Authentifier les Utilisateurs avec Apache

Apache HTTPD offre plusieurs méthodes pour authentifier les utilisateurs avant qu'ils n'accèdent à une ressource protégée. Ces mécanismes d'authentification peuvent être configurés à l'aide des modules intégrés ou en combinant Apache avec des systèmes externes comme LDAP, bases de données, ou applications tierces.

---

1. Concepts Clés

1. **Authentification** : Vérifie l'identité d'un utilisateur.
2. **Autorisation** : Détermine si l'utilisateur authentifié a les droits nécessaires pour accéder à une ressource.
3. **Modules Apache Utilisés** :
   - `mod_auth_basic` : Authentification HTTP de base.
   - `mod_auth_digest` : Authentification HTTP avec chiffrement (Digest).
   - `mod_authn_file` : Authentification basée sur un fichier.
   - `mod_authn_dbd` : Authentification avec une base de données.
   - `mod_authnz_ldap` : Authentification via un annuaire LDAP.

---

2. Authentification de Base (HTTP Basic Authentication)

Principe

- Les identifiants (nom d'utilisateur et mot de passe) sont transmis en clair (base64 encodé, mais non chiffré).
- Utilisation idéale pour des environnements internes ou des connexions sécurisées (HTTPS).

Configuration

1. **Créer un Fichier d'Utilisateurs**
   Utilisez la commande `htpasswd` pour créer un fichier contenant les noms d'utilisateurs et mots de passe.

   ```
   sudo htpasswd -c /etc/apache2/.htpasswd user1
   ```

   - `-c` : Crée un nouveau fichier.
   - Pour ajouter un utilisateur, utilisez la commande sans `-c` :

     ```
     sudo htpasswd /etc/apache2/.htpasswd user2
     ```

2. **Configurer le VirtualHost**
   Ajoutez les directives suivantes dans la configuration du site ou dans un fichier `.htaccess` :

```
<Directory /var/www/html/secure>
    AuthType Basic
    AuthName "Restricted Area"
    AuthUserFile /etc/apache2/.htpasswd
    Require valid-user
</Directory>
```

3. **Redémarrez Apache**

```
sudo systemctl restart apache2
```

Test

- Accédez à `http://example.com/secure`.
- Une boîte de dialogue s'affiche pour demander les identifiants.

---

3. Authentification Digest (HTTP Digest Authentication)

Principe

- Les identifiants sont chiffrés avant d'être transmis, offrant une meilleure sécurité que la méthode de base.

Configuration

1. **Créer un Fichier Digest**
   Utilisez la commande `htdigest` :

```
sudo htdigest -c /etc/apache2/.htdigest "Restricted Area" user1
```

2. **Configurer le VirtualHost**

```
<Directory /var/www/html/secure>
    AuthType Digest
    AuthName "Restricted Area"
    AuthUserFile /etc/apache2/.htdigest
    Require valid-user
</Directory>
```

3. **Redémarrez Apache**

```
sudo systemctl restart apache2
```

Test

- Accédez à la ressource protégée. Les identifiants sont chiffrés lors de la transmission.

---

4. Authentification via LDAP

Principe

Apache peut se connecter à un annuaire LDAP pour valider les identifiants.

Modules Nécessaires

Activez les modules :

```
sudo a2enmod authnz_ldap ldap
sudo systemctl restart apache2
```

Configuration

1. **Configurer le VirtualHost**

```
<Directory /var/www/html/secure>
    AuthType Basic
    AuthName "LDAP Authentication"
    AuthBasicProvider ldap
    AuthLDAPURL
"ldap://ldap.example.com/ou=users,dc=example,dc=com?uid"
    AuthLDAPBindDN "cn=admin,dc=example,dc=com"
    AuthLDAPBindPassword "admin_password"
    Require valid-user
</Directory>
```

2. **Explications** :
   - `AuthLDAPURL` : Spécifie l'URL de connexion à l'annuaire LDAP.
   - `AuthLDAPBindDN` : DN de l'utilisateur qui se connecte à LDAP pour valider les identifiants.
   - `Require valid-user` : Tous les utilisateurs valides sont autorisés.

