tentatives répétées échouées et bannit l'IP source, ce qui ralentit fortement les attaques par dictionnaire. C'est une solution légère et facile à mettre en place sur un serveur.

9. Sources utiles

- https://doc.ubuntu-fr.org/fail2ban
- https://github.com/lanjelot/patator