



# *Appropriation d'un serveur Apache*

*Jacquelin Charbonnel - CNRS LAREMA*

*Aide à la détection des faiblesses d'un site web*  
*Journées UREC - Montpellier - Septembre 2008*

# Introduction

Constat :

- ☉ un Apache fraîchement installé dispose d'un niveau de sécurité satisfaisant
- ☉ au fil du temps :
  - ☉ le nombre de documents croît, les webmasters sont plus nombreux => la configuration s'étoffe
  - ☉ Apache évolue => mises à jour successives
  - ☉ rotation des sysadmins

Question :

- ☉ comment un sysadmin nouvellement affecté peut-il s'approprier un serveur Apache en activité ?
- ☉ comment évaluer le niveau de sécurité induit par la configuration en place ?
- ☉ comment contenir l'activité des webmasters ?

# *Introduction*

- Cet exposé
  - se place du point de vue du sysadmin hébergeur de sites web et d'une communauté de webmasters
  - se focalise sur quelques aspects de la configuration de base d'Apache et de son environnement système (UNIX)
  - n'est pas un panorama des possibilités de configuration d'Apache impactant la sécurité (391 directives de configuration pour Apache 2.2)
  - n'a donc pas la prétention d'être complet ou exhaustif



# *Plan*

- Généralités sur la configuration d'Apache
- Déterminer l'espace web sous contrôle
- Contenir les débordements de l'espace web par défaut
- Restreindre l'espace web
- Identifier les scripts activables
- Contrôler le périmètre d'action des scripts
- Etanchéifier les territoires des webmasters



# Vocabulaire

- Espace web (*URL-space*) : fichiers et répertoires du filesystem accessibles par HTTP
- Webmaster : un compte, déclaré sur le serveur, ayant des droits d'écriture sur une partie de l'espace web (hors pages perso)
- Utilisateur : un compte, déclaré sur le serveur, ayant au moins la possibilité d'écrire des pages perso



# Généralités sur la configuration d'Apache

# *Fichiers de configuration*

- Un fichier de config principal
  - sous contrôle de root
  - hors de l'espace web
- Des fichiers de config inclus
  - sous contrôle de root
  - hors de l'espace web
- Des fichiers .htaccess,
  - sous contrôle des webmasters
  - dans l'espace web





# *Fichiers de configuration*

- root peut activer/désactiver/limiter l'usage des .htaccess
- Une modification du fichier de config principal nécessite un redémarrage d'Apache
- Toute modification d'un .htaccess est prise en compte instantanément
- L'activation des .htaccess implique un travail supplémentaire pour Apache

# *Syntaxe de la configuration*

- Fichier de config principal
  - le nom par défaut est défini à la compil
  - il peut être spécifié en ligne de commande
- Il contient des lignes de la forme :

```
directive arguments
```

```
<section>  
    directive arguments  
    directive arguments  
</section>
```

```
<section>  
    <section>  
        directive arguments  
        directive arguments  
    </section>  
</section>
```

# Section <Directory>

```
<Directory /var/www/html>  
    directive ...  
</Directory>
```

```
<Directory /home/*/public_html>  
    directive ...  
</Directory>
```

```
<DirectoryMatch "^/www/(.+/?)[0-9]{3}">  
    directive ...  
</DirectoryMatch>
```

- Les directives s'appliquent aux répertoires s'identifiant à l'expression et à ses sous-répertoires
- Pas appliquées si accès via un chemin différent (symlink)
- S'appliquent dans l'ordre de la correspondance de la plus courte à la plus longue : /var, /var/www, /var/www/html



## *Section <Files>*

- Les directives s'appliquent aux objets ayant un basename s'identifiant à l'expression
- Les sections sont appliquées dans l'ordre d'apparition dans le fichier de conf
- Peut être incluse dans une section <Directory>

```
<Files .htaccess>  
  ...  
</Files>
```

```
<FilesMatch "\.(gif|jpe?g|png)$">  
  ...  
</FilesMatch>
```

## Section <Location>

```
<Location /status>  
    SetHandler server-status  
</Location>
```

- Les directives s'appliquent à une URL
- Utilisée pour un contenu résidant hors du filesystem
- Appliquées dans l'ordre d'apparition dans le fichier de conf
- Prioritaires sur <Directory> et <Files> 



<Location /> est un moyen commode d'appliquer une directive à tout l'espace web

## Section <Limit>

```
<Limit POST PUT DELETE>  
  Require valid-user  
</Limit>
```

```
<LimitExcept GET>  
  Require valid-user  
</Limit>
```

