LP R&T RSF IR TD1
APACHE

Arborescence simplifiée d'Apache :

```
usr
 |
 +-- local
      |
      +-- apache
           |
           +-- bin       exécutables
           |
           +-- cgi-bin   scripts cgi
           |
           +-- conf      fichiers de configuration
           |   |
           |   +-- httpd.conf  fichier de configuration d'apache
           |   |
           |   +-- mimes.types type de fichier suivant son
           |   |               extension
           |   +-- magic
           |   |
           |   +-- php.ini
           +-- libexec   modules
           |   |
           |   +-- libphp5.so
           +-- logs
           |   |
           |   +-- access.log liste des accès au serveur
           |   |
           |   +-- error-log  liste des erreurs
           +-- htdocs     contient la page d'accueil d'apache
           |
           +-- include
           |
           +-- icons
```

Pour lancer le serveur apache, on démarre le daemon httpd

/etc/rc.d/init.d/httpd start

Après tout modification du fichier de configuration http.conf, le daemon httpd doit être relancé

/etc/rc.d/init.d/httpd start

Le fichier de configuration d'apache

```
# ----------------------Server Configuration----------------------

# ServerType is either inetd, or standalone.
# vous pouvez soit lancer Apache par le super daemon inetd (config dans /etc/inetd.conf) soit tout seul avec un script dans /etc/rc.d/init.d, à noter qu'avec la première méthode vous pouvez contrôler l'accès avec les TCP_Wrappers (hosts.deny, hosts.allow)
# voir la note en bas du paragraphe pour les avantages de l'un ou de l'autre
# moi j'ai choisi lancement en standalone (voir paragraphe suivant)

ServerType standalone

(...)

# Do NOT add a slash at the end of the directory path.
# Répertoire racine d'Apache
ServerRoot "/usr/local/apache"

(...)

# PidFile: The file in which the server should record its process
# identification number when it starts.
# C'est dans ce fichier qu'on trouvera l'identification du process d'Apache
# c'est utile pour ne pas lancer deux fois Apache par exemple
PidFile /usr/local/apache/logs/httpd.pid

(...)

# Limit on total number of servers running, i.e., limit on the
```

```
number
# of clients who can simultaneously connect --- if this limit is ever
# reached, clients will be LOCKED OUT, so it should NOT BE SET TOO LOW.
# It is intended mainly as a brake to keep a runaway server from taking
# Unix with it as it spirals down...
```

 # ici on fixe le nombre de clients max qui peuvent se connecter à un moment donné

```
MaxClients 150
```

(...) suit ensuite une liste de modules, ceux avec un # devant ne seront pas inclus

```
LoadModule vhost_alias_module libexec/mod_vhost_alias.so
LoadModule env_module         libexec/mod_env.so
LoadModule config_log_module  libexec/mod_log_config.so
LoadModule mime_magic_module  libexec/mod_mime_magic.so
LoadModule mime_module        libexec/mod_mime.so
LoadModule negotiation_module libexec/mod_negotiation.so
LoadModule status_module      libexec/mod_status.so
LoadModule info_module        libexec/mod_info.so
```

```
# Port: The port the standalone listens to. For ports < 1023, you will
# need httpd to be run as root initially.
```

 # numéro de port pour le serveur, traditionnellement à 80

```
Port 80
```

```
(...)
```

 # On lance initialement httpd en tant que root, puis immédiatement

 # c'est l'utilisateur nobody (groupe nobody) qui en devient le proprio

 # ainsi s'il y a une faille dans Apache, le hacker au lieu de devenir root

 # devient nobody avec les droits qui vont avec

 # pour vérifier que nobody est bien le proprio

 # ps aux | grep httpd

User nobody

Group nogroup

ServerAdmin: Your address, where problems with the server should be

e-mailed. This address appears on some server-generated pages, such

as error documents.

 # En cas de problème un email sera envoyé au webmaster, mettez donc

 # ici l'adresse email du webmaster

ServerAdmin olivier@asterix.kervao.fr

DocumentRoot: The directory out of which you will serve your

documents. By default, all requests are taken from this directory, but

symbolic links and aliases may be used to point to other locations.

 # C'est dans ce répertoire qu'on va trouver la page d'accueil d'Apache

DocumentRoot "/usr/local/apache/htdocs"

(...)

UserDir: The name of the directory which is appended onto a user's home

directory if a ~user request is received.

 # Chaque utilisateur pourra mettre ses pages dans sa homedirectory

 # dans un répertoire public_html, les pages seront accessibles à l'URL

http://localhost/~user

<IfModule mod_userdir.c>

    UserDir public_html

</IfModule>

(...)

