Technologie Web

Serveur HTTP Apache

Alexandre Pauchet

INSA Rouen - Département ASI

BO.B.RC.18, pauchet@insa-rouen.fr

Plan

Description (1/3)
Présentation : A PAtCHy sErver

- Basé sur du code existant et une série de patchs
- Serveur open source : `http://httpd.apache.org`
- Conforme au protocole `HTTP/1.1` (rfc2616)
- Serveur le plus utilisé au monde
- 2 versions actuelles : 1.3.X et 2.X (réécrit et multi-threads)
- Multi-plateformes (Unix, Windows, Novell Netware)
- Très configurable et modulaire

Description (2/3)
Installation par paquets

- Le serveur Apache Http peut s'installer directement par paquets (`apt-get` ou via le gestionnaire de paquets)
- Paquets à installer : `libapr1`, `apache2`, `apache2.2-common`, `apache2-utils` et `apache2-mpm-worker`
- Modules complémentaires :
  - PHP5 : `php5-common`, `php5`, `php5-gd` et `libapache2-mod-php5` + modules optionnels php5
  - SQLite : `php5-sqlite` et `sqlite3`
  - MySQL : `php5-mysql`, `mysql-server` et `libapache2-mod-auth-mysql`
  - POSTGRESQL : `postgresql` `libapache2-mod-auth-pgsql`

Description (3/3)
Installation par compilation des sources

1. Récupération des dernières versions Apache et PHP sur `http://www.apache.org` et `http://www.php.net`
2. Configuration et compilation d'Apache Http

```
$ ./configure --prefix=Apache-Path --with-included-apr
$ make
$ make install
```

3. Configuration, compilation et installation du module PHP

```
$ ./configure --prefix=PHP-Path --with-apxs2=Apache-Path/bin/apxs
$ make
$ make install
```

4. Prise en compte des fichiers .php dans Apache
   Ajouter "`AddType application/x-httpd-php .php`" dans le `httpd.conf`

Configuration générale (1/5)
Systèmes de fichiers

- Version classique (VC)
  - Principal fichier de configuration : `httpd.conf`
  - Ajout de modules par inclusion de sous-fichiers de configuration dans `httpd.conf` :
    `Include conf/extra/fichier-à-ajouter.conf`
- Version Debian/Ubuntu (VDU)
  - Principal fichier de configuration : `apache2.conf`
  - Ajout de modules par création de liens symboliques du répertoire `mods-available` vers `mods-enabled`.
    Pour chaque module, 2 fichiers à lier : `.conf` et `.load`.
  - Ajout de sous-fichiers de configuration dans `httpd.conf` :
    `Include fichier-à-ajouter.conf`

Configuration générale (2/5)
Identification du serveur et emplacement des fichiers

- Configuration générale (VC : `httpd.conf`, VDU : `sites-enabled/000-default`)
  - `ServerName` : nom du serveur (VDU : `apache2.conf`)
    Ex : `ServerName 127.0.0.1`, `ServerName localhost:8080`, `ServerName www.example.com:80`
  - `ServerRoot` : spécifie le répertoire où est installé le serveur (VDU : `apache2.conf`)
    Ex : `ServerRoot "/tmp/apache2"`
  - `ServerAdmin` : adresse mail de l'administrateur
  - `DocumentRoot` : spécifie le répertoire racine d'Apache
  - `ErrorLog` : spécifie le fichier de log des erreurs
  - `CustomLog` : spécifie le fichier de log contenant les requêtes au serveur
- Types MIME reconnus : directive `TypesConfig` dans `mime.types` (VDU : fichier présent dans `mods-enabled`)