- GET, POST, PUT, DELETE, CONNECT, OPTIONS, PATCH, PROPFIND, PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK
- Noms des méthodes sensibles à la casse 
- Peut apparaître dans <Directory>



## *Section <VirtualHost>*

```
<VirtualHost 10.1.2.3>  
  DocumentRoot /www/docs/host.foo.com  
  ServerName host.foo.com  
</VirtualHost>
```

```
<VirtualHost [2001:db8::a00:20ff:fea7:ccea]>  
  ...  
</VirtualHost>
```



# *Imbrication des sections*

```
<VirtualHost ...>  
  <Directory ...>  
    <Files ...>  
      <Limit ...>  
        ...  
      <Limit ...>  
    </Files>  
  </Directory>  
</VirtualHost>
```

# *.htaccess*

- Lors du traitement d'une requête, Apache cherche la présence d'un fichier `.htaccess` dans tous les répertoires du chemin menant au document
- Exemple, avant de retourner `/usr/local/web/index.html`, Apache examine les fichiers :
  - `/.htaccess`,
  - `/usr/.htaccess`,
  - `/usr/local/.htaccess`
  - `/usr/local/web/.htaccess`
- Mieux vaut désactiver cette fonctionnalité sur / :

```
<Directory />  
    AllowOverride None  
</Directory>
```



# *.htaccess*

- Le fichier `/var/www/.htaccess` :

```
directive1  
directives2
```

est équivalent à :

```
<directory /var/www>  
  directive1  
  directive2  
</directory>
```

# Contexte des directives

- A chaque directive est associé un contexte d'utilisation :
- server config : hors de tout contexte, dans le fichier de configuration principal
- virtual host : dans une section <VirtualHost>
- directory : dans une section <Directory>, <Location> ou <Files>
- .htaccess : dans un fichier .htaccess

## AllowOverride Directive

**Description:** Types of directives that are allowed in .

**Syntax:** AllowOverride All|None|*directiv*

**Default:** AllowOverride All

**Context:** directory

**Status:** Core

**Module:** core

## Satisfy Directive

**Description:** Interaction between host-level access control and user authentication

**Syntax:** Satisfy Any|All

**Default:** Satisfy All

**Context:** directory, .htaccess

**Override:** AuthConfig

**Status:** Core

**Module:** core

**Compatibility:** Influenced by [<Limit>](#) and [<LimitExcept>](#) in version 2.0.51 and later

# Priorité des sections

- A connaître, car les conséquences sont importantes 
  1. <Directory> et .htaccess  
pour un niveau donné, .htaccess prévaut sur <directory> 
  2. <DirectoryMatch>
  3. <Files> et <FilesMatch>
  4. <Location> et <LocationMatch>
- Sinon, chaque groupe identique est traité suivant l'ordre d'apparition

# Priorité des sections

- Les sections dans <VirtualHost> sont appliquées après les sections correspondantes globales
- Ne pas abuser des imbrications. Exemple de mauvaise idée :

```
<VirtualHost ...>  
  <Directory ...>  
    ...  
  </Directory>  
</VirtualHost>
```



- Attention : <Location> a le dernier mot



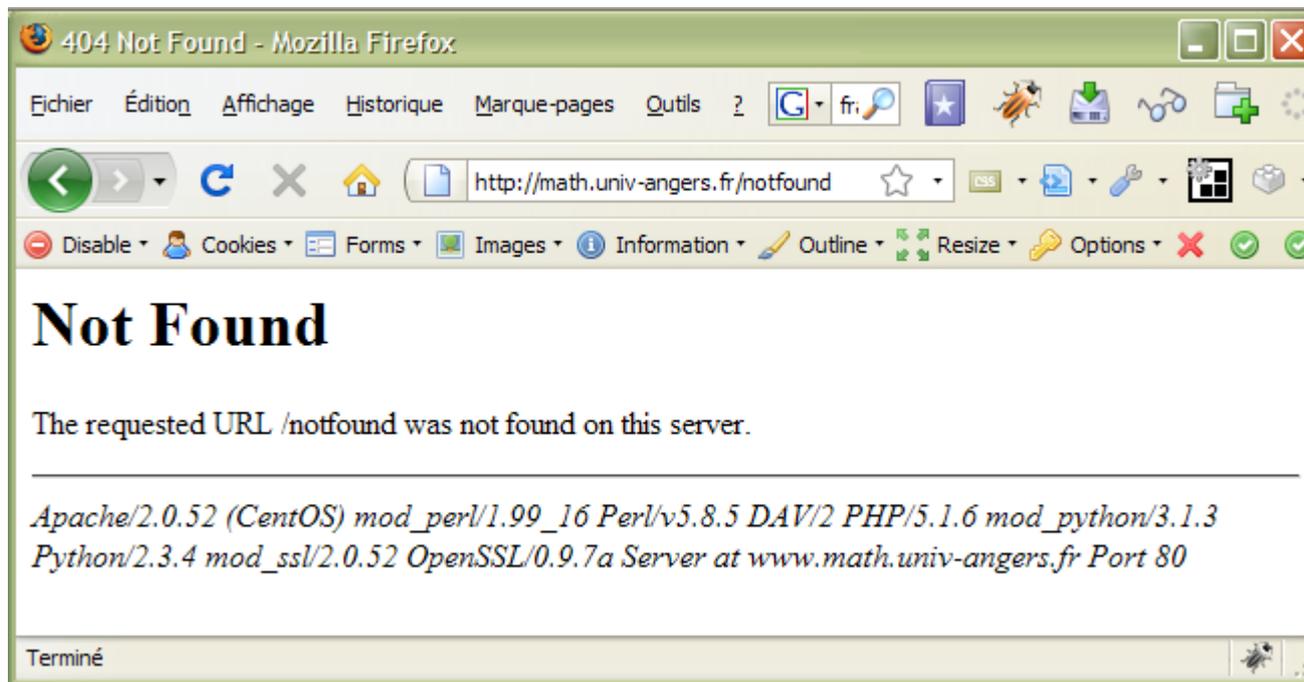
```
<Location />  
  order deny,allow  
  allow from all  
</Location>
```