Test

- Accédez à la ressource protégée. Apache interagit avec l'annuaire LDAP pour valider les utilisateurs.

---

5. Authentification avec Base de Données

Principe

Apache interagit avec une base de données (MySQL, PostgreSQL, etc.) pour valider les identifiants.

Modules Nécessaires

Activez les modules :

```
sudo a2enmod authn_dbd dbd
sudo systemctl restart apache2
```

Configuration

1. **Configurer le Backend DBD**
   Ajoutez la configuration globale dans `apache2.conf` :

   ```
   DBDriver mysql
   DBDParams "host=127.0.0.1 dbname=mydb user=myuser pass=mypass"
   DBDMin 4
   DBDKeep 8
   DBDMax 20
   DBDExptime 300
   ```

2. **Configurer le VirtualHost**

   ```
   <Directory /var/www/html/secure>
       AuthType Basic
       AuthName "Database Authentication"
       AuthBasicProvider dbd
       AuthDBDUserPWQuery "SELECT password FROM users WHERE username =
   %s"
       Require valid-user
   </Directory>
   ```

Explications

- `DBDParams` : Connexion à la base de données.
- `AuthDBDUserPWQuery` : Requête SQL pour valider les identifiants.

6. Authentification via OpenID Connect (OIDC)

Principe

Apache peut interagir avec des fournisseurs d'identité (Keycloak, Google, etc.) via OpenID Connect pour authentifier les utilisateurs.

Modules Nécessaires

Activez le module :

```
sudo a2enmod auth_openidc
sudo systemctl restart apache2
```

Configuration

1. Installez les dépendances nécessaires :

   ```
   sudo apt install libapache2-mod-auth-openidc
   ```

2. Configurez le VirtualHost :

   ```
   <Location /secure>
       AuthType openid-connect
       Require valid-user
       OIDCProviderMetadataURL https://accounts.google.com/.well-
   known/openid-configuration
       OIDCClientID your_client_id
       OIDCClientSecret your_client_secret
       OIDCRedirectURI https://example.com/secure/redirect_uri
   </Location>
   ```

3. **Explications** :

   - `OIDCProviderMetadataURL` : URL du fournisseur OpenID Connect.
   - `OIDCClientID` et `OIDCClientSecret` : Identifiants pour interagir avec le fournisseur.

7. Logs et Débogage

1. **Activer les Logs pour Authentification**
   Ajoutez dans la configuration Apache :

```
LogLevel authn_core:debug auth_basic:debug
```

2. **Consultez les Logs**
   Vérifiez les fichiers de log pour détecter les problèmes :

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

---

8. Bonnes Pratiques

1. **Utilisez HTTPS** :
   - Protégez les identifiants avec un chiffrement HTTPS.
2. **Minimisez les Permissions** :
   - Limitez l'accès uniquement aux ressources nécessaires.
3. **Combinez Authentification et Autorisation** :
   - Utilisez `Require` pour restreindre l'accès à des utilisateurs ou groupes spécifiques.
4. **Surveillez les Logs** :
   - Détectez les tentatives de connexion non autorisées.

---

9. Exemple Complet : Authentification de Base avec HTTPS

```
<VirtualHost *:443>
    ServerName example.com

    SSLEngine On
    SSLCertificateFile /etc/letsencrypt/live/example.com/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/example.com/privkey.pem

    <Directory /var/www/html/secure>
        AuthType Basic
        AuthName "Restricted Area"
        AuthUserFile /etc/apache2/.htpasswd
        Require valid-user
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/auth_error.log
    CustomLog ${APACHE_LOG_DIR}/auth_access.log combined
</VirtualHost>
```

Redémarrez Apache et testez.