ErrorLog: The location of the error log file.

```
# If you do not specify an ErrorLog directive within a
<VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a
<VirtualHost>
# container, that host's errors will be logged there and not here.
# L'emplacement du fichier de log peut ne pas vous convenir
# si vous voulez le placer dans /var/log/httpd il faut modifier
ici
ErrorLog /usr/local/apache/logs/error_log

(...)

# The location and format of the access logfile (Common Logfile
Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here.  Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
# De même pour l'emplacement du fichier de log des accès à apache
# c'est cette variable qu'il faut modifier
CustomLog /usr/local/apache/logs/access_log common

(...)

# DirectoryIndex: Name of the file or files to use as a pre-
written HTML
# directory index.  Separate multiple entries with spaces.
# liste des fichiers d'entrée des pages
<IfModule mod_dir.c>
    DirectoryIndex index.html index.htm index.php3 index.php
index.php4
</IfModule>

(...)

# priorité dans les langages à utiliser, mettez fr en premier
    <IfModule mod_negotiation.c>
        LanguagePriority fr en da nl et de el it ja pl pt pt-br
ltz ca es sv
    </IfModule>
```

```
(...)

# Apache peut se comporter comme un proxy comme squid
# par défaut cette fonction n'est pas activée
# Proxy Server directives. Uncomment the following line to
# enable the proxy server:

# ProxyRequests On

# To enable the cache as well, edit and uncomment the following
lines:

# CacheRoot /var/cache/httpd
# CacheSize 5
# CacheGcInterval 4
# CacheMaxExpire 24
# CacheLastModifiedFactor 0.1
# CacheDefaultExpire 1
# NoCache a_domain.com another_domain.edu joes.garage_sale.com
```

Les pages Web des utilisateurs

Le problème avec le répertoire public_html des utilisateurs est qu'il faut mettre 755 au niveau de la home directory, ce qui est particulièrement gênant au niveau sécurité. Vous pouvez spécifier que chaque utilisateur doit créer ses pages sous /home/http/login-utilisateur

Ainsi pour l'utilisateur toto quand vous taperez comme URL

http://serveur-apache/~toto,

apache ira chercher le fichier index.htm sous /home/httpd/toto.

Q1 :

Donner la modification à apporter au fichier httpd.conf

```
UserDir /home/httpd
```

On peut aller plus loin en spécifiant un répertoire particulier, /home/httpd/toto/html par exemple.

Q2 :

Donner la modification à apporter au fichier httpd.conf

UserDir /home/httpd/*/html

Les alias

Si vous ne voulez pas mettre en place un serveur DNS, vous avez un moyen plus simple, les alias. Concrètement, votre serveur s'appelle obelix , vous voulez rendre accessible les fichiers html se trouvant sous /usr/doc/html , les utilisateurs devront taper dans leur navigateur préféré:

http://obelix/doc.

Q3 :

Quelles sont les modifications à apporter au fichier httpd.conf ?

```
Alias /icons/ "/usr/local/apache/icons/"
Alias /doc "/usr/doc/html/"
```

NOTE Si vous mettre /doc/ à la place de /doc dans l'URL il faudra taper http://obelix/doc/, si vous omettez le dernier /, vous aurez une erreur.

Protection d'une page

La protection d'une page pour l'utilisateur olivier se fait de manière très simple, tous les fichiers à accès limité doivent être concentrés dans un même répertoire

/home/olivier/public_html/reserve

par exemple.

Quand un utilisateur donne comme URL

http://obelix/~olivier/reserve,

une fenêtre popup va s'ouvrir en lui demandant de rentrer son nom d'utilisateur et son mot de passe.

Q4 :

Comment limiter l'accès des pages du répertoire réservé aux utilisateurs olivier et veronique

Il suffit de créer dans

/home/olivier/public_html/reserve

un fichier qu'on devra nommer .htaccess contenant:

```
AuthUserFile auth/olivier.users
```

AuthGroupFile auth/olivier.group

AuthName "Acces Restreint"

AuthType Basic

require group autorise

Le fichier

olivier.users

va contenir une liste d'utilisateurs, il va se trouver sous le répertoire

/usr/local/apache/auth(éventuellement à créer),

pour info vous pouvez changer le chemin /usr/local/apache en modifiant la valeur de la variable ServerRoot qu'on trouve dans le fichier httpd.conf.

Le fichier olivier.group correspond à une liste des groupes de personnes, ces mêmes personnes ayant été défini dans le fichier olivier.users.

