

# Applications Réparties T.P. 2

## Installation et configuration Apache 2.0 sous Linux

---

Ce TP se déroulera sur vos PCs portables, sous Linux. Vous utiliserez votre compte **root** (commande **sudo**).

### 1 Installation et configuration de base

#### 1.1 Installation du serveur HTTP

Installer (si ce n'est pas déjà fait) le paquetage **httpd** ou **apache2** avec votre gestionnaire de paquetage **yum**, **apt-get**... :

```
Sudo yum install httpd
```

ou

```
Sudo apt-get install apache2
```

Comment vérifier qu'il est lancé dans les niveaux d'exécution 3, 4 etc. ?

Pour démarrer votre serveur, tapez la commande :

```
/etc/rc.d/init.d/httpd start
```

Ou :

```
/etc/init.d/apache2 start
```

Si vous exécutez Apache comme serveur sécurisé, une invite vous demandera votre mot de passe. Votre serveur démarrera dès que vous l'aurez tapé.

Pour arrêter votre serveur, tapez la commande :

```
/etc/rc.d/init.d/httpd stop
```

Ou :

```
/etc/init.d/apache2 stop
```

La commande **restart** est la façon la plus rapide d'arrêter et de redémarrer votre serveur. La commande **restart** arrête puis redémarre votre serveur. Une invite vous demande alors votre mot de passe, si vous exécutez Apache en tant que serveur sécurisé. La commande **restart** ressemble à ceci :

```
/etc/rc.d/init.d/httpd restart
```

Ou :

```
/etc/init.d/apache2 restart
```

Si vous venez de finir de modifier quelque chose dans votre fichier **httpd.conf**, il n'est pas nécessaire d'arrêter et de redémarrer votre serveur. Vous devriez par contre utiliser la commande **reload**. Lorsque vous utilisez **reload**, vous ne devez pas taper votre mot de passe. Votre mot de passe demeure en cache durant les rechargements, mais pas durant les arrêts et redémarrages. La commande **reload** ressemble à ceci :

```
/etc/rc.d/init.d/httpd reload
```

Ou :

# Applications Réparties T.P. 2

## Installation et configuration Apache 2.0 sous Linux

```
/etc/init.d/apache2 reload
```

Par défaut, le processus **httpd/apache2** démarre automatiquement au lancement de votre ordinateur. Si vous exécutez Apache en tant que serveur sécurisé, une invite vous demandera le mot de passe après le lancement de l'ordinateur, à moins que vous n'ayez généré une clé pour votre serveur sécurisé sans protection avec mot de passe.

### 1.2 Configuration du serveur HTTP

Dans apache 1, le fichier de configuration principal du serveur est `/etc/httpd/conf/httpd.conf`. Il est conservé vide dans Apache2 pour assurer la rétrocompatibilité, le fichier de configuration principal est désormais `/etc/apache2/apache2.conf`. Editer ce fichier (ou les fichiers inclus) et répondre aux questions suivantes, qui se réfèrent toutes à la configuration par défaut :

1. Quelle directive spécifie le port TCP sur lequel écouter ? Quel est le port par défaut ?
2. Il y a-t-il d'autres fichiers de configuration chargés depuis `httpd.conf/apache2.conf` ? Où sont-ils placés ? Donner leur liste.
3. Quelle est la directive chargeant un module ?
4. Sous quelle identité Unix (utilisateur et groupe) le serveur va-t-il s'exécuter ? Donner les UID et GID correspondants.
5. Quel est le répertoire racine pour les documents (pages) servis ?
6. Quelle page Apache renvoie-t-il lorsque l'URL demandée correspond à un répertoire ?
7. Comment sont traitées les URL de la forme `http://serveur/cgi-bin/toto` ?
8. Quels sont les fichiers de logs générés ? Où sont-ils placés ? Quel est leur format et comment est-il contrôlé ?

### 1.3 Création de documents à servir par HTTP

Créer une page HTML minimale et la placer à la racine de l'arborescence servie. Lancer le service `httpd/apache2`. Dans quel fichier de log peut-on constater le démarrage ?

### 1.4 Accéder à votre document

Accéder à votre page web. Quelle URL utiliser ? Peut-on y accéder depuis une autre machine de la salle ? Avec quelle URL ? Observe-t-on des traces dans les logs ?

## 2 Mais première pages dynamiques : Scripts CGI simples et Formulaires

### 2.1 Pages dynamiques CGI

1- Ecrire en Shell BASH un script CGI nommé `date` qui renvoie la date et l'heure dans une page HTML. Où placez-vous le script ? Faut-il configurer quelque chose ?

2- Modifier la configuration pour que toutes les pages suffixées par `.sh` soient considérées comme des CGI et exécutées.

(Note : cette façon de faire n'est pas recommandée sur un vrai serveur car l'exécution de CGI peut poser des problèmes de sécurité, et il vaut donc mieux les regrouper dans un répertoire bien surveillé).

3- Ecrire un CGI qui permette d'afficher la liste des processus appartenant à un utilisateur donné. `http://serveur/listeprocs.sh?user=toto` afficherait dans une page HTML la liste des processus de `toto` s'exécutant sur le serveur.

# Applications Réparties T.P. 2

## Installation et configuration Apache 2.0 sous Linux

### 2.2 Formulaire

Ecrire une page HTML avec un formulaire simple permettant la saisie du nom de l'utilisateur.

### 3 Protection des accès

1- Créez un répertoire secret dans l'arborescence web et placez-y une page HTML. Configurez Apache afin que vous ne puissiez accéder au répertoire secret que depuis le serveur lui-même.

Quel est le code renvoyé par Apache lorsqu'on tente d'accéder à ce répertoire depuis une autre machine ? Qu'observe-t-on dans les logs ?