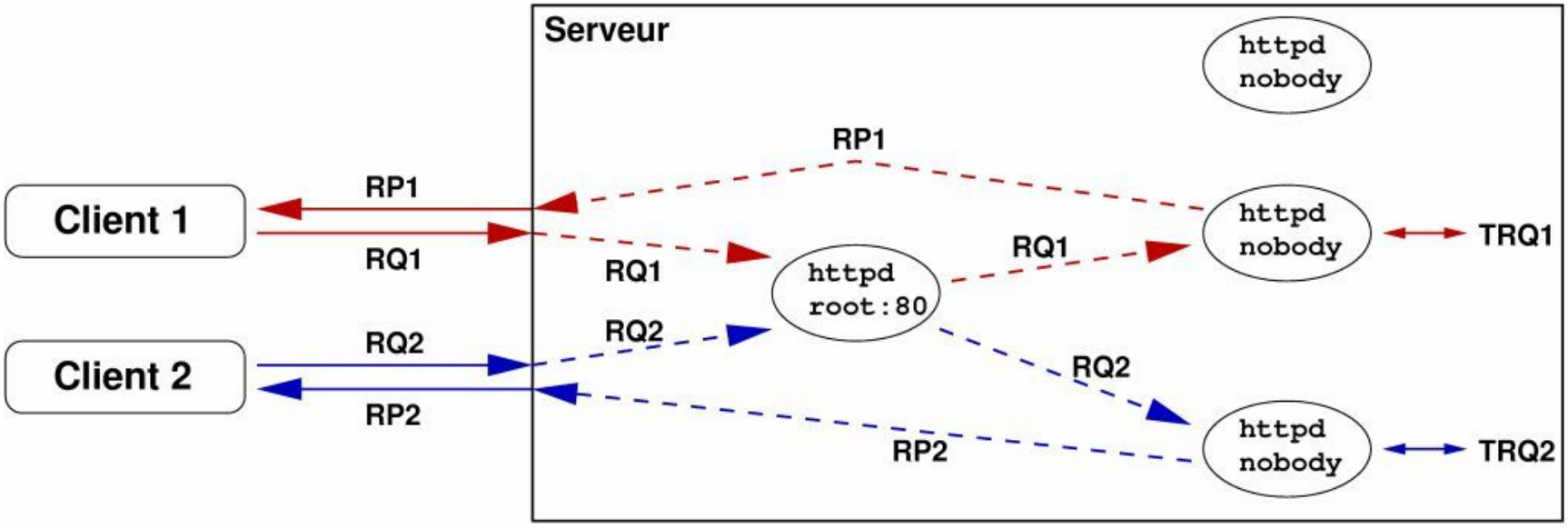
Configuration générale (3/5)
Configuration réseau

- `Listen` détermine le port de connexion au serveur (classiquement 80 pour http, parfois 8080)
  VC : `httpd.conf` / VDU : `ports.conf`
- `KeepAlive` spécifie si les connexions persistantes sont autorisées
  VC : `httpd-default.conf` / VDU : `apache2.conf`
- `KeepAliveTimeout` temps à partir duquel la connexion est coupée
  VC : `httpd-default.conf` / VDU : `apache2.conf`
- `Timeout` temps de réception d'une requête ou d'une réponse VC : `httpd-default.conf` / VDU : `apache2.conf`

Remarque

- C'est le serveur qui définit le mode de connexion par défaut
- La rfc2616 préconise une connexion persistante

Configuration générale (4/5)
Configuration des processus



- `StartServer`, `MaxSpareServers`, `MinSpareServers` définissent le nombre de processus fils
  VC : `httpd-mpm.conf` / VDU : `apache2.conf`
- `MaxRequestsPerChild` nombre de requêtes traitées par fils
  VC : `httpd-mpm.conf` / VDU : `apache2.conf`

Configuration générale (5/5)

Démon HTTP

- `User/Group` : utilisateur et groupe sous lesquels est exécuté le démon HTTP.

  Dans `httpd.conf` pour VC

```
<IfModule !mpm_netware_module>
    User daemon
    Group daemon
</IfModule>
```

