Réseaux et Programmation
Le serveur Web Apache

Qu'il s'agisse d'accéder à des pages Web statiques, dynamiques depuis un client de type browser ou d'accéder à un service web depuis une application cliente, le serveur Web est incontournable.
Ce TD a donc pour objectif de mettre en place un serveur Web apache (https://httpd.apache.org), d'en mesurer les possibilités au travers ses multiples configurations.
Ce TD sera aussi l'occasion de retrouver dans nombre des cas le lien entre fonctionnalité du serveur Web et protocole HTTP.

1 Introduction

Pourquoi Apache ? … Un an après la création du groupe apache en 1995, l'Apache HTTP Server devient le serveur le plus utilisé (57 % des serveurs du Web). Depuis apache équipe toujours autour de la moitié des serveurs Web de la planète.

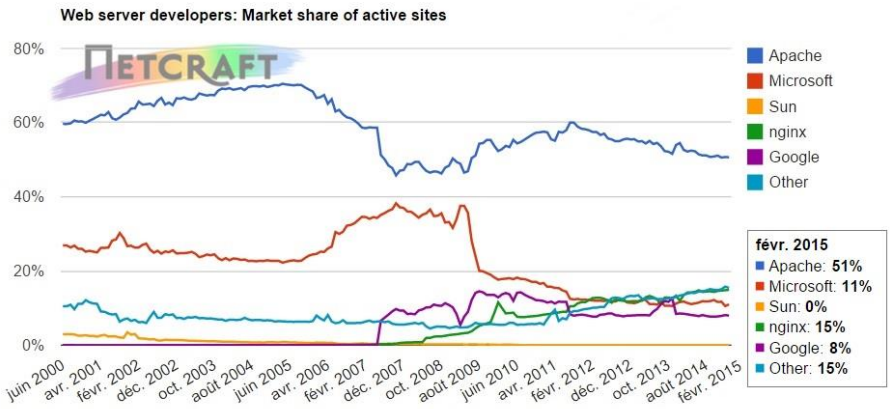

Figure 1 : Taux de marché des principaux servers web dans le monde (source : NetCraft)

Le serveur Web apache est aussi une solution pour cible légère comme le Raspberry Pi fonctionnant sous Linux/Raspbian. Ces opportunités technologiques permettent dès lors d'introduire des objets physiques connectés à Internet, dans le monde du World Wide Web. Les objets deviennent ainsi des fournisseurs de pages et de services Web.

2 Mise en œuvre du serveur Web Apache 2.4

Le serveur web Apache2 est disponible dans Ubuntu Linux. Pour installer Apache2 saisissez la commande suivante dans un terminal :

```
sudo apt-get install apache2
```

Vérifier qu'une version antérieure n'est pas installée, sinon la désinstallée car l'upgrade n'est pas systématique.

Le service sera lancé par la commande :

```
sudo service apache2 start
```

2.1 Configuration

Apache2 doit être configuré à l'aide de directives décrites dans des fichiers texte de configuration. Ces directives sont réparties entre les fichiers et dossiers suivants :

- Le fichier de configuration principal est **apache2.conf**. Il contient les paramètres globaux d'Apache2.
- **httpd.conf** : historiquement le principal fichier de configuration Apache2, nommé d'après le démon httpd. Maintenant, le fichier n'existe plus. Dans les anciennes versions d'Ubuntu, le fichier peut être présent mais vide, toutes les options de configurations ayant été déplacées dans le répertoire référencé ci-dessous.
- **conf-available** : ce répertoire contient les fichiers de configuration disponibles. Tous les fichiers qui étaient auparavant dans /etc/apache2/conf.d peuvent être déplacés dans /etc/apache2/conf-available.
- **conf-enabled** : conserve les liens symboliques dans /etc/apache2/conf-available. Quand un fichier de configuration est pointé, il sera activé au prochain démarrage d'apache2.
- **envvars** : fichier où les variables d'environnement d'Apache2 sont définies.
- mods-available : ce répertoire contient les fichiers de configuration qui permettent de charger les modules et de les paramétrer. Certains modules peuvent ne pas avoir de fichier de configuration.
- **mods-enabled** : contient les liens symboliques (symlinks) vers les fichiers de /etc/apache2/mods-available. Lorsqu'un lien symbolique vers un module de configuration est créé, il sera activé au prochain redémarrage d'Apache2.
- **ports.conf** : héberge les directives déterminant les ports TCP sur lesquels Apache2 est en écoute.
- **sites-available** : ce dossier contient des fichiers de configuration pour les serveurs virtuels d'Apache2. Les serveurs virtuels permettent de configurer Apache2 pour plusieurs sites ayant différentes configurations.
- **sites-enabled** : comme mods-enabled, sites-enabled contient des liens symboliques vers le dossier /etc/apache2/sites-available. De la même manière que lorsqu'un lien symbolique vers un fichier de configuration de sites-available est créé, le site ainsi configuré sera activé au prochain redémarrage d'Apache2.
- **magic** : instructions pour déterminer le type MIME à partir des premiers bytes d'un fichier.