# Exemple

## ServerSignature

default: ServerSignature Off (apache 2.2)

context: server config, virtual host, directory, .htaccess



# Exemple

## ServerTokens

default: ServerTokens Full

context: server config

```
$ telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Sat, 02 Jun 2001 13:11:40 GMT
Server: Apache/1.3.14 (Unix) (Redhat-
PHP/4.0.3pl1 mod_perl/1.24
Connection: close
Content-Type: text/html

Connection closed by foreign host.
```



ServerSignature On  
ServerTokens Prod





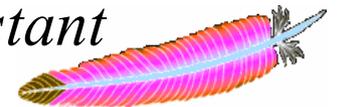
# Déterminer l'espace web sous contrôle

## *L'espace web (URL-space)*

- Ensemble des répertoires/fichiers qu'Apache peut servir
- Les fichiers/répertoires inaccessibles par l'UID sous lequel tourne Apache sont hors de l'espace web (sauf utilisation de suExec)
- Beaucoup de fichiers accessibles en lecture par tous n'ont pas vocation à être servis par Apache : /etc/\*, /home/\*, /proc/\*...
- Parades :
  - chroot / virtualisation
  - restreindre les droits d'accès via l'OS : les droits standard Unix ne permettent pas d'affecter des permissions application par application (u/g/o insuffisants)
  - utiliser les contrôles d'accès d'Apache

# User & Group

- définissent l'identité sous laquelle le serveur répond aux requêtes
- contexte : server config
- normalement, le serveur httpd est lancé par root
  - le process initial reste sous l'identité root, et
  - les process fils prennent l'identité spécifiée par User:Group
- « *L'identité spécifiée ne doit avoir aucun privilèges lui permettant d'accéder à des fichiers qui n'ont pas à être visible hors du serveur, ni d'exécuter du code sans rapport avec le traitement de requêtes HTTP. Il est déconseillé d'utiliser un compte déjà existant (nobody), qui peut servir à autre chose* »





# *User & Group*

- Donc créer un compte et un groupe spécifique pour Apache
- Si le serveur n'est pas lancé par root,
  - il ne peut pas changer l'identité de ses fils,
  - et par conséquent sert les documents sous l'identité qui l'a lancé.
- depuis Apache 2, User et Group ne peuvent plus être utilisés dans un contexte VH

# *Espace web principal*

- **ServerRoot**
  - répertoire à partir duquel Apache s'est déployé
  - contient par défaut des répertoires conf, logs, bin, htdocs...
  - sert de répertoire de base pour tout chemin relatif de la configuration
- **DocumentRoot**
  - définit le répertoire racine de l'espace web principal
  - dans le cas général (hors Alias par exemple), Apache ajoute à DocumentRoot le chemin requis dans l'URL pour obtenir le chemin du document à servir

```
ServerRoot /usr/local
```

```
DocumentRoot web
```

# *Espace web principal*

- DocumentRoot peut apparaître dans un contexte VirtualHost
- espace web principal =  $\cup$  espaces web principaux  
vh
- ServerRoot et DocumentRoot possibles que dans la config principale,
- 👉 ● donc la définition de l'espace web principal est sous contrôle du sysadmin