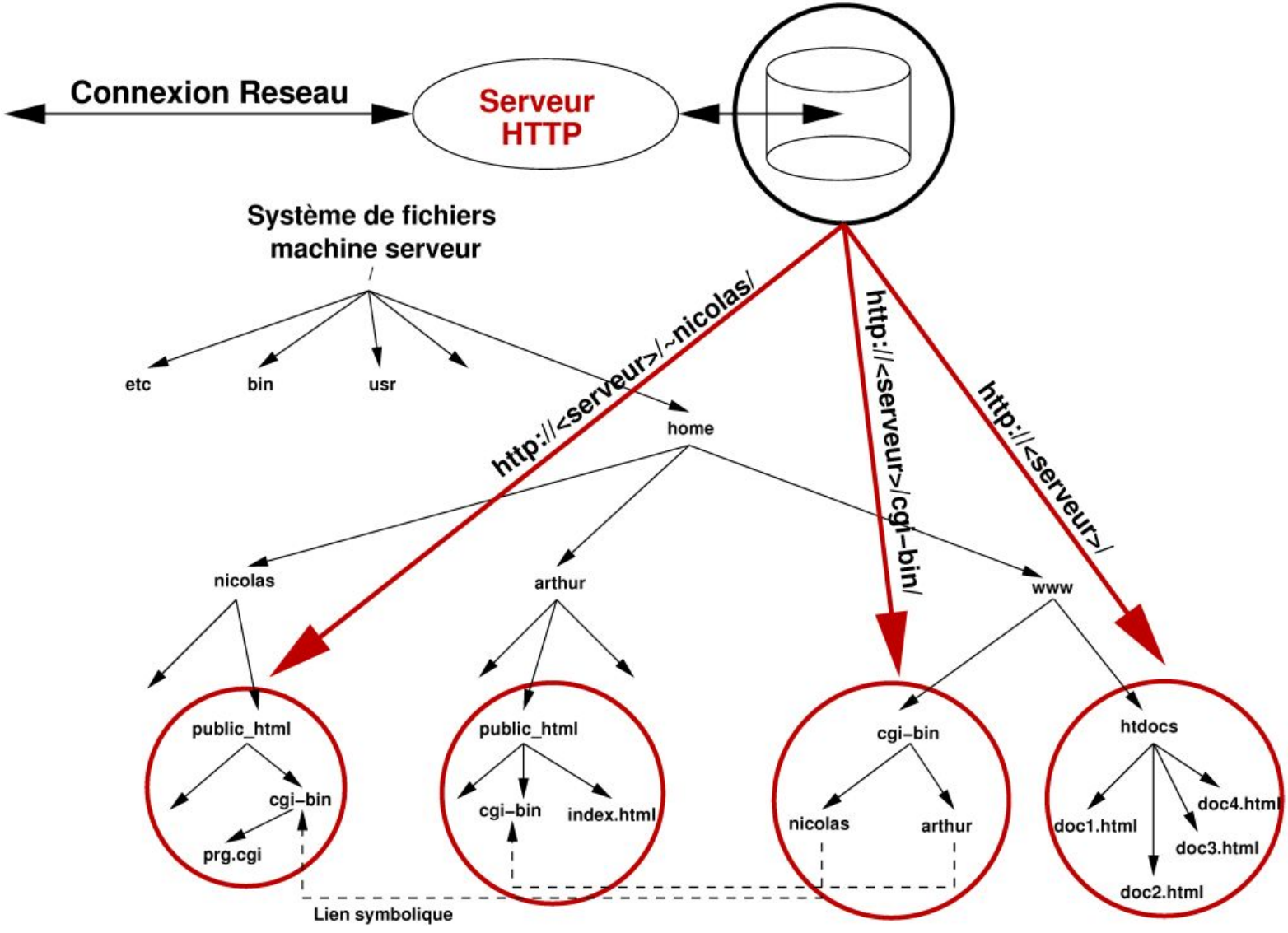
  Dans `apache2.conf` (et `envvars` par délégation) pour VDU

Configuration des accès et opérations (1/7)
Fichier par défaut

- `DirectoryIndex` : le(s) fichier(s) par défaut (`index.html index.htm index.xhtml index.cgi index.php` ...)
  VC : `httpd.conf` / VDU : `dir.conf`

Configuration des accès et opérations (2/7)

Système de fichiers



Configuration des accès et opérations (3/7)
Système de fichiers

- `UserDir` spécifie le nom du répertoire web des utilisateurs (usuellement `public_html`)
  VC : `httpd-userdir.conf` / VDU : `userdir.conf`
- `Alias` : association d'une URL à un répertoire local
  VC : `httpd.conf` / VDU : `alias.conf`
- `ScriptAlias` : idem, mais uniquement pour les CGI
  VC : `httpd.conf` / VDU : `alias.conf`

- `AccessFileName` : fichier de contrôle d'accès (`.htaccess`)
  VC : `httpd-default.conf` / VDU : `apache2.conf`

Configuration des accès et opérations (4/7)

Le contrôle des traitements

- `Redirect` : redirection d'une URL sur une autre
  Exemple : `Redirect /google http://www.google.fr`
  VC : `httpd.conf` / VDU : `apache2.conf`
- `AddHandler` : association d'une extension à un gestionnaire
  Exemple : `AddHandler cgi-script .cgi`
  VC : `httpd.conf` / VDU : `mime.conf`
- `ErrorDocument` : spécifie le document à renvoyer si erreur
  Exemple : `ErrorDocument 404 /doc_missing.html`
  VC : `httpd.conf` / VDU : `apache2.conf`