Les changements effectués dans les fichiers de configuration principaux ne sont pris en compte qu'au (re)démarrage d'Apache2.

Le serveur lit également un fichier contenant les documents de type MIME, le nom du fichier est défini par la commande TypesConfig, généralement par l'intermédiaire de **/etc/apache2/mods-available/mime.**conf, qui peut également inclure des ajouts ou des remplacements, et est /etc/mime.types par défaut.

2.1.1 Gestion des modules

Si dans ce TD, tous les modules nécessaires sont normalement déjà disponibles, voici les étapes de la gestion d'un module **mod_xxx** :

- Les modules étant avant tout des librairies, un fichier **mod_xxx.so** doit être présent dans le répertoire /usr/lib/apache2/modules. Par exemple pour le **mod_alias** de gestion des directives **Alias** et **ScriptAlias** vous devez y trouver la librairie mod_alias.so.
- Le répertoire **/etc/apache2/mods-available** contient en général deux fichiers pour chaque module installé sur le système, un dont l'extension est **.load** qui contient la directive le chargement du réel module (généralement contenu dans le répertoire **/usr/lib/apache2/modules/)** et un autre dont l'extension est .conf qui contient la configuration par défaut du module en question. Au lancement d'apache, les modules figurant dans le **/etc/apache2/mods-available** sont donc chargés.

2.1.1.1 L'installation, l'activation et la désactivation de modules

Prenons l'exemple de l'installation du module php pour Apache.

On l'installe :

```
apt-get install libapache2-mod-php5
```

Les distributions à base de Debian fournissent par défaut des outils très simples pour l'activation et la désactivation des modules. Pour activer le nouveau module installé il suffit de lancer la commande :

```
sudo a2enmod php5
```

Dans le cas où l'on souhaiterait désactiver le module la commande inverse existe :

```
sudo a2dismod php5
```

`a2enmod` (pour Apache enable module) se contente de faire un lien symbolique de la définition du module choisi contenu dans `/etc/apache2/mods-available` vers `/etc/apache2/mods-enabled`

`a2dismod` (pour pache disable module) se contente quand à lui de supprimer tout bonnement ce lien symbolique.

Il suffit ensuite de recharger la configuration d'Apache pour que le module (avec sa configuration par défaut) soit pris en compte :

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

2.1.2 Réglages de base

Cette partie a pour objectif de manipuler les paramètres de configuration de base. Vous vous référerez à la documentation apache2.4 pour les détails.

Lire le fichier apache2.conf puis les fichiers inclus correspondant pour 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) servi(e)s ?

Réseaux et Programmation
Le serveur Web Apache

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 so nt les fichiers de logs générés ? Où sont-ils placés ? Quel est leur format et comment est-il contrôlé ?

2.1.3 Création de documents à servir par HTTP

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

2.1.4 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 ?

2.1.5 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.

3 Protection des accès

3.1 Répertoire protégé

Créer un répertoire secret dans l'arborescence web et placez-y une page HTML. Configurer 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 ?

3.2 Site Protégé

Protégeons l'accès au sous site privé d'un établissement, supposons qu'il s'agit du sous répertoire /var/www/html/prive.
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 boite de dialogue par laquelle l'utilisateur devra s'authentifier (nom et mot de passe).

3.2.1 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 boite 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.

3.2.2 Mise en œuvre

1. Créer le répertoire /var/www/html/prive, 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
# 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**
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.

3.3 Paramètres httpd

Cette section explique certains paramétrages basiques du démon httpd.

LockFile - La directive LockFile définit le chemin vers le fichier lockfile utilisé lorsque le serveur est compilé soit avec USE_FCNTL_SERIALIZED_ACCEPT, soit USE_FLOCK_SERIALIZED_ACCEPT. Ce fichier doit être enregistré sur le disque local. On doit laisser la valeur par défaut à moins que le répertoire des journaux se situe sur un partage NFS. Dans ce cas, la valeur par défaut doit être modifiée pour un emplacement, sur le disque local, lisible uniquement par le super-utilisateur (root).

PidFile - La directive PidFile définit le fichier dans lequel le serveur enregistre son ID de processus (pid : process ID). Ce fichier ne doit être accessible en lecture que par le super-utilisateur (root). Dans la plupart des cas, la valeur par défaut peut être conservée.