# *Espace des pages perso*

- UserDir définit la racine des pages perso
- Mapping de `http://www.x.fr/~gaston/page.html`

## Userdir

public\_html

/var/pages\_perso

/var/\*/pages\_perso

http://autre.fr/~\*/

## mapping

~gaston/public\_html/page.html

/var/pages\_perso/gaston/page.html

/var/gaston/pages\_perso/page.html

http://autre.fr/~gaston/page.html

# *Espace des pages perso*

- Il est recommandé de désactiver les pages perso de root

```
UserDir disabled
```

```
UserDir disabled root
```

```
UserDir disabled  
UserDir enable user1 user2 user3
```

```
UserDir enabled  
UserDir disabled root nobody apache
```

- UserDir peut apparaître dans un contexte VH

- Espace pages perso =  $\cup$  espace pages perso  
vh

- UserDir interdit dans .htaccess, donc la définition des espaces pages perso est sous contrôle



# Alias

- ☉ Rattache n'importe quelle arborescence de répertoires du filesystem à l'espace web

```
Alias /doc /usr/linux/docs
```

- ☉ <http://www.exemple.fr/doc/page.html>

mappé en : </usr/linux/docs/page.html>

- ☉ ScriptAlias fait la même chose, et spécifie en plus que tous les fichiers doivent être traités comme des CGI
- ☉ AliasMatch, ScriptAliasMatch
- ☉ Alias interdit dans .htaccess, donc l'extension de l'espace web via des alias est sous contrôle



# *Espace web total*

- espace web total  $\equiv$ 
  - $\cup_{vh} \text{DocumentRoot} + \cup_{vh} \text{UserDir} + \cup \text{Alias}$
- sous contrôle du sysadmin 
- trouver le fichier de conf principal, puis

```
grep -i '^\\s*include' httpd.conf # récursivement
grep -i '^\\s*ServerRoot' *.conf
grep -i '^\\s*DocumentRoot' *.conf
grep -i '^\\s*UserDir' *.conf
grep -i '^\\s*Alias' *.conf
grep -i '^\\s*ScriptAlias' *.conf
```

# *Config par défaut (tarball)*

```
$configure --prefix /usr/local/apache  
$make
```

```
$ grep -i '^ *include' httpd.conf
```

```
$ grep -i '^ *ServerRoot' *.conf  
ServerRoot "/usr/local/apache"
```

```
$ grep -i '^ *DocumentRoot' *.conf  
DocumentRoot "/usr/local/apache/htdocs"
```

```
$ grep -i '^ *UserDir' *.conf
```

```
$ grep -i '^ *Alias' *.conf
```

```
$ grep -i '^ *ScriptAlias' *.conf  
ScriptAlias /cgi-bin/ "/usr/local/apache/cgi-bin/"
```

# Config par défaut (Fedora)

```
$ grep -i '^\\s*include' httpd.conf
Include conf.d/*.conf

$ grep -i '^\\s*ServerRoot' *.conf
ServerRoot "/etc/httpd"

$ grep -i '^\\s*DocumentRoot' *.conf
DocumentRoot "/var/www/html"

$ grep -i '^\\s*UserDir' *.conf
UserDir disable

$ grep -i '^\\s*Alias' *.conf
Alias /icons/ "/var/www/icons/"
Alias /error/ "/var/www/error/"

$ grep -i '^\\s*ScriptAlias' *.conf
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
```



# Contenir les débordements de l'espace web par défaut

# *Liens symboliques*

- Les liens symboliques situés dans l'espace web ouvrent des brèches vers le filesystem
- Hors contrôle du sysadmin 
- Comment les limiter ?

```
<Directory /var/www/html>  
  Options FollowSymlinks  
</Directory>
```

```
<Directory /var/www/html>  
  Options FollowSymlinksIfOwnerMatch  
</Directory>
```

# *Héritage des options*

```
<Directory dir>  
  Options [+|-]option [[+|-]option] ...  
</Directory>
```

## ● Sans + ni -, Options écrase les options héritées

```
<Directory /var/www>  
  Options MultiViews # les liens ne seront pas suivis  
</Directory>
```

## ● Avec + ou -, Options modifie les options héritées

```
<Directory /var/www>  
  Options -MultiViews # les liens seront peut-être suivis  
</Directory>
```

# *Héritage des options*

```
<Directory /htdocs>  
  Options -FollowSymlinks  
</Directory>
```

- Les symlinks sont-ils pour autant désactivés ?
  - pas forcément, s'il existe un fichier .htaccess contenant :

```
Options FollowSymlinks
```

- Comment contrôler les .htaccess ?

