Protection d'un serveur Apache

Jacquelin Charbonnel - CNRS LAREMA

(modifié M. Libes pour la formation ADF en DR12 Fév. 2009)

ADF- Aide à la Détection des Faiblesses d'un site web

Journées UREC - Montpellier - Septembre 2008

version 8

Introduction

- un serveur Apache fraîchement installé possède un niveau de sécurité « satisfaisant »
  - au fil du temps : le nombre de documents croît, la configuration s'étoffe et est modifiée, la sécurité de l'ensemble diminue!
    - Rajout de Virtual Hosts, de machines Virtuelles
    - Installation de nouveaux logiciels web
    - Ajouts de nouveaux répertoires et fichiers de test qu'on oublie
    - Développement de scripts
    - Possibilité de sysadmins différents
- Comment sécuriser le serveur Apache au fil du temps comment avoir une vue synoptique d'un serveur Apache en activité ?
    - comment évaluer le niveau de sécurité induit par la configuration en place ?
    - Problèmes plus importants si plusieurs webmaster

Introduction

- Cet exposé
  - A pour but de donner une vision de la sécurité « l'espace Web » sous contrôle d'Apache, et le contenir
  - se focalise sur quelques aspects de la configuration de base d'Apache et de son environnement système sous Unix/Linux
    - 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

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

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

Fichiers de configuration

- Un fichier de config principal /etc/apache2/apache2.conf
  - 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 fichiers .htaccess
  - *Une modification du fichier de config principal nécessite un redémarrage d'Apache*
  - *... Alors que toute modification d'un .htaccess est prise en compte instantanément par le serveur*
- L'activation des .htaccess implique un travail supplémentaire pour Apache (scruter les répertoires)

Syntaxe de la configuration

- Fichier de configuration 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>
```

Configuration d'Apache

- Le paramétrage d'apache permet de désigner
- Des répertoires : <Directory>
- Des URL : <Location>
- Des fichiers : <Files>
- Des « Hôtes virtuels » *(serveurs web virtuels = meme adresse IP, mais nom différent)*

Section <Directory>, <DirectoryMatch>

```
<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> <FilesMatch>

- 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 de <Location> s'appliquent à une URL
  - Utilisée pour un contenu résidant dans l'espace Web
- 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>
```

- <Limit> pour limiter l'accès à des espaces Web
- 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 [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

- Les fichiers .htaccess se placent dans des répertoires
- Une directive dans le fichier /var/www/.htaccess

