

Restriction de l'accès utilisateur

Apache offre de nombreuses possibilités de restrictions de l'accès utilisateur et il existe de nombreux moyens de gérer l'accès aux données. Il est à noter que ce paragraphe ne traite que de méthodes d'authentification des utilisateurs, mais n'apporte aucune confidentialité. C'est-à-dire que l'accès aux données pourra être soumis à authentification, mais que les pages web circuleront en clair entre le serveur et le navigateur. Si on souhaite la confidentialité des échanges, il faudra utiliser le protocole SSL vu plus loin.

1. Restriction de l'accès aux pages web

a. Déclaration du répertoire à protéger

La restriction se fait pour un répertoire dont le contenu n'est visible qu'après authentification. Si on souhaite restreindre l'accès à un site complet, il suffit de restreindre l'accès au répertoire racine du contenu web. Toutefois, l'usage veut qu'une page d'accueil soit disponible à tous, et que toute navigation ultérieure soit soumise à authentification.

Déclaration d'une section de répertoire dans apache2.conf

```
<Directory répertoire>
...
</Directory>
```

Où *répertoire* représente le chemin absolu du répertoire dont il faut protéger l'accès. Notez les points de suspension qui représentent les directives spécifiques qui seront appliquées au contenu de ce répertoire.

b. Directives d'authentification

Dans la section de répertoire soumis à authentification, il faut ensuite ajouter toutes les directives nécessaires selon le mode d'authentification choisi. Certaines de ces directives se retrouveront dans la plupart des circonstances.

Demande d'authentification

```
AuthName "texte"
Require valid-user
Directive_auth paramètre_auth
```

Fichier apache2.conf : demande d'authentification pour un répertoire	
AuthName	Définit le titre de la boîte de dialogue qui imposera une authentification à l'utilisateur.
texte	Titre de la boîte de dialogue qui imposera une authentification à l'utilisateur.
Require valid-user	Directive imposant un fonctionnement spécifique avec son paramètre valid-user qui exige que l'utilisateur soit correctement authentifié.
Directive_auth	La ou les directives d'authentification en fonction de la méthode choisie.

Nous avons employé ici le paramètre **valid-user** pour indiquer que tout utilisateur authentifié peut accéder aux données protégées. On aurait aussi pu employer **user x** ou **group y** pour limiter l'accès à un utilisateur ou aux membres d'un groupe.

2. Authentification locale

a. Création d'une base de compte locale

La création d'un fichier de comptes locaux pour l'authentification des visiteurs apache est une façon simple et

courante de gérer les identités. Il s'agit d'un fichier unique contenant une ligne par utilisateur déclaré. Cette méthode convient donc bien dans le cadre d'un nombre limité d'utilisateurs.

La commande **htpasswd** permet de créer la base de compte puis de l'alimenter.

Création du premier compte utilisateur

```
htpasswd -c fichier nom_utilisateur
```

Ajout de compte utilisateur

```
htpasswd fichier nom_utilisateur
```

Suppression d'un compte utilisateur

```
htpasswd -D fichier nom_utilisateur
```

Commande htpasswd : options et paramètres	
-c	Nécessaire si le fichier n'existe pas encore. Si le fichier existe déjà, il est écrasé.
fichier	Le fichier contenant la base de compte.
nom_utilisateur	Le nom du compte créé.
-D	Supprime l'utilisateur dont le nom est donné en référence.

Exemple de fichier de mot de passe

Le fichier affiche sur chaque ligne le nom du compte utilisateur et son mot de passe crypté.

```
alpha:~# cat httpmdp
toto:x4OpUoo9KBR1o
titi:o9zeMsxnhS45M
tata:hQcfUWksuIAmk
alpha:~#
```

b. Chargement des modules d'authentification

Certains modules doivent être chargés pour que nos directives soient reconnues. Il faudra charger au minimum le module **auth_basic** pour permettre l'authentification par fichier local, le module **authn_file** pour gérer cette authentification, et enfin le module **authz_user**, qui gère l'autorisation de l'accès aux pages protégées. Cette profusion de modules peut inquiéter, mais un minimum de rigueur rend les choses plus faciles : pour chacune des directives employées, la documentation en ligne précisera systématiquement quels modules doivent être chargés.

Chargement des modules

Les trois modules sont nécessaires à l'authentification des utilisateurs.

```
LoadModule auth_basic_module /chemin/mod_auth_basic.so
LoadModule authn_file_module /chemin/mod_authn_file.so
LoadModule authz_user_module /chemin/mod_authz_user.so
```

c. Configuration de l'authentification locale

Dans la section de répertoire soumis à authentification, il faudra ensuite ajouter les directives nécessaires à l'authentification locale.

Directives pour authentification locale

```
AuthType Basic
AuthUserFile fichier
```

Où *fichier* représente le fichier contenant la base de compte utilisée pour l'authentification avec les mots de passe des utilisateurs.