# *.htaccess*

- Est-ce bien .htaccess ? Vérifier :

```
AccessFileName .htaccess
```

- le vérifier pour tous les vh 

- Même si tout semble « normal », le vérifier quand même :



```
AccessFileName .htaccess readme
```

- Activer/désactiver les .htaccess : AllowOverride

```
<Directory />  
  AllowOverride None  
</Directory>
```

```
<Directory /var/www/html/permiffif>  
  AllowOverride All  
</Directory>
```

- 
- Autoriser les webmasters à activer le suivi des symlinks dans leurs répertoires :

```
AllowOverride All
```

```
AllowOverride Options
```

- et depuis apache 2.2 :

```
AllowOverride Options=FollowSymlinks
```

```
AllowOverride Options=FollowSymlinksIfOwnerMatch
```

# Config par défaut (tarball & Fedora)

```
<Directory />
  Options FollowSymLinks
  AllowOverride None
</Directory>

<Directory "/usr/local/apache/htdocs">
  Options Indexes FollowSymLinks
  AllowOverride None
</Directory>

<Directory "/usr/local/apache/cgi-bin">
  AllowOverride None
  Options None
</Directory>
```

- Pourquoi ce laxisme ? Explication personnelle :
  - ne pas suivre les symlinks (ou les suivre si propriétaires identiques) est coûteux : 1 appel de lstat pour chaque répertoire du chemin et pour le fichier final



# Restreindre l'espace web

# *filtrage sur la source*

## ● Allow, deny

```
Allow from apache.org
Allow from .net
Allow from 10.1.2.3
Allow from 10.1
Allow from 10.1.0.0/255.255.0.0
Allow from 10.1.0.0/16
Allow from 2001:db8::a00:20ff:fea7:ccea/10
```

```
SetEnvIf User-Agent ^Firefox/2\.0 il_passe
<Directory /docroot>
    Allow from env=il_passe
</Directory>
```

## ● Allow et deny n'ont de sens que si l'on connaît la valeur de order

# *filtrage sur la source*

## ● Order deny,allow

tout est autorisé par défaut, sauf ce qui est interdit, à moins que ce soit autorisé

```
Order Deny,Allow  
Deny from ennemi.com  
Allow from agent-double.ennemi.com
```

## ● Order allow,deny

tout est interdit par défaut, sauf ce qui est autorisé, à moins que ce soit interdit

```
Order Allow,Deny  
Allow from partenaire.fr  
Deny from agent-double.partenaire.fr
```

## ● Autoriser le positionnement de order, allow et deny dans un .htaccess :

```
AllowOverride Limit
```



# *filtrage par authentication*

```
<Location /secure>
  AuthType basic
  AuthName "private area"
  AuthBasicProvider dbm
  AuthDBMType SDBM
  AuthDBMUserFile /www/etc/dbmpasswd
  Require valid-user
</Location>
```

# Directive Satisfy

- Valeurs : All ou Any
  - satisfy all : require ET allow/deny
  - satisfy any : require OU allow/deny
- Exemple :

```
$ cat /htdocs/.../x/.htaccess
Order Allow,Deny
Allow from all
Satisfy Any
```

```
<Directory /htdocs/.../x/.../y>
  Require valid-user
</Directory>
```

l'authentification n'a pas lieu !



- Autoriser Satisfy dans .htaccess :

```
AllowOverride AuthConfig
```

# *Option Indexes*

- Lorsqu'une URL chemine vers un répertoire
  - si ce répertoire contient un index.html (vérifier quand-même DirectoryIndex), c'est ce fichier qui est renvoyé
  - sinon
    - si l'option Indexes est positionnée, la liste des fichiers et répertoires de ce répertoire est retournée
    - sinon, une erreur est retournée



- Autoriser Options Indexes dans .htaccess :

```
AllowOverride Options
```

```
AllowOverride Options=Indexes
```

- Autoriser DirectoryIndex dans .htaccess :

```
AllowOverride Indexes
```

# *Interdire des cibles précises*

```
<Directory />  
  <FilesMatch "^\. ">  
    order deny,allow  
    deny from all  
  </FilesMatch>  
</Directory>
```

```
<LocationMatch "/\.\."/ >  
  order deny,allow  
  deny from all  
</LocationMatch>
```

```
RewriteEngine On  
RewriteLog logs/rewrite.log  
RewriteLogLevel 9
```

```
RewriteRule /\. / / [F]  
RewriteRule /\.\. / / [F]  
RewriteRule etc/passwd / [F]  
RewriteRule etc/shadow / [F]
```