```
directive1
diretives2
```

est équivalent à la même directive placée dans le fichier de configuration principal de pache, dans une section <Directory>:

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

Contexte des directives

- Les directives apache fonctionnent dans un certain « contexte » d'utilisation (l'endroit ou peut apparaître une directive):
  - « server config » : dans le fichier de configuration principal, hors de tout contexte
  - « 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 i
Syntax:	`AllowOverride All|None|direct`
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 :

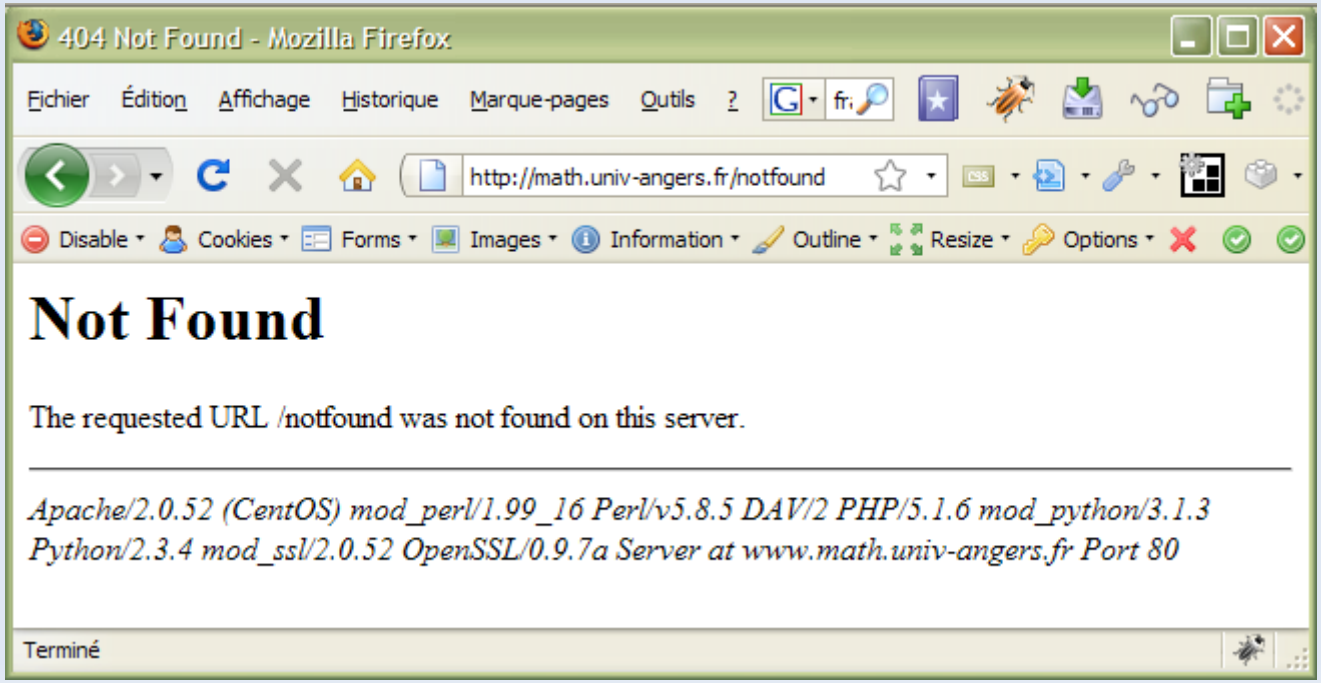
```
<VirtualHost ...>
    <Directory ...>
        ...
    </Directory>
</VirtualHost>
```

- Attention : <Location> a le dernier mot

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

La signature des serveurs

- ServerSignature [On | Off]
- Attention : « On » donne trop d' informations interessantes en cas de failles
  - default: ServerSignature Off (apache 2.2)
  - context: server config, virtual host, directory, .htaccess



Signature des Serveurs

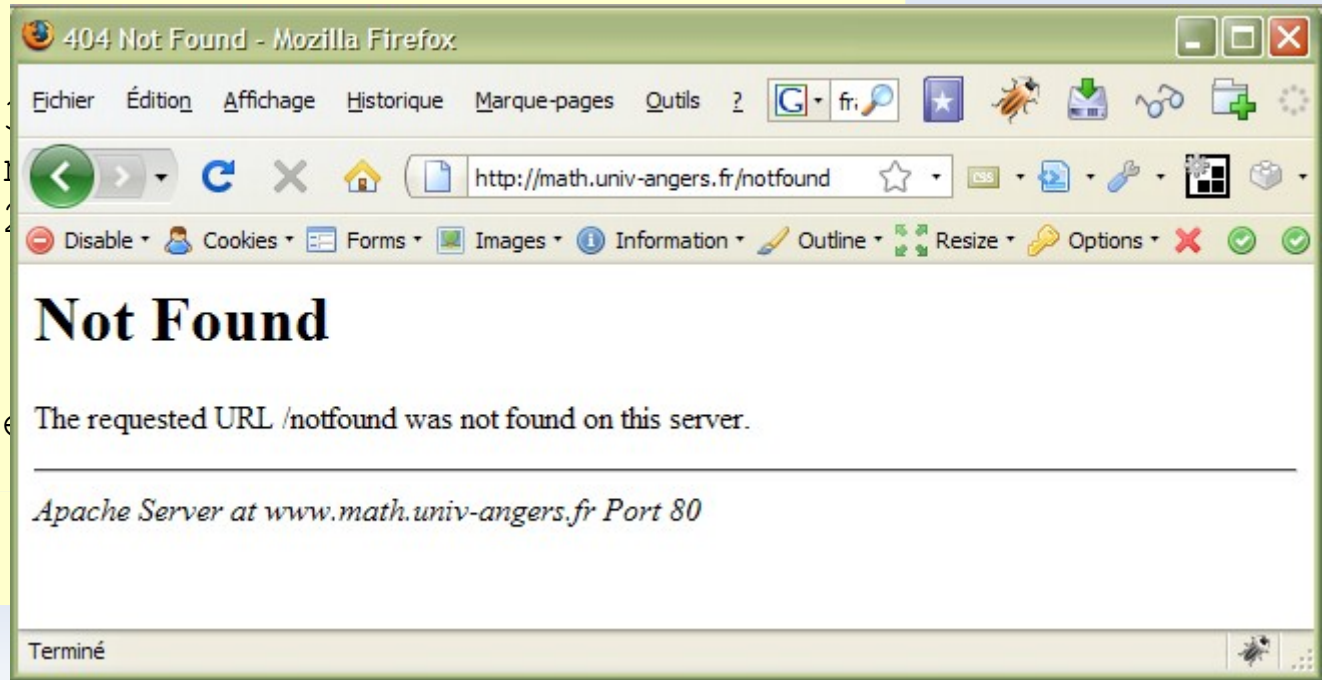
- ServerTokens
  - ServerTokens Major|Minor|Min[imal]|Prod[uctOnly]|OS|Full
  - default: ServerTokens Full
  - context: server config