2- Soit à protéger l'accès au sous site privé d'un établissement, supposons qu'il s'agit du sous répertoire /var/www/html/privé.

Il ne devra être accessible qu'à un ensemble limité de comptes Apache (et non Linux) à créer.

Une requête s'adressant à ce répertoire protégé provoquera l'affichage d'une boîte de dialogue par laquelle l'utilisateur devra s'authentifier (nom et mot de passe).

#### Principe :

La clause **AccessFileName .htaccess** fixe globalement le nom des fichiers de paramètres locaux.

Un fichier de ce nom, présent dans un répertoire, peut contrôler complètement les accès à ce répertoire, pourvu que la permission soit accordée par la directive

**AllowOverride AuthConfig** ou **AllowOverride All**

Alors, les directives contenues dans ce fichier seront systématiquement respectées avant toute autorisation. Voici les directives usuelles et leur signification :

**AuthType basic**, type d'authentification communément adopté, fait hélas circuler les mots de passe en clair ;

**AuthName texte**, affichera le texte comme invite dans la boîte de dialogue ;

**AuthUserFile chemin/fichier**, précise le fichier qui contient les comptes et mots de passe des utilisateurs ayant droit d'accès ;

**Require valid-user** | **liste-noms** tous, ou seulement les comptes énumérés dans la liste, auront accès au répertoire.

1. Créer le répertoire /var/www/html/privé, y placer quelques pages HTML. Tester leur accessibilité pour tous. Sinon penser à modifier les permissions Linux sur ces fichiers.

2. Créer dans ce répertoire à protéger le fichier .htaccess. En voici une écriture standard :

```
AuthUserFile /etc/apache2/users
AuthGroupFile /dev/null
AuthName "Accès privé"
AuthType Basic
# autres clauses
# AuthGroupFile /etc/httpd/conf/groups

<limit GET>
# ATTENTION : GET en majuscules !
require valid-user
```

## Applications Réparties T.P. 2

# Installation et configuration Apache 2.0 sous Linux

```
# require user toto dupond
# require group profs
</limit>
```

3. Dans ces conditions où se trouvera le fichier d'authentification ?
4. Créer un premier compte Apache avec la commande htpasswd.

```
cd /etc/apache2/
htpasswd -c users admin
--> mot de passe demandé (admin), puis confirmé.
```

5. Examiner le fichier **/etc/ apache2/users**  
L'utilitaire htpasswd a créé (option -c) le fichier users dans le répertoire courant, ici **/etc/ apache2**, et y a enregistré admin avec son mot de passe crypté.
6. Ajouter un second compte, toto, puis d'autres

```
htpasswd users toto
--> mot de passe demandé, puis confirmé
```

7. Tester l'accès au répertoire **http://serveur/prive**. Pourquoi la protection ne semble-t-elle pas fonctionner ? (Remarque : service httpd reload permet de prendre en compte les changements de configuration)

8. Rechercher dans le fichier de configuration la section **<Directory /var/www/html>** qui fixe des directives par défaut pour le site principal. Par sécurité mettre si nécessaire la clause **AllowOverride None**

9. Ajouter une directive concernant le répertoire privé

```
<Directory /var/www/html/prive>
AllowOverride ...
Options -Indexes
.....
</Directory>
```

10. Re-tester normalement avec succès ! N'oubliez pas de relancer le navigateur quand on change de compte.

## 4 Serveur "virtuel"

Lorsqu'une machine doit héberger plusieurs sites web différents et même si elle ne dispose que d'une adresse IP, on utilise la technique des "serveurs virtuels" (Virtual Hosts).

- 1- Donner plusieurs noms à votre machine (soit via /etc/hosts soit en modifiant le DNS de la salle s'il y en a un).
- 2- Configurez Apache, en vous inspirant de ces extraits (à adapter) :

```
# Protection maximale de la racine de l'hôte du serveur
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

# Paramétrage du site web usuel accessible par http://serveur/
#####
<Directory "/var/www/html">
```

## Applications Réparties T.P. 2

### Installation et configuration Apache 2.0 sous Linux

```
# Options possibles : "All", ou une combinaison de
Indexes,Includes,FollowSymLinks,ExecCGI,MultiViews

Options -Indexes FollowSymLinks

# Pour empêcher l'action "outrepasante" des fichiers .htaccess dans les répertoires
# les paramètres possibles sont All, ou une combinaison qqc de Options,FileInfo,AuthConfig,L

    AllowOverride None

# Pour contrôler les permissions d'accès au serveur
# pour des droits restrictifs interdire D'ABORD de partout, puis ENSUITE
# accorder à des machines particulières
    order deny,allow
    deny from all
    allow from localhost 10.0.0.0/255.255.255.0
</Directory>

# Paramétrage d'un site web virtuel accessible par http://toto.gtr.org/
#####

<Directory "/home/totoweb">
Options Indexes FollowSymLinks
order deny,allow
    deny from all
    allow from localhost 10.0.0.0/255.255.255.0
</Directory>

Site principale et sites virtuels :

#####
# Hotes virtuels nommés
#####

# le numéro ip de la machine
NameVirtualHost 10.0.0.5

# Le premier paragraphe décrit le site principal
#####

<VirtualHost 10.0.0.5>
    DocumentRoot /var/www/html
    ServerName poste01.perp77.fr
</VirtualHost>

# serveur virtuel pointant dans une autre partition
#####

<VirtualHost 10.0.0.5>
```

## Applications Réparties T.P. 2

# Installation et configuration Apache 2.0 sous Linux

```
DocumentRoot /home/totoweb  
ServerName toto.gtr.org  
</VirtualHost>
```

- 3- Tester votre configuration.
- 4- Comment séparer les logs d'accès de chaque serveur virtuel ?