# *Configuration initiale*

```
<Directory />
  Order deny,allow
  Deny from all
</Directory>

<Directory "/usr/local/apache/htdocs">
  Order allow,deny
  Allow from all
</Directory>

<FilesMatch "^\.ht">
  Order allow,deny
  Deny from all
  Satisfy All
</FilesMatch>

<Directory "/usr/local/apache/cgi-bin">
  Order allow,deny
  Allow from all
</Directory>
```



# Identifier les scripts activables



# Action

- Une action peut être :
  - liée statiquement au serveur,
  - ajoutée comme un module,
  - ajoutée avec une directive Action
  - définie en tant que filtre



## *Actions prédéfinies*

- default-handler : envoie le contenu statique du fichier
- send-as-is : envoie un fichier contenant ses propres entêtes HTTP
- cgi-script : traite le fichier comme un CGI
- imap-file : traite le fichier comme une image map
- server-info : renvoie des infos sur la config
- server-status : renvoie des infos sur l'état du serveur



# *Directive Action*

- Définir une action et l'associer à un type MIME

```
Action image/gif /cgi-bin/images.cgi
```

- Définir et nommer une action (sans association)

```
Action add-footer /cgi-bin/footer.pl
```



# *Handler*

- Déclencheur d'actions
- Active une action en fonction :
  - d'un type de contenu
  - d'une extension de fichier
  - d'un emplacement de fichier

# *Directive AddHandler*

- Crée un handler basé sur une extension de fichier
- Contexte : server config, virtual host, directory, .htaccess 
- Exemples : pour que les fichiers .html soient traités par le script add-footer.pl :

```
Action add-footer /cgi-bin/footer.pl  
AddHandler add-footer .html
```

# Directive SetHandler

- Créer un handler pour tous les fichiers (inconditionnel)
- Contexte : server config, virtual host, directory, .htaccess



- Exemple :

```
Action gif-handler /cgi-bin/images.cgi  
  
<Directory /htdocs/images>  
    SetHandler gif-handler  
</Directory>
```

- Interdire AddHandler et SetHandler dans .htaccess  
ne pas spécifier : AllowOverride FileInfo



# *Filtres*

- Spécifie des traitements à exécuter avant ou après l'action
  - SetInputFilter
  - AddInputFilter
  - SetOutputFilter
  - AddOutputFilter
  - AddOutputFilterByType

# *Conf par défaut*

## ● config initiale (tarball) :

```
$ grep -Ei "handler|action|filter" httpd.conf
#AddHandler cgi-script .cgi
#AddHandler type-map var
#AddOutputFilter INCLUDES .shtml
```

## ● config initiale (Fedora) :

```
$ grep -Ei "handler|action|filter" httpd.conf
#AddHandler cgi-script .cgi
AddHandler type-map var
AddOutputFilter INCLUDES .shtml
```



# Contrôler le périmètre d'action des scripts

- 
- Où sont les scripts ?
  - Que font les scripts ?
  - Que consomment les scripts ?

# SSI

- Permet d'insérer du code dans les pages HTML
- Exemple :

```
$ cat ls.html
```

```
<html>  
  <body>  
    <!-- #exec cmd="ls ~`echo $QUERY_STRING| sed 's/^.*=//'`" -->  
  </body>  
</html>
```

permet d'obtenir la liste des fichiers d'un utilisateur par :

<http://servername/ls.html?user=tartempion>

- Autorisé par :

```
Options Includes
```

# SSI

- Plus anodin :

```
<!-- #printenv -->
```

```
DOCUMENT_ROOT=/usr/local/etc/httpd/htdocs  
PATH=/bin:/sbin:/usr/bin:/usr/sbin  
SCRIPT_FILENAME=/usr/local/httpd/htdocs/support/printenv.html  
SERVER_SOFTWARE=Apache/1.3.37 (Unix)  
USER_NAME=root  
SERVER_SIGNATURE=
```

- Autorisé par :

```
Options Includes
```

```
Options IncludesNoExec
```

- Possible à spécifier dans .htaccess si :

```
AllowOverride Options  
AllowOverride Options=Includes  
AllowOverride Options=IncludesNoExec
```

- Même avec `Options IncludesNoExec`, toujours possible de lancer un script avec `<!--#include virtual="/cgi-bin/xx.pl" -->`

# Où sont les CGI ?

- Répérer dans la config :
- les définitions de handlers :

```
SetHandler cgi-script
```

```
AddHandler cgi-script .cgi
```

- les options d'exécution :

```
ScriptAlias /cgi-bin/ /web/cgi-bin/
```

équivalent à :

```
Alias /cgi-bin/ /web/cgi-bin/  
<Location /cgi-bin >  
    SetHandler cgi-script  
    Options +ExecCGI  
</Location>
```