User - La commande User définit l'identité de l'utilisateur utilisée par le serveur pour répondre aux demandes. Ce paramètre détermine l'accès au serveur. Tous les fichiers inaccessibles à cet utilisateur seront également inaccessibles aux visiteurs de votre site web. La valeur par défaut pour l'utilisateur est "www-data".

À moins que vous ne sachiez exactement ce que vous faites, ne définissez jamais la directive User à root. Utiliser root (super-utilisateur) comme valeur pour User créera des failles de sécurité béantes dans votre serveur Web.

Group - La commande Group est similaire à la commande User. Group définit le groupe sous lequel le serveur répond aux requêtes. Le groupe par défaut est également "www-data".

3.3.1 Modules Apache2

Apache2 est un serveur modulaire. Cela signifie que seules les fonctionnalités les plus basiques sont incluses dans le cœur même du serveur. Les fonctionnalités étendues sont disponibles grâce aux modules qui peuvent être chargés dans Apache2. Par défaut, un ensemble de modules de base est inclus dans le serveur au moment de la compilation. Si le serveur est compilé pour utiliser les modules chargés dynamiquement, alors ces derniers peuvent être compilés séparément et ajouté à tout moment en utilisant la directive LoadModule. Sinon, Apache2 doit être recompilé pour ajouter ou supprimer des modules.

Ubuntu compile Apache2 pour permettre le chargement dynamique de modules. Les directives de configuration peuvent être incluses dans un module particulier en les délimitant par un bloc <IfModule>.

Vous pouvez installer des modules additionnels d'Apache2 et les utiliser avec votre serveur Web. Par exemple, exécutez la commande suivante dans un terminal pour installer le module Authentification MySQL :

```
sudo apt-get install libapache2-mod-auth-mysql
```

Consultez le répertoire /etc/apache2/mods-available pour voir les modules supplémentaires.

L'utilitaire a2enmod permet d'activer un module :

```
sudo a2enmod auth_mysql
sudo service apache2 restart
```

Inversement, a2dismod désactivera un module :

```
sudo a2dismod auth_mysql
sudo service apache2 restart
```

3.4 Configuration HTTPS

Le module mod_ssl ajoute d'importantes caractéristiques au serveur Apache2 - la capacité de chiffrer les communications. De plus, lorsque votre navigateur Web communique en utilisant SSL, le préfixe https:// est utilisé au début de l'adresse Web (URL) dans la barre d'adresse du navigateur.

Le module mod_ssl est disponible dans le paquet apache2-common. Saisissez la ligne de commande suivante pour activer le module mod_ssl :

```
sudo a2enmod ssl
```

There is a default HTTPS configuration file in /etc/apache2/sites-available/default-ssl.conf. In order for Apache2 to provide HTTPS, a certificate and key file are also needed. The default HTTPS configuration will use a certificate and key generated by the ssl-cert package. They are good for testing, but the auto-generated certificate and key should be replaced by a certificate specific to the site or server. For information on generating a key and obtaining a certificate see Certificats
Pour configurer Apache2 pour HTTPS, saisissez les informations suivantes :

```
sudo a2ensite default-ssl
```

Les répertoires par défaut sont /etc/ssl/certs et /etc/ssl/private. Si vous installez le certificat et la clé dans un autre répertoire, assurez-vous de modifier SSLCertificateFile et SSLCertificateKeyFile en conséquence.
Avec Apache2 maintenant configuré pour HTTPS, redémarrez le service pour activer les nouveaux paramètres :

```
sudo service apache2 restart
```

Selon comment vous avez obtenu votre certificat, vous devrez entrer un mot de passe lors du redémarrage d'Apache2.
Vous pouvez accéder aux pages du serveur sécurisé en tapant https://votre_nomdhote/url/ dans la barre d'adresse de votre navigateur.

3.5 Références

- Apache2 Documentation contains in depth information on Apache2 configuration directives. Also, see the apache2-doc package for the official Apache2 docs.
- Consultez le site Mod SSL Documentation (en anglais) pour plus d'informations concernant SSL.
- O'Reilly's Apache Cookbook (en anglais) est une bonne ressource pour réaliser des configurations spécifiques d'Apache2.
- Pour les questions concernant Apache2 mais spécifiques à Ubuntu, consultez le #ubuntu-server canal anglophone IRC sur freenode.net, ou le canal généraliste francophone #ubuntu-fr.
- Apache étant habituellement couplé avec PHP et MySQL, la page du Wiki Ubuntu consacrée à Apache MySQL PHP est également une très bonne source d'information.