```
$ 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 1
Server: Apache/1.3.14 (U
PHP/4.0.3pl1 mod_perl/1.
Connection: close
Content-Type: text/html

Connection closed by fore
```

```
ServerSignature On
ServerTokens Prod
```



Plan

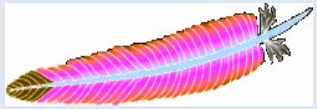
- Généralités sur la configuration d'Apache
- **Déterminer l'espace web sous contrôle**
- *Contenir les débordements de l'espace web par défaut*
- *Restreindre l'espace web*
- *Identifier les scripts activables*
- *Contrôler le périmètre d'action des scripts*
- *Etanchéifier les territoires des webmasters*

L'espace web (URL-space)

- Ensemble des répertoires/fichiers qu'Apache peut servir
  - Les répertoires/fichiers/ qui n'ont pas vocation a être servi par Apache doivent être inaccessibles par l'UID sous lequel tourne Apache : ils sont hors de l'espace web (sauf utilisation de suExec)
- Or, il existe des fichiers accessibles en lecture par tous (rw-r—r--) qui n'ont pas vocation a être servi par Apache : /etc/*, /home/*, /proc/*...
- Parades :
  - chroot / virtualisation
  - utiliser les contrôles d'accès d'Apache
  - restreindre les droits d'accès via l'OS en exécutant Apache sous un uid gid particuliers

Directives : User & Group

- « user » et « group » définissent l'identité sous laquelle tourne le serveur et il répond aux requêtes http
- contexte : server config
- 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 privilège lui permettant d'accéder à des fichiers en dehors du Web Space (/etc ...)*
- *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

- Il faut donc créer un compte et un groupe spécifiques pour Apache (www-data , 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 de Virtual Host

Espace web principal

- « ServerRoot »
  - Racine du serveur = répertoire à partir duquel Apache est installé et configuré "/etc/apache2"
  - contient par défaut des répertoires conf, logs, bin, htdocs... (cela varie selon les distributions)
- « DocumentRoot »
  - définit la racine de l'espace web principal (/var/www)
  - 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 /etc/apache2
DocumentRoot /var/www
http://www.monserveur.fr/toto.html
```

Espace web principal

- DocumentRoot peut apparaître dans un contexte VirtualHost : c'est la racine de l'hôte virtuel (vh)
  - espace web principal = ∪ (DocumentRoot Principal, DocumentRoot des VH)
- ServerRoot et DocumentRoot et VirtualHost ne sont possibles que dans la config principale,
  - donc la définition de l'espace web principal est sous contrôle du sysadmin

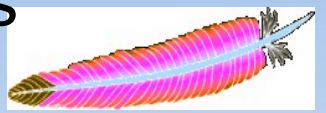
Espace des pages personnelles

- La directive « UserDir » définit la racine des pages personnelles
  - Par défaut : UserDir public_html
- Mapping de http://www.x.fr/~gaston/page.html

Userdir	mapping
public_html	~gaston/public_html/page.html
/var/pages_perso	/var/pages_perso/gaston/page.html
/var/*/pages_perso	/var/gaston/pages_perso/page.html
http://autre.fr/~*/	http://autre.fr/~gaston/page.html

Espace des pages personnelles

- On peut désactiver ou activer des pages personneles (recommandé pour les pages perso de root)



```
UserDir disabled raoul gaston
```

```
UserDir disabled root
```

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

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

- UserDir peut apparaître dans un contexte Virtual Host
- Espace pages perso = ∪ espace pages perso des VH
- UserDir interdit dans .htaccess, donc la définition des espaces pages perso est sous contrôle du webmaster

Directive « Alias », « ScriptAlias »