- *Mettre les scripts CGI hors de l'espace web principal (pas sous DocumentRoot) pour éviter que le code source soit accidentellement exposé xx.cgi.~ xx.cgi.bak*



# *Où sont les autres scripts ?*

## ● Exemple : PHP

```
LoadModule php5_module modules/libphp5.so  
AddHandler php5-script .php  
AddType text/html .php
```

- pas de localisation particulière
- pas d'options nécessaires pour les répertoires

## *Attention : extensions multiples*

- Les fichiers peuvent avoir plusieurs extensions 
-  L'ordre des extensions n'est pas significatif. Exemples :
  - si le fichier exemple.html.fr est associé au content-type: text/html et au content-language fr, alors le fichier exemple.fr.html sera considéré de façon identique
  - si .html est associé au type MIME text/html et .cgi associé au handler cgi-script, alors x.cgi.html sera traité comme un CGI (de même que x.html.cgi)



## *Attention : extensions multiples*

- Pour éviter ce comportement, ne pas utiliser Add\* directives :

```
AddHandler cgi-script .cgi
```

- Par exemple, pour que x.html.cgi doit considéré comme un CGI, mais pas x.cgi.html :

```
<FilesMatch \.cgi$>  
    SetHandler cgi-script  
</FilesMatch>
```

# *Identité des processus*

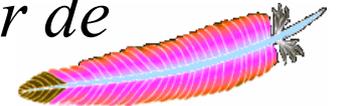
- Tous les scripts (SSI, CGI) tournent sous une même identité (celle du serveur Apache)
  - => un script défaillant peut impacter les données utilisées par les scripts des autres webmasters
- Précaution (nécessaire mais insuffisante)
  - le compte Apache doit être :
    - désactivé
    - sans shell



# *suExec*

- suExec exécute les CGI sous l'identité du propriétaire du fichier et non pas sous l'identité du serveur web

*Utilisé proprement, suExec réduit considérablement les risques induits par le développement de CGI par les webmasters. Mal utilisé, il peut par contre ouvrir de nouveaux trous de sécurité.*



- SuExec est un wrapper entre Apache et le CGI
- Apache appelle le wrapper avec le nom du script à exécuter, l'UID et le GID sous lequel il doit s'exécuter

# *SuExec : mise en oeuvre*

- Nécessite un binaire suexec

```
$ ls -l /usr/sbin/suexec  
-r-s--x--- 1 root apache /usr/sbin/suexec
```

- Installé avec apache sur Fedora, CentOS

- Tarball :

```
./configure --enable-suexec
```

- Au lancement de httpd :

```
[notice] suEXEC mechanism enabled (wrapper: /path/to/suexec)
```

- 2 utilisations :

- par virtual hosts

- par userdir

# *SuExec par Virtual Host*

```
<VirtualHost ...>  
    SuexecUserGroup userA groupB  
</VirtualHost>
```

- suExec vérifie :
  - que le path du CGI ne commence par / et ne contient pas ..
  - que ni le user, ni le groupe ne sont root
  - que l'UID (resp GID) est  $>$  au minimum UID (resp GID)
  - que seul l'utilisateur a un droit d'écriture dans le répertoire
  - que seul l'utilisateur a un droit d'écriture sur le script
  - que l'utilisateur/groupe de SuExecUserGroup correspond au propriétaire/groupe du script
- Si OK, suExec prend l'identité et le groupe mentionné



# *SuExec par Virtual Host*

- Si 1 webmaster unique par virtual host => OK
- Sinon, pas exploitable (avis personnel)

# *SuExec par UserDir*

- Suexec exécute le CGI sous l'identité du propriétaire du userdir
- suExec vérifie :
  - que l'UID (resp GID) est  $>$  au minimum UID (resp GID)
  - que seul l'utilisateur a un droit d'écriture dans le répertoire
  - que seul l'utilisateur a un droit d'écriture sur le script
  - que l'utilisateur/groupe propriétaire du userdir correspond au propriétaire/groupe du script
- Si OK, suExec prend l'identité et le groupe du UserDir
- => à utiliser sans modération

# *Contrôler les consommations*

- Ajuster le nombre de process/threads

StartServers, MinSpareServers, MaxSpareServers,  
StartThreads, MinSpareThreads, MaxSpareThreads,  
ThreadsPerChild, MaxThreads, ThreadLimit, ServerLimit,  
MaxRequestsPerChild, MaxKeepAliveRequests

- Ajuster les timeout

TimeOut, KeepAliveTimeout

- Ajuster la taille et le nb max des requêtes

MaxClients, LimitRequestBody, LimitXMLRequestBody,  
LimitRequestFields, LimitRequestFieldSize,  
LimitRequestLine