Le principe consiste à créer un groupe de personnes autorisées et à leur attribuer un mot de passe à chacune, seul ce groupe pourra accéder à la section réservée.

Pour créer ces fichiers il suffit, en tant que root, d'une part de créer le répertoire

/usr/local/apache/auth,

puis de taper:

htpasswd -c /usr/local/apache/auth/olivier.users olivier

L'option -c correspond à la création du fichier olivier.users.

On va alors avoir à rentrer un mot de passe pour l'utilisateur olivier.

New password:

On confirme

Re-type new password:

Adding password for user olivier

Pour créer un autre utilisateur veronique vous taperez la même

commande sans l'option de création :

htpasswd /etc/httpd/auth/olivier.users veronique

Toujours en tant que root, créer le fichier

/usr/local/apache/auth/olivier.group

qui contiendra pour autoriser par exemple olivier et veronique à accéder aux pages réservées :

autorise: olivier veronique

Il faut décommenter maintenant les lignes suivantes dans le fichier httpd.conf

```
<Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    <Limit GET POST OPTIONS PROPFIND>
        Order allow,deny
        Allow from all
    </Limit>
    <LimitExcept GET POST OPTIONS PROPFIND>
        Order deny,allow
        Deny from all
    </LimitExcept>
</Directory>
```

Ces lignes autorisent l'emploi du fichier .htaccess dans les pages de vos utilisateurs (répertoire /home/*/public_html)

Notez que pour que quelqu'un ne puisse pas jeter un coup d'oeil dans les fichiers .htaccess de vos utilisateurs, le fichier httpd.conf, contient la directive suivante:

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

Les hôtes virtuels

On peut mettre en place des hôtes virtuels, en d'autres termes un utilisateur pour un même serveur Apache croira en voir plusieurs.

Exemple, soit votre serveur Apache obelix (adresse IP 192.168.13.11),

votre domaine breizland.bz, on va créer les hôtes virtuels

www.asterix.breizland.bz

et www.idefix.breizland.bz

qui vont pointer chacun vers un endroit différent du disque (respectivement

/usr/local/asterix

et /usr/local/idefix

chacun contenant des pages html).

Q5 :

Comment mettre en place les hôtes virtuels ?

Dans le fichier httpd.conf le module est déjà chargé :

```
LoadModule vhost_alias_module libexec/mod_vhost_alias.so
AddModule mod_vhost_alias.c
```

On va rajouter tout à la fin du fichier:

```
NameVirtualHost 192.168.13.11

<VirtualHost 192.168.13.11>
ServerName obelix.breizland.bz
DocumentRoot /usr/local/apache/htdocs
ErrorLog logs/obelix-error_log
TransferLog logs/obelix-access_log
</VirtualHost>

<VirtualHost 192.168.13.11>
ServerName www.asterix.breizland.bz
DocumentRoot /usr/local/asterix
ErrorLog logs/asterix-error_log
TransferLog logs/asterix-access_log
</VirtualHost>

<VirtualHost 192.168.13.11>
ServerName www.idefix.breizland.bz
DocumentRoot /usr/local/idefix
```

ErrorLog logs/idefix-error_log

TransferLog logs/idefix-access_log

</VirtualHost>

Relancez Apache en tapant:

/etc/rc.d/init.d/httpd restart

Maintenant nous allons créer nos hôtes asterix et idefix, pour cela vous avez deux méthodes:

- rajouter www.asterix.breizland.bz et www.idefix.breizland.bz dans /etc/hosts

sur la même ligne que votre serveur Apache (obelix dans notre exemple).

192.168.13.11 obelix obelix.breizland.bz www.asterix.breizland.bz www.idefix.breizland.bz

Normalement si vous faites un ping sur www.idefix.breizland.bz ça

devrait marcher, pour les postes clients il faudra rajouter la même ligne dans le fichier hosts (non nécessaire).

- si vous disposez d'un serveur DNS sur votre machine, au niveau de votre config DNS dans votre fichier breizland.bz qui se trouve sous /var/named vous devez rajouter tout à la fin:

www.asterix A 192.168.13.11

www.idefix A 192.168.13.11

Relancez le DNS en tapant:

/etc/rc.d/init.d/named restart

Pour tester tapez dans un shell:

ping www.asterix.breizland.bz

Maintenant dans le champ URL de votre navigateur préféré:

http://www.asterix.breizland.bz

Et là, normalement vous devriez voir s'afficher la page que vous avez placé sous /usr/local/asterix