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

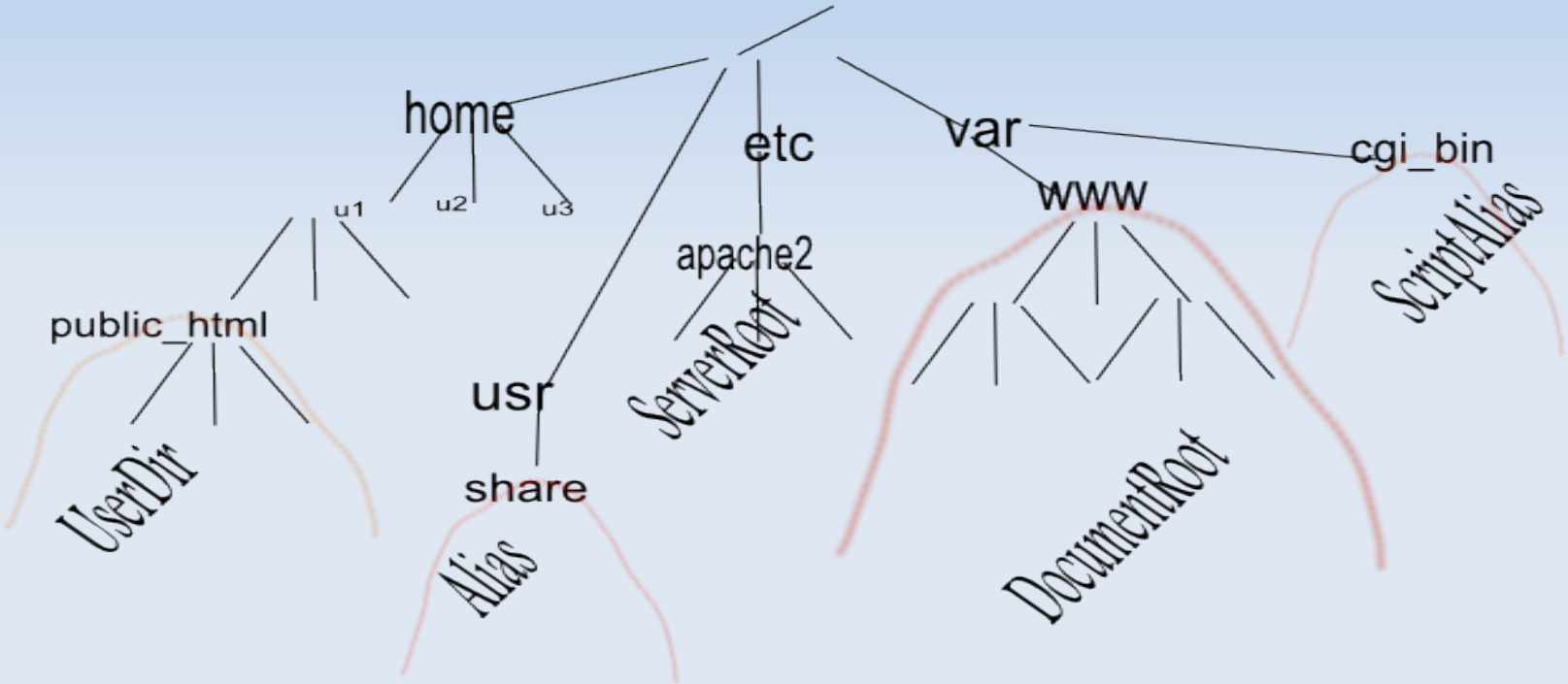
```
Alias /doc /usr/linux/docs
```

  - http://www.exemple.fr/doc/page.html

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

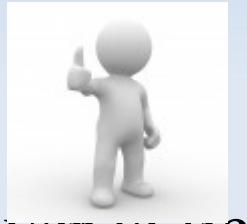
- ScriptAlias fait la même chose, et spécifie en plus que tous les fichiers doivent être traités comme des scripts CGI
  - AliasMatch, ScriptAliasMatch
- « Alias » est interdit dans .htaccess, donc l'extension de l'espace web via des alias est sous contrôle du webmaster

Résumé de l'URL-Space



Espace web total

- espace web total ≡ ∪ DocumentRoot + ∪ UserDir + ∪ Alias + U VirtualHost



- sous contrôle du sysadmin
- Pour avoir une idée de l'espace Web, trouver le fichier de conf principal (httpd.conf, apache2.conf), 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
```

Exemple de recherche de l'URL space