- Adapter le fonctionnement du serveur

LimitInternalRecursion, RLimitCPU, RLimitMEM, RLimitNPROC

- => à ne pas modifier à la légère

# Contrôler les consommations

- HostnameLookups

- off par défaut

- logresolve pour traiter les logs

- attention : sur les anciennes versions off =>



deny from nom.domain

jamais vérifié

- context: server config, virtual host, directory

```
HostnameLookups off
<Files ~ "\.(html|cgi)$">
  HostnameLookups on
</Files>
```

# *Contrôler les consommations*

## ● /robots.txt

```
User-agent: *  
Disallow: /cgi-bin/  
Disallow: /tmp/  
Disallow: /~joe/
```

```
User-agent: Google  
Disallow:
```

```
User-agent: *  
Disallow: /
```



# Etanchéifier les territoires des webmasters

- 
- Objectif : limiter le rayon d'action des webmasters
  - Le minimum :
    - apache doit pouvoir lire tous les espaces web de tous les vh
    - les webmasters d'un site doivent
      - pouvoir écrire dans leur espace
      - avoir le minimum de droit sur les espaces des autres sites

# *Solution 1*

- Créer un groupe wmsite par site
  - y mettre tous les webmasters du site
  - propriétaire des fichiers : créateur:wmsite
  - droits : rwxrwxr-x
- => tout le monde a accès en lecture
  - accès aux .htaccess, .htpasswd, source des scripts

## *Solution 2*

- Créer un groupe wmsite par site
  - y mettre tous les webmasters du site + apache
  - propriétaire des fichiers : créateur:wmsite
  - droits : rwxrwx---
- => apache à accès en écriture à tous les fichiers

# Les ACL



- Possibilités :
  - donner des droits pour divers utilisateurs
  - donner des droits pour divers groupes
- Ici :
  - définir un groupe de webmasters wmsite par site
  - pour le DocumentRoot d'un site :
    - propriétaire : root:wmsite
    - droits :
      - propriétaire : rwx
      - wmsite : rwx
      - user ou groupe apache : r-x
      - autre : ---

# ACL : mode d'emploi

- Activer les ACL sur la partition :

```
$ grep htdocs /etc/fstab
```

```
LABEL=htdocs          /htdocs  ext3      defaults,acl  1 2
```

- Activation initiale (sans démontage)

```
mount -o remount,acl -L htdocs
```

- Positionner les ACL sur un fichier :

```
setfacl -m u::rw -m g:apache:r -m g:wmsite:rw -m o:: page.html
```

- Positionner les ACL sur une arborescence de répertoires :

```
setfacl -R \  
-m u::rwX -m g:apache:r-X -m g:wmsite:rwX -m o:: \  
-m d:u::rwx -m d:g:apache:r-x -m d:g:wmsite:rwx -m d:o:: \  
/htdocs/site
```

# Consultation des ACL

```
$ getfacl /htdocs/site
```

```
user::rwx  
group::rwx  
group:apache:r-x  
group:wmsite:rwx  
mask::rwx  
other:---  
default:user::rwx  
default:group::r-x  
default:group:apache:r-x  
default:group:wmsite:rwx  
default:mask::rwx  
default:other:---
```

```
$ getfacl /htdocs/site/index.html
```

```
user::rw-  
group::rw-  
group:apache:r--  
group:wmsite:rw-  
mask::rwx
```



# Conclusion

# *Sources d'inspiration*

## Expérience issue de la réorganisation du web au LAL IN2P3 (2003-2004)

- ☉ au départ : un cluster central disposant
  - ☉ d'un SAN pour tout le stockage du labo : homedir, web, données d'expérience, etc.
  - ☉ de tous les utilisateurs du labo (plusieurs centaines de comptes)
  - ☉ d'un site web monolithique
- ☉ après réorganisation :
  - ☉ éclatement de l'espace web en 120 vh (120 DocumentRoot *étanches*)

## Expérience issue de la mise en place de la plate-forme d'hébergement de sites web de Mathrice (2007)

# Références

- [http://httpd.apache.org/docs/2.2/misc/security\\_tips.html](http://httpd.apache.org/docs/2.2/misc/security_tips.html)
- <http://httpd.apache.org/docs/2.2/misc/perf-tuning.html>
- <http://www.hsc.fr/ressources/>
- <http://www.w3.org/Security/>
- Apache Security - Ivan Ristic - O'Reilly

*La dernière version de ce document se trouve sur* <http://larema.math.cnrs.fr/~charbonnel>