Exemple de fichier apache2.conf avec authentification

```
ServerRoot /etc/apache2
User www-data
Group www-data
ErrorLog /var/log/apache2/error.log
Listen 80
DocumentRoot /var/www

LoadModule dir_module /usr/lib/apache2/modules/mod_dir.so
DirectoryIndex index.html

LoadModule auth_basic_module /usr/lib/apache2/modules/mod_auth_basic.so
LoadModule authn_file_module /usr/lib/apache2/modules/mod_authn_file.so
LoadModule authz_user_module /usr/lib/apache2/modules/mod_authz_user.so
<Directory /var/www>
    AuthType basic
    AuthUserFile /root/httpmdp
    AuthName "Veuillez vous identifier"
    Require valid-user
</Directory>
```

3. Authentification par annuaire LDAP

Il existe plusieurs moyens pour exploiter un annuaire LDAP en tant que base d'authentification. La configuration ci-dessous est donc donnée à titre d'exemple.

a. Vérification de disponibilité des informations de l'annuaire

Considérons que nous disposons d'un annuaire LDAP avec les caractéristiques suivantes :

- Adresse IP : 192.168.1.11
- Contexte des comptes utilisateurs : ou=users,dc=annu,dc=fr
- Nom distinctif de l'administrateur : cn=admin,dc=annu,dc=fr

Exemple d'interrogation de l'annuaire

Il est vivement recommandé de vérifier que l'annuaire est bien en ligne et accessible avec les bonnes informations (adresse IP et contexte LDAP) avant de configurer l'authentification LDAP pour une application quelle qu'elle soit.

```
toto@serveur# ldapsearch -x -D cn=admin,dc=annu,dc=fr -W -h
192.168.1.11 -b ou=users,dc=annu,dc=fr -s sub ObjectClass=*
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <dc=annu,dc=fr> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
# annu.fr
dn: dc=annu,dc=fr
objectClass: domain
```

```
dc: annu
(...)
# toto, users.annu.fr
dn: uid=toto,ou=users,dc=annu,dc=fr
objectClass: top
objectClass: posixAccount
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
(...)
toto@serveur#
```

b. Chargement des modules nécessaires

Les modules nécessaires à l'authentification LDAP sont les suivants :

- auth_basic
- authn_file
- authz_user
- authnz_ldap

c. Configuration de l'authentification

Il faudra ensuite utiliser les directives d'authentification dans une section de répertoire. Le point d'orgue de la configuration sera déclaration de la directive **AuthLDAPUrl**.

Directives utilisées pour l'authentification LDAP

Si la syntaxe peut sembler impressionnante, elle n'est pas due au hasard ou à l'imagination du développeur. Le format des URL LDAP est défini dans la RFC2255 et est utilisé dans quelques applications.

```
AuthName "Texte"
AuthType Basic
Require Valid-user
AuthLDAPUrl ldap://192.168.1.11/ou=users,dc=annu,dc=fr?cn=sub?objectclass=*
```

4. Authentification simple par fichier .htaccess

Les bonnes pratiques Apache indiquent d'utiliser une directive **Directory** à chaque fois qu'une configuration spécifique doit s'appliquer au contenu d'un répertoire et de placer dans le bloc défini par cette directive les éléments de configuration spécifiques à ce répertoire. Une méthode alternative consiste à placer un fichier **.htaccess** dans le répertoire en question, et d'y intégrer les directives devant s'appliquer au contenu du répertoire.

Exemples comparés d'exploitation


Extrait de fichier de configuration Apache avec la directive Directory :

```
...
<Directory /var/www/prot>
AllowOverride all
authType basic
AuthName "Veuillez vous identifier"
Require valid-user
AuthUserFile /etc/httpd/mdp
</Directory>
...
```

```
authType basic
AuthName "Veuillez vous identifier"
Require valid-user
AuthUserFile /etc/httpd/mdp
```

Il peut arriver que les deux configurations se trouvent en conflit, et qu'une directive soit configurée de façon spécifique pour un répertoire dans un contexte directory, et que la même directive soit configurée différemment dans un fichier .htaccess du même répertoire. En ce cas, la directive **AllowOverride** permet de préciser quelle sera la configuration retenue.

Valeurs courantes de la directive AllowOverride	
all	Valeur par défaut : Toute directive est autorisée dans les fichiers .htaccess.
none	Aucune directive n'est autorisée dans les fichiers .htaccess et les fichiers .htaccess sont ignorés.
AuthConfig	Autorise les directives relatives aux mécanismes d'authentification.

 Il est vivement recommandé de ne pas appliquer de directive AllowOverride All au répertoire racine de votre serveur web (Valeur par défaut !). Il faut plutôt définir cette valeur à None, et créer une directive Directory avec la directive AllowOverride configurée pour le seul répertoire concerné. Cette méthode augmente la sécurité, mais aussi la performance en évitant au serveur web de chercher un hypothétique fichier .htaccess dans chacun des répertoires visités.