```
$ 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/"
```

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*

Liens symboliques

- Les liens symboliques situés dans l'espace web ouvrent des brèches vers le filesystem (extérieur au Web space)
  - ln -s /etc/passwd ~gaston/
- Hors contrôle du sysadmin

- Comment les permettre, tout en les limitant ?

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

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

Les options de Apache

- **All** : toute les options ci dessous sauf MultiViews (config par défaut) 
- **ExecCGI** : permet l'exécution de scripts CGI (nécessité du module mod_cgi)
- **FollowSymLinks** : Le serveur peut suivre les liens symboliques dans ce répertoire
- **SymLinksIfOwnerMatch**: Le serveur peut suivre les liens symboliques uniquement si le fichier source et le fichier cible ont le même propriétaire!
- **Includes** : permission d'utiliser les « Server-side includes » commandes qu'on peut intégrer dans des pages html (nécessité du module mod_include)
- **IncludesNOEXEC** : permission d'utiliser les « Server-side includes SAUF les #exec cmd and #exec cgi
- **Indexes** : en l'absence d'un fichier « index.html, le serveur affiche la liste des fichiers présents dans un répertoire DirectoryIndex (e.g., index.html)
- **MultiViews** : permet la négociation de contenu, méthode par laquelle le serveur réalise une recherche par motifs implicites, et choisit parmi les résultats 

Héritage des options inter répertoires

```
<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 rajoute ou enleve les options héritées

```
<Directory /var/www/labo>
   Options -Indexes +MultiViews
</Directory>
```

Héritage des options

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

- Les symlinks sont-ils pour autant désactivés ?
  - pas forcément, s'il existe un fichier .htaccess contenant : (rappel les fichiers .htacces sont évalués en dernier)

```
    Options FollowSymlinks
```

- => Comment contrôler la présence des .htaccess dans les répertoires des utilisateurs ?

Maîtriser les .htaccess

- Le nom des fichiers de « surcharge » est-il bien .htaccess ? vérifier :

```
AccessFileName .htaccess
```

  - le vérifier pour tous les virtual host
- 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/permissif>
    AllowOverride All
</Directory>
```

AllowOverride : que peut-on surcharger?

- ***AuthConfig*** permet utilisation des fichiers d'authentification *(AuthDBMGroupFile, AuthDBMUserFile, AuthGroupFile, AuthName, AuthType, AuthUserFile, Require, etc.).*
- ***FileInfo*** permet utilisation de directives controlant les types de fichiers *(DefaultType, ErrorDocument, ForceType, LanguagePriority, SetHandler, SetInputFilter, SetOutputFilter..)*
- ***Indexes*** permet utilisation de directives controlant l'indexation des répertoires *(AddDescription, AddIcon, AddIconByEncoding, AddIconByType, DefaultIcon, DirectoryIndex, FancyIndexing, HeaderName, IndexIgnore, IndexOptions, ReadmeName, etc.).*
- ***Limit*** permet utilisation de directives controlant l'acces des clients (Allow, Deny and Order).
- ***Options*** permet utilisation de la directive « option »
- All |none : toutes ou aucune des options ci dessus

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

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*

Protection de l'espace Web

- Plusieurs types de protection de l'espace web sont permises par Apache :
- Par filtrage des adresses IP sources
- Par authentification:
  - Par login/passwd
  - Par certificat x509

filtrage sur l'adresse IP source

- Allow, deny : pour permettre ou interdire des adresses

```
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 connait 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 authentification

- Pour protéger un répertoire «prive»
  - Nécessité d'un fichier de login+mot de passe spécifique

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

  - Le serveur va demander « require » un utilisateur valide existant dans le fichier dbmpasswd

Directive « Satisfy »

- La directive « satisfy » permet de combiner les 2 types de protection (par adresse et/ou authentification)
- Valeurs : All ou Any
  - satisfy all : require ET allow/deny
  - satisfy any : require OU allow/deny

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

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

dans cet exemple l'authentification n'a pas lieu (allow from all et Any)

- Pour autoriser Satisfy dans .htaccess :

```
AllowOverride AuthConfig
```

Option Indexes

- Lorsque Apache accède à un répertoire du web space
  - si ce répertoire contient un index.html (nom de fichier défini par la directive « 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 affichée
    - sinon, une erreur est retournée

