TP serveur web

Olivier AUBERT

11 mars 2002

1 Installation d'un serveur web

1.1 Accès aux paquetages debian

On installera `apache` à partir du paquetage debian correspondant. Pour cela, on configure le gestionnaire de paquetages debian `apt` pour qu'il aille chercher les données sur une machine serveur (dont l'adresse est précisée par l'encadrant en début de TP) via un serveur web. Il faut donc éditer le fichier `/etc/apt/sources.list` et ajouter la ligne

```
deb http://134.214.90.7/debian stable main contrib
```

Pour mettre à jour la liste des paquetages sur la machine, lancez la commande

```
apt-get update
```

Il est possible de chercher la liste des paquetages existants par la commande `apt-cache search` suivie d'un mot-clé.

Pour partir d'une configuration propre, on va commencer par supprimer les restes des paquetages qui peuvent être encore présents, par la commande

```
dpkg --purge apache apache-common
```

Installez ensuite les paquetages `apache`, `apache-common` et `apache-doc` avec la commande `apt-get install`.

1.2 Exploration des fichiers

Vous pouvez obtenir la liste des fichiers installés par le paquetage `apache` par la commande `dpkg -L apache | less`. Les fichiers de configuration du serveur son situés dans le répertoire `/etc/apache/`. Ce sont les fichiers `httpd.conf`, `access.conf` et `srm.conf`. Dans les dernières versions d'apache, toutes les directives sont dans l'unique fichier `httpd.conf`. Pour réaliser ce TP, vous pourrez vous inspirer de l'exemple plus complet de fichier de configuration situé dans `/usr/share/doc/apache/examples/httpd.conf-dist`.

Utilisez la même commande pour déterminer où se trouvent les modules fournis par le paquetage `apache-common` (ce sont des fichiers avec l'extension `.so`).

Un historique des accès et des erreurs du serveur web sont conservés dans des fichiers *de log*, qui se trouvent dans le répertoire `/var/log/apache`.

Au cours de ce TP, vous serez amenés à modifier les fichiers de configuration. Il est nécessaire d'informer le démon `apache` de ces modifications afin qu'elles soient prises en compte. Ceci s'effectue par la commande

```
/etc/init.d/apache reload
```

1.3 Accès au serveur

Un fichier de configuration par défaut étant en place, vous pouvez d'ores et déjà vous y connecter via un navigateur web, en précisant l'adresse `http://localhost/`. La documentation est accessible via le serveur web lui-même, à l'adresse `http://localhost/doc/apache/manual/`.

Effectuez un accès à ce serveur avec un navigateur web (indice : `lynx`, navigateur en mode texte, prend moins de ressources sur la machine et est plus rapide à lancer). Consultez le fichier de log `access.log` pour vérifier les informations qui s'y trouvent. Utilisez un analyseur de réseaux (`ethereal`) pour voir quelles sont les informations échangées entre le serveur et le client.

Modifiez le fichier de configuration `httpd.conf` afin d'enrichir les informations de ce fichier de log, pour indiquer notamment le document ayant amené vers le document courant. Effectuez quelques accès pour vérifier que les nouvelles informations sont bien enregistrées.

Trouvez dans le fichier de configuration l'endroit où est spécifié le répertoire racine du serveur.

2 Configuration du serveur

2.1 UserDir

L'objectif est de permettre aux utilisateurs d'avoir un espace accessible dans leur propre répertoire. Trouvez (dans le fichier `httpd.conf-dist`) le nom du répertoire qu'un utilisateur doit créer dans son *HOME directory* pour stocker ces fichiers. Créez un utilisateur avec votre nom de login (commande `adduser nom-de-login`). Connectez-vous sous son identité (par la commande `su nom-de-login` ou `telnet localhost`) et, sous son identité, créez le répertoire adéquat pour pouvoir mettre à disposition des données. Vérifiez que les données sont bien accessibles.

Modifiez le fichier de configuration afin d'interdire l'indexation automatique des documents des répertoires utilisateur.

Ajoutez un alias permettant d'accéder votre compte utilisateur par l'adresse `http://localhost/login` plutôt que `http://localhost/~login/`.

