Démonstration complète et réaliste : Contrôle d'accès avec les modules `mod_authz*`

Dans cette démonstration, nous allons configurer un serveur Apache HTTPD pour gérer le contrôle d'accès à une application web en utilisant plusieurs modules `mod_authz*`. La configuration inclura des restrictions basées sur :

1. **Adresses IP (`mod_authz_host`)**
2. **Utilisateurs (`mod_authz_user`)**
3. **Groupes (`mod_authz_groupfile`)**
4. **Combinaisons avancées (`mod_authz_core`)**

Contexte du scénario

Configuration cible :

1. **Application web protégée** située dans `/var/www/secure-app`.
2. **Accès limité** :
   - Aux utilisateurs authentifiés appartenant au groupe `admin` ou `developer`.
   - À des IP spécifiques pour les utilisateurs non authentifiés (réseau interne : `192.168.1.0/24`).
3. **Protection HTTPS** via SSL.
4. **Fichier de groupes utilisateurs** : `/etc/apache2/groups`.

Prérequis :

1. Apache HTTPD 2.4 installé avec les modules nécessaires (`mod_authz_host`, `mod_authz_user`, `mod_authz_groupfile`, `mod_authz_core`).
2. Configuration SSL active.

Étape 1 : Activer les modules nécessaires

1. **Vérifier les modules disponibles** :

```
apache2ctl -M
```

2. **Activer les modules requis** (si non activés) :

```
sudo a2enmod authz_host authz_user authz_groupfile authz_core ssl
sudo systemctl restart apache2
```

Étape 2 : Configurer les utilisateurs et groupes

1. **Créer un fichier de mots de passe utilisateur**

- Utilisez `htpasswd` pour ajouter des utilisateurs avec des mots de passe :

```
sudo htpasswd -c /etc/apache2/.htpasswd john
sudo htpasswd /etc/apache2/.htpasswd alice
```

2. **Créer un fichier de groupes**

- Fichier : `/etc/apache2/groups`

```
admin: john
developer: alice
```

3. **Vérifiez les permissions des fichiers**

- Assurez-vous que le serveur Apache peut lire les fichiers :

```
sudo chown root:www-data /etc/apache2/.htpasswd
/etc/apache2/groups
sudo chmod 640 /etc/apache2/.htpasswd /etc/apache2/groups
```

Étape 3 : Configurer le Virtual Host

1. **Créer un Virtual Host sécurisé**

- Fichier : `/etc/apache2/sites-available/secure-app.conf`

```
<VirtualHost *:443>
    ServerName secure.example.com
    DocumentRoot "/var/www/secure-app"

    SSLEngine On
    SSLCertificateFile "/etc/ssl/certs/example.com.crt"
    SSLCertificateKeyFile "/etc/ssl/private/example.com.key"

    # Configuration d'accès au répertoire sécurisé
    <Directory "/var/www/secure-app">
        Options Indexes FollowSymLinks
        AllowOverride None

        # Règles de contrôle d'accès
        AuthType Basic
        AuthName "Restricted Access"
        AuthUserFile /etc/apache2/.htpasswd
        AuthGroupFile /etc/apache2/groups
```

```
        # Accès combiné
        <RequireAny>
            # Autoriser les utilisateurs du groupe admin
            Require group admin

            # Autoriser les utilisateurs du groupe developer
            Require group developer

            # Autoriser les adresses IP internes
            Require ip 192.168.1.0/24
        </RequireAny>
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/secure-app-error.log
    CustomLog ${APACHE_LOG_DIR}/secure-app-access.log combined
</VirtualHost>
```

2. **Activer le site et redémarrer Apache**

```
sudo a2ensite secure-app.conf
sudo systemctl reload apache2
```

Étape 4 : Tester la configuration

a. Vérifier l'accès utilisateur

1. Accédez à `https://secure.example.com`.
2. Résultat attendu :
   - Les utilisateurs non authentifiés doivent recevoir une invite de connexion.
   - Seuls les utilisateurs `john` et `alice` peuvent se connecter.
   - Les utilisateurs des groupes `admin` et `developer` ont accès.

b. Vérifier l'accès IP

1. Depuis une machine avec l'IP dans la plage `192.168.1.0/24`, accédez au site.
2. Résultat attendu :
   - Aucun mot de passe n'est requis.
   - L'accès est autorisé directement.

c. Logs

- Consultez les journaux pour vérifier les connexions et les tentatives d'accès non autorisées :

```
tail -f /var/log/apache2/secure-app-access.log
tail -f /var/log/apache2/secure-app-error.log
```

Étape 5 : Scénario avancé (blocage et redirections)

Ajouter une règle pour bloquer certaines IPs

Modifiez la configuration pour bloquer explicitement une plage IP :

```
<Directory "/var/www/secure-app">
    <RequireAll>
        Require not ip 10.0.0.0/8
        RequireAny
            Require group admin
            Require ip 192.168.1.0/24
    </RequireAll>
</Directory>
```

Ajouter une redirection pour les utilisateurs non autorisés

Ajoutez une directive pour rediriger les accès non autorisés vers une page d'erreur personnalisée :

```
ErrorDocument 403 /error_pages/403.html
```

Étape 6 : Résumé des fichiers

Fichier	Description
`/etc/apache2/.htpasswd`	Fichier contenant les utilisateurs et mots de passe.
`/etc/apache2/groups`	Fichier définissant les groupes d'utilisateurs.
`/etc/apache2/sites-available/secure-app.conf`	Configuration du Virtual Host sécurisé.

Conclusion

Cette démonstration montre comment utiliser les modules `mod_authz*` pour :

1. Gérer les utilisateurs et groupes via des fichiers externes.
2. Combiner des règles d'accès basées sur des groupes, utilisateurs et adresses IP.
3. Protéger les ressources avec des règles avancées et sécurisées.