Configuration des accès et opérations (5/7)

Le contrôle des opérations : configuration des répertoires

- Encapsulation de directives liées à un **répertoire** :

```
<Directory répertoire>
  Options ...
  AllowOverride ...
  <Limit GET POST ...>
    ...
  </Limit>
  ...
</Directory>
```

- `Options` : options du répertoire
  - `None/All`
  - `ExecCGI FollowSymLinks/SymLinksIfOwnerMatch Indexes ...`
- `AllowOverride` : ce que le `.htaccess` peut outrepasser
  - `All/None AuthConfig FileInfo Indexes Limit ...`

$\Rightarrow$ s'appliquent aux répertoires du serveur et aux répertoires utilisateurs.

Configuration des accès et opérations (6/7)

Le contrôle sur les méthodes d'accès à un répertoire

- Limitation de l'accès d'un répertoire

```
<Limit GET POST ...>
  ...
</Limit>
```

`<Limit>` est (souvent) associé avec les directives :

  - de type `require` :
    - `require [group|user]` $nom_1, \ldots, nom_n$
    - `require valid-user`
  - `AuthName` : Information transmise à l'utilisateur
  - `AuthType` : `Basic|Digest`
  - `AuthUserFile` : chemin absolu vers le `.htpasswd`
  - `AuthGroupFile` : chemin absolu vers le `.htgroup`

Configuration des accès et opérations (7/7)

Le contrôle de l'accès à certaines machines

- Dans les `<Directory>` comme dans les `<Limit>`, il est possible de contrôler l'accès de certaines machines :
  - `Deny from` *nom*|*nom partiel*|*IP*|*IP partiel*
  - idem `Allow from ...`
  - `Order Deny,Allow` ou `Order Allow,Deny`

Exemple

```
<Directory /docroot>
    Order Deny,Allow
    Allow from 192.1
    Deny from apache.org .net
</Directory>
```

Répertoires et authentification (1/4)

.htaccess & AllowOverride

`AllowOverride` autorise la redéfinition de directives locales à un répertoire dans un fichier `.htaccess` situé dans ce dernier :

- `None` : les fichiers `.htaccess` sont ignorés
- `All` : tout type de redéfinition est autorisé dans le `.htaccess`
- `AuthConfig` : autorise l'authentification d'utilisateurs
- `FileInfo` : autorise les directives liées aux types de documents
- `Indexes` : autorise l'indexation des répertoires
- `Limit` : autorise les directives de gestion d'accès
- `Options` : autorise les directives se rapportant aux fonctionnalités des répertoires

Répertoires et authentification (2/4)
Exemples de <Directory>

Déclaration d'authentification nécessaire

```
<Directory /home/ofrais/public_html/technoweb>
   AuthUserFile /home/ofrais/login/.htpasswd
   AuthGroupFile /dev/.groups
   AuthName "Acces Restreint"
   AuthType Basic
   <Limit GET POST>
      require valid-user
   </Limit>
</Directory>
```

Délégation au .htaccess avec AllowOverride

Exemple, dans userdir.conf de technoweb :

```
<Directory /home/*/public_html>
   AllowOverride FileInfo AuthConfig Limit
   Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
</Directory>
```

Répertoires et authentification (3/4)

Le fichier .htaccess

Le fichier .htaccess

```
AuthName TEXTE
AuthType Basic
AuthUserFile <chemin absolu>/.htpasswd
AuthGroupFile  <chemin absolu>/.htgroup
<Limit GET POST>
   require group groupe1 ... groupeN
   require user utilisateur1 ... utilisateurN
</Limit>
```

Remarque : la directive `require valid-user` accepte tout utilisateur déclaré

Répertoires et authentification (4/4)

Utilisateur et mots de passe

Fichier .htgroup

```
groupe1 utilisateur1 ... utilisateurN
...
groupeN utilisateur1 ... utilisateurN
```

Fichier .htpasswd

```
utilisateur1:mot de passe crypté
...
utilisateurN:mot de passe crypté
```

fichier créé à l'aide de la commande `htpasswd` fourni avec Apache

Remarque

Attention : tous ces fichiers doivent être inaccessibles ! ! !