2.2 Page d'erreur

Configurez le serveur de manière à retourner une page d'erreur (404) différente de la page d'erreur standard, que vous créerez dans la racine du site. Une utilisation courante est d'ajouter un lien vers un moteur de recherche local au site, afin que la personne puisse chercher le document adéquat.

2.3 Affichage de l'information sur le serveur

Activez l'affichage d'informations d'état sur le serveur. Pour cela, il faut charger le module `mod_info`, et fournir un point d'entrée au serveur afin qu'il active ce module (via la directive `Handler` associée à une `Location`). L'accès aux informations s'effectuera alors par l'adresse `http://localhost/server-info`.

2.4 Redirection

Effectuez la redirection de l'adresse `http://localhost/134.214.89.xx` vers la racine du serveur web `http://134.214.89.xx/` géré par votre voisin, par l'utilisation de la directive `Redirect`.

2.5 Authentification

On va ajouter une procédure d'authentification par mot de passe pour accéder au compte de l'utilisateur créé précédemment. Pour cela, on crée un fichier nommé `.htaccess` dans le répertoire à protéger. La syntaxe de ce fichier est la suivante :

```
AuthUserFile           /home/nom-de-login/private/htpasswd
AuthName "Domaine protégé"
AuthType Basic
<Limit GET POST>
require valid-user
</Limit>
```

Pour créer le fichier `htpasswd`, il faut utiliser la commande `htpasswd` (avec l'option `-h` pour obtenir l'aide).

L'exemple présenté ici n'autorise l'accès qu'aux utilisateurs déclarés dans le fichier `htpasswd`. Il est possible de restreindre à un utilisateur particulier via la commande `require user nom-de-login`, ou à un groupe d'utilisateurs. Dans ce dernier cas, il faut créer et déclarer un fichier de groupes similaires à `htpasswd` par la directive `AuthGroupFile` et indiquer la restriction `require group nom-de-groupe`.

Enfin, il est possible de restreindre l'accès à certaines machines seulement. Pour cela, la syntaxe devient

```
AuthName "Domaine protégé"
AuthType Basic
<Limit GET POST>
order deny,allow
deny from all
allow from localhost
</Limit>
```

pour n'autoriser l'accès que depuis la machine locale par exemple.

Faites des tests d'accès en utilisant diverses méthodes, et en les combinant éventuellement.

2.6 VirtualHost

Un même serveur web peut être utilisé pour servir les documents de plusieurs sites différents. Pour cela, la distinction s'effectue par le nom de la machine utilisée.

Créez dans le fichier `/etc/hosts` de votre machine un nouveau nom de machine (votre nom de login) pointant vers la même adresse IP. Vérifiez par la commande `ping` que votre modification est prise en compte.

Nous allons maintenant faire en sorte que les accès à l'adresse `http://nom-de-login/` renvoient une information distincte de celle renvoyée par `http://localhost/`, celle contenue dans le répertoire `/home/nom-de-login/public_html`. Pour cela, il va falloir utiliser la directive `VirtualHost` dont un exemple est donné à la fin du fichier de configuration. Les messages d'accès et d'erreur seront stockés dans le répertoire `/home/nom-de-login/apache` (qu'il faudra créer). L'adresse mail de l'administrateur sera fixée à votre propre adresse.

2.7 Bonus

Si vous êtes arrivés jusqu'ici, vous pouvez vous amuser avec les nombreux autres modules disponibles dans apache. Vous pouvez vous intéresser par exemple aux modules suivants :

- `mod_unique_id` génère un identificateur unique pour chaque requête.
- `mod_rewrite` permet d'effectuer une réécriture des URL afin d'effectuer des redirections intelligentes.
- `mod_speling` corrige automatiquement les erreurs mineures dans les URL
- `mod_expires` permet de spécifier les dates d'expiration des documents
- `mod_usertrack` active la génération de cookies pour suivre le parcours d'un utilisateur.

3 Fin du TP

Merci de laisser la machine aussi propre que possible : supprimez le compte utilisateur que vous avez créé (commande `userdel -r nom-de-login`) et enlevez les modifications que vous avez apporté au fichier `/etc/hosts`.