OPENCLASSROOMS

Installez un serveur Linux et sécurisez son accès distant

Exemple de corrigé pour l'activité "À vous de jouer"

1. Vérification de l'état actuel du pare-feu UFW

Sur le serveur, vérifiez si UFW est disponible et consultez son état actuel.

```
None
sudo ufw status
```

Résultat attendu si UFW est installé mais inactif :

```
None
Status: inactive
```

➜ UFW est bien présent, mais le pare-feu n'est pas encore actif. C'est l'état attendu avant la mise en place des règles de filtrage.

Si la commande n'est pas reconnue, installez UFW :

```
None
sudo apt install ufw
```

Résultat attendu :

```
None
Setting up ufw ...
```

➜ UFW est maintenant installé et prêt à être configuré.

2. Autorisation de l'accès SSH

Avant d'activer le pare-feu, autorisez explicitement l'accès SSH.

```
None
sudo ufw allow ssh
```

Résultat attendu :

```
None
Rules updated

Rules updated (v6)
```

➜ Le trafic SSH est autorisé. Cette étape est indispensable pour conserver votre accès distant au serveur après l'activation du pare-feu.

3. Activation du pare-feu et vérification des règles

Activez UFW.

```
None
sudo ufw enable
```

Résultat attendu :

```
None
Command may disrupt existing ssh connections. Proceed with
operation (y|n)?
```

Répondez :

```
None
y
```

Résultat attendu :

```
None
Firewall is active and enabled on system startup
```

Vérifiez ensuite les règles actives.

```
None
sudo ufw status verbose
```

Résultat attendu :

```
None
Status: active


To                         Action      From

--                         ------      ----

22/tcp                     ALLOW IN    Anywhere

22/tcp (v6)                ALLOW IN    Anywhere (v6)
```

➜ Le pare-feu est actif. Les connexions entrantes non autorisées sont bloquées, tandis que l'accès SSH reste autorisé.

4. Vérification que l'accès SSH reste fonctionnel

Sans fermer votre session actuelle, ouvrez un nouveau terminal sur votre machine cliente et testez une nouvelle connexion SSH.

```
None
ssh admin@192.168.1.50
```

Remplacez `192.168.1.50` par l'adresse IP de votre serveur.

Résultat attendu :

```
None
admin@serveur:~$
```

➜ L'accès SSH reste fonctionnel après l'activation du pare-feu. Vous ne vous êtes pas coupé votre accès distant au serveur.

5. Installation et vérification de Fail2Ban

Installez Fail2Ban sur le serveur.

```
None
sudo apt install fail2ban
```

Résultat attendu :

```
None
Setting up fail2ban ...
```

Vérifiez ensuite l'état du service.

```
None
systemctl status fail2ban
```

Résultat attendu :

```
None
Active: active (running)
```

➜ Fail2Ban est installé et son service fonctionne en arrière-plan. Dans une machine virtuelle de pratique, vous n'observerez pas forcément d'attaques réelles, mais l'outil est prêt à surveiller les tentatives répétées.

Vous pouvez aussi consulter son état général :

```
None
sudo fail2ban-client status
```

Résultat attendu, selon la configuration :

```
None
Status

|- Number of jail: ...
```

```
`- Jail list: ...
```

➜ Fail2Ban utilise des règles appelées *jails* pour surveiller certains services et appliquer des blocages temporaires en cas de comportements suspects.

6. Création d'un fichier ou dossier de test

Sur votre machine locale, créez un fichier de test à transférer vers le serveur.

```
None
echo "Test de transfert sécurisé" > test-transfert.txt
```

Vérifiez que le fichier existe bien.

```
None
ls -l test-transfert.txt
```

Résultat attendu :

```
None
-rw-r--r-- 1 user user ... test-transfert.txt
```

➜ Le fichier de test est prêt. Il va permettre de vérifier un transfert sécurisé entre votre machine locale et le serveur.

Vous pouvez aussi créer un dossier de test pour la synchronisation avec `rsync`.

```
None
mkdir dossier-test

echo "Fichier synchronisé avec rsync" >
dossier-test/fichier-rsync.txt
```

Résultat attendu :

```
None
dossier-test/

└── fichier-rsync.txt
```

➜ Le dossier de test est prêt pour la synchronisation.

7. Test des transferts sécurisés avec SFTP et rsync

Commencez par tester un transfert interactif avec `sftp`.

```
None
sftp admin@192.168.1.50
```

Résultat attendu :

```
None
Connected to 192.168.1.50.

sftp>
```

Envoyez le fichier vers le serveur.

```
None
put test-transfert.txt
```

Résultat attendu :

```
None
Uploading test-transfert.txt to
/home/admin/test-transfert.txt
```

Quittez ensuite SFTP.

```
None
bye
```

➜ Le fichier a été transféré vers le serveur via une connexion chiffrée reposant sur SSH.

Testez maintenant une synchronisation avec `rsync`.

```
None
rsync -av dossier-test/
admin@192.168.1.50:/home/admin/dossier-test/
```

Résultat attendu :

```
None
sending incremental file list

./

fichier-rsync.txt



sent ... bytes  received ... bytes  ... bytes/sec

total size is ...  speedup is ...
```

➜ Le dossier local a été synchronisé vers le serveur. `rsync` transfère efficacement les fichiers via SSH.

Pour vérifier la présence des fichiers côté serveur, connectez-vous en SSH :

```
None
ssh admin@192.168.1.50
```

Puis listez les fichiers reçus :

```
None
ls -l

ls -l dossier-test
```

Résultat attendu :

```
None
test-transfert.txt

dossier-test/
```

Et dans le dossier :

```
None
fichier-rsync.txt
```

➜ Les fichiers ont bien été reçus sur le serveur. Les transferts ont été réalisés sans utiliser de protocole non chiffré comme FTP.