Serveurs virtuels (1/3)
Principe des serveurs virtuels

Plusieurs serveurs web sur une même machine
Deux possibilités de serveurs virtuels

- basés sur les IP :
  une adresse IP pour chaque serveur virtuel
- basés sur les noms :
  même adresse IP, mais nom pour chaque serveur virtuel

- Pour VC : `httpd-vhosts.conf`
- Pour VDU : `ports.conf` et `sites-enabled/000-default`

Remarque

`Host:` dans l'en-tête `Http` est indispensable en cas de serveur virtuel

Serveurs virtuels (2/3)

Serveurs virtuels basés sur les IPs

Directive

`<VirtualHost nom/IP> ...</VirtualHost>`

Exemples

```
<VirtualHost 193.254.105.78>
ServerAdmin canari@titi.fr
DocumentRoot /titi/documents
ServerName www.titi.fr
ErrorLog /titi/logs/error_log
TransferLog /titi/logs/access_log
</VirtualHost>

<VirtualHost 193.254.105.79>
ServerAdmin matou@gros-minet.fr
DocumentRoot /gros-minet/documents
ServerName www.gros-minet.fr
ErrorLog /gros-minet/logs/error_log
TransferLog /gros-minet/logs/access_log
</VirtualHost>
```

Configuration des serveurs par ajout de directives internes.
Chaque serveur virtuel doit connaître à la fois son IP et son nom.

Serveurs virtuels (3/3)

Serveurs virtuels basés sur les noms

Directives

`NameVirtualHost` *IP[:Port]*

`<VirtualHost` *nom|IP*`> ...</VirtualHost>`

Exemple

```
NameVirtualHost 193.254.105.78

<VirtualHost 193.254.105.78>
ServerAdmin canari@titi.fr
DocumentRoot /titi/documents
ServerName www.titi.fr
ErrorLog /titi/logs/error_log
TransferLog /titi/logs/access_log
</VirtualHost>

<VirtualHost 193.254.105.78>
ServerAdmin matou@gros-minet.fr
DocumentRoot /gros-minet/documents
ServerName www.gros-minet.fr
ErrorLog /gros-minet/logs/error_log
TransferLog /gros-minet/logs/access_log
</VirtualHost>
```

Optimisation et sécurité (1/3)

Droits sur la racine du serveur

Remarque

- Les répertoires `bin`, `conf`, `logs` sont des répertoires très sensibles
- Le propriétaire du répertoire du serveur et l'utilisateur `User` doivent être différents
- Seul le propriétaire du serveur peut modifier les fichiers (chmod 755)
- Il faut faire particulièrement attention à la configuration des SSI (Server Side Includes) et des CGI (Common Gateway Interface)

Optimisation et sécurité (2/3)
Sécurisation des SSI et des CGI

- Sécurisation des SSI :
  - Extension des fichiers SSI différentes des fichiers HTML
  - Désactivation de la commande `exec` (`IncludesNOEXEC`)

- Sécurisation des CGI : Non ScriptAlias *versus* ScriptAlias CGI
  - Non ScriptAlias (faille de sécurité importante)
    - appel de type `http://<serveur>/~user/cgi-bin/script.cgi`
    - confiance envers les utilisateurs
    - aucun contrôle sur les scripts des utilisateurs
  - ScriptAlias (solution la plus utilisée)
    - appel de type `http://<serveur>/cgi-bin/<user>/script.cgi`
    - confiance envers les utilisateurs
    - centralisation des scripts CGI
    - contrôle relativement aisé
  - Problème : tous les CGI s'exécutent sous le même utilisateur
    - solution : configurer Apache en `suexec` (CGI et SSI sous des UID différents de l'utilisateur)

Optimisation et sécurité (3/3)

Protection des fichiers de configuration et des fichiers du serveur

Dispositifs de sécurité minimums

- Interdire l'utilisation des `.htaccess` à partir de la racine
- Interdire l'accès à la racine du serveur

```
<Directory />
    AllowOverride None
    Order Deny,Allow
    Deny from all
</Directory>
```

Références

- http://www.apache.org : Apache software foundation
- http://doc.ubuntu-fr.org/apache2 : Documentation Ubuntu sur Apache Http
- http://apachetoday.com : site d'informations sur Apache Http
- http://www.jalix.org/ressources/reseaux/apache/_fiches_httpd/HTTPD.PS : cours de Maurice Szmurlo
- http://www.php.net/ : site officiel PHP