- Pour autoriser Options Indexes dans .htaccess :

```
AllowOverride Options
```

```
AllowOverride Options=Indexes
```

- Autoriser DirectoryIndex dans .htaccess :

```
AllowOverride Indexes
```

Interdire l'accès à des répertoires ou fichiers

```
<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]      ## [F] = Forbidden
```

Configuration initiale

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

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

## ne pas interdire QUE les fichiers .ht* mais aussi
toutes les autres fichiers  qu'on laisse par erreur
<Files ~ "^\.ht|~$|^\.bash|README|\.bak|\.ini|
[\.\-]old|[\.\-]sav|\.inc|\.sh">
    Order allow,deny
    Deny from all
</Files>

<Directory "/usr/local/apache/cgi-bin">
```

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*

Action

- Une action exécutable de Apache peut être :
  - liée statiquement au serveur,
  - ajoutée comme un module chargé par le serveur
    - mod_action, mod_status, mod_server
  - ajoutée avec la directive « Action » pour traiter des fichiers d'un type particulier
    - Action image/gif /cgi-bin/images.cgi
  - Pris en charge par des filtres
    - SetInputFilter
    - SetOutputFilter

Actions prédéfinies

- *default-handler* : envoie le contenu statique du fichier
- *send-as-is* : envoie un fichier contenant ses propres en-têtes 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 »

- Permet de 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

- Handler = définition et déclenchement d'une certaine action

- en fonction :
  - d'un type de contenu
  - d'une extension de fichier
  - d'un emplacement de fichier

Directive AddHandler

- Établit une relation entre une extension de fichier et un script qui va les traiter :
  - Contexte : server config, virtual host, directory, .htaccess
- Exemples :
  - pour que les fichiers .html soient traités par le script « footer.pl »
  - Les fichiers .xyz seront traités par les scripts program.cgi

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

# Files of a particular file extension
AddHandler my-file-type .xyz
Action my-file-type /cgi-bin/program.cgi
```

Directive SetHandler

Créer un handler pour tous les fichiers (inconditionnel) situés dans un emplacement

- Contexte : server config, virtual host, directory, .htaccess

- Exemple : toute les images de ce répertoire « images » seront pris en charge par images.cgi

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

<Directory /htdocs/images>
   SetHandler gif-handler
</Directory>
<Location /server-status>
        SetHandler server-status
        Order allow,deny
        Allow from com.univ-mrs.fr
</Location>
```

- Pour Interdire AddHandler et SetHandler dans .htaccess

  ne pas spécifier : AllowOverride FileInfo

Filtres

- Permet de spécifier des traitements à exécuter avant ou après l'action
  - SetInputFilter : défini le filtre qui va prendre en charge la requête du client avant qu'elle soit envoyée au serveur
  - AddInputFilter
  - SetOutputFilter : défini le filtre qui va prendre en charge la réponse du serveur avant qu'elle ne soit envoyée au client
  - AddOutputFilter
  - AddOutputFilterByType

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*

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

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
```

Server Side Include : 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
```

Server Side Include : 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
```

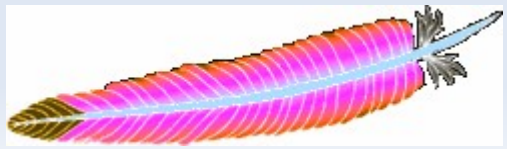
  - les options d'exécution :

```
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 ?

- Pour 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.
  - 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 a 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 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,
```

- 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 : resolution DNS des clients
  - 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: /
```

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***

- 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

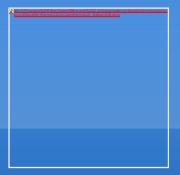
Solution 1

- 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

- Acti
```
$ grep htdocs /etc/fstab

LABEL=htdocs      /htdocs  ext3    defaults,acl   1 2
```
- Acti
```
mount -o remount,acl -L htdocs
```
- P
```
setfacl -m u::rw -m g:apache:r -m g:wmsite:rw -m o:: page.html
```
- F
r
```
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
```

A propos de permission...

- ne pas faire `chmod 777` comme indiqué dans les docs d'installation des CMS et wikis

```
chown apache fichier_ou_rep
```

```
chgrp apache fichier_ou_rep
chmod g+rw fichier
chmod g+rwx rep
```

```
setacl -m u:apache:rwx d:u:apache:rwx rep
```

```
setacl -m g:apache:rwx d:g:apache:rwx rep
```

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/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`