

TP - APACHE 2

Exercice 1: Un serveur simple

- a) Clonez le .vdi de la machine Fedora contenant une version supérieure à 2.4.23 du serveur Apache

```
$ VBoxManage clonehd nom_image.vdi apache.vdi<cr>
```

- b) Créez une MV 'apache' liée au disque cloné

Voir avant

- c) Préfixez tous les fichiers d'extension .conf du dossier /etc/httpd/conf.d par '01-' et créez un fichier 00-1main.conf et 00-0server.conf

```
# cd /etc/httpd/conf.d<cr>
# for i in `ls *.conf` ; do mv $i 01-$i ; done<cr>
# touch 00-0server.conf 00-1main.conf<cr>
```

- d) Configurez le serveur pour:

- . qu'il écoute sur le port 80 sur toutes les interfaces
- . qu'il présente une page d'accueil index.html lors d'une requête vers l'URL de ce site (inventez son contenu)
- . qu'il affiche une page html personnalisée en cas d'erreur 404

/etc/httpd/conf/httpd.conf

```
...
Listen 80                                → inchangé
ServerRoot    "/etc/httpd"              → inchangé
DocumentRoot  "/var/www/html"           → inchangé
ServerAdmin   root@localhost             → inchangé
...
```

/etc/httpd/conf.d/00-server.conf

```
ServerName     www.mysite.be
ServerAdmin    moi@mysite.be             → redéfini
ErrorDocument  404 /erreur/erreur_404.html
```

/var/www/html/index.html

```
<html>
<head>
<title> Test du site web de base </title>
</head>
<body>
<h1>Site web de base. Ca marche !!! </h1>
</body>
</html>
```

/var/www/html/erreur/erreur_404.html

```
<html>
<head>
```

```
<title> Test de la page d'erreur 404</title>
</head>
<body>
<h1>La page demandée n'existe pas. </h1>
</body>
</html>
```

/etc/hosts

```
127.0.0.1 localhost ... www.mysite.be
```

Pour que www.mysite.be corresponde à 127.0.0.1 sur la machine locale (pas de DNS ici!)

/etc/nsswitch.conf

```
...
```

```
hosts:      files dns
```

e) Vérifiez la configuration du serveur

```
# apachectl configtest<cr>
Syntax OK
```

f) Testez votre serveur pour vérifier si la page d'accueil est bien offerte:

. à l'aide de l'utilitaire telnet

```
# systemctl restart httpd.service<cr>
Lancement du serveur web.
```

```
# ps ax<cr>
On doit voir une série d'instances de httpd.
```

```
# netstat -tnl<cr>
On voit le serveur qui écoute sur le port 80.
```

```
# ping localhost<cr>
# ping 127.0.0.1<cr>
# ping www.mysite.be<cr>
Doivent toutes répondre sur l'IP 127.0.0.1.
# telnet www.mysite.be 80<cr>
Trying 127.0.0.1...
```

```
...
```

```
GET / HTTP/1.1<cr>
host: www.mysite.be<cr>
```

```
<cr>
```

```
HTTP/1.1 200 OK
Date: SAT, 29 Sep ...
Server: Apache/2.4.25 (Fedora)
```

```
...
```

```
Content-Type: text/html; ...
```

```
...
```

```
<html>
```

```
...
```

. à l'aide d'un navigateur quelconque

Dans la barre d'URL, tapez: `http://www.mysite.be`
On doit voir la page d'accueil du site.

g) Reconfigurez Apache pour qu'il écoute cette fois sur le port 8080:

/etc/httpd/conf/httpd.conf

...
Listen 8080 → *modification*

h) Testez votre serveur:

. à l'aide de l'utilitaire telnet

`systemctl restart httpd.service`<cr>

Relancement du serveur web.

`ps ax`<cr>

On doit voir une série d'instances de httpd.

`netstat -tnl`<cr>

On voit le serveur qui écoute sur le port 8080.

`telnet www.mysite.be 8080`<cr>

Trying 127.0.0.1...

...

GET /coucou.html HTTP/1.1<cr>

host: www.mysite.be<cr>

<cr>

HTTP/1.1 404 Not Found

Date: SAT, 29 Sep ...

Server: Apache/2.4.25 (Fedora)

...

Content-Type: text/html; ...

...

<Code html de la page d'erreur personnalisée>

...

. à l'aide d'un navigateur quelconque

Dans la barre d'URL, tapez: `http://www.mysite.be:8080/coucou.html`

On doit voir la page d'erreur personnalisée

i) Compilez, installez et testez un module tiers (mod_pony)

Google: Recherchez le mot clé 'mod_pony'...

Lien → http://svn.rcbowen.com/svn/public/mod_pony/mod_pony.c

```
# yum install gcc httpd-devel libtool -y<cr>
```

Rem: httpd-devel contient le binaire apxs

*(# yum whatprovides */apxs + # yum info httpd-devel)*

```
# wget http://svn.rcbowen.com/svn/public/mod_pony/mod_pony.c<cr>
```

```
# systemctl stop httpd.service<cr>
```

```
# mkdir /usr/lib/rpm/redhat<cr>
```

```
# touch /usr/lib/rpm/redhat/redhat-hardened-cc1<cr>
```

```
# touch /usr/lib/rpm/redhat/redhat-hardened-ld<cr>
```

} *Pas nécessaire sous CentOS...*

```
# /usr/bin/apxs -cia mod_pony.c
```

Rem:

- *mod_pony.so est copié dans /usr/lib64/httpd/modules*
- *le module est activé dans /etc/httpd/conf/httpd.conf*
Loadmodule ... /mod_pony.so

- *commenter cette ligne dans l'httpd.conf et la rajouter dans /etc/httpd/conf.modules.d/00_pony.conf*
Loadmodule ... /mod_pony.so

A créer /etc/httpd/conf.d/01-main.conf :

```
<Location /pony>
```

```
    SetHandler pony    → On souhaite que le serveur active le module  
                        pony lorsqu'il reçoit l'URL  
</Location>           http://127.0.0.1/pony
```

```
# systemctl restart httpd.service<cr>
```

```
# lynx http://127.0.0.1:8080/pony<cr>
```

--- Un poney ou 'not yours' s'affichera ---

Exercice 2

- a) Reprendre l'exercice 2 et reconfigurez le serveur pour qu'il écoute sur le port 80 sur votre interface.

/etc/httpd/conf/httpd.conf

...
Listen 80 → *modification*

- b) Quel est le Module Multi-Processus (MPM) utilisé actuellement ?

```
# httpd -L | grep -i startservers<cr> ou # httpd -V | grep MPM
StartServers (prefork.c)           Server MPM : prefork
→ c'est prefork
```

- c) Combien de processus enfants y a-t-il actuellement ?

```
# ps -ef | grep httpd<cr>
```

root	646	1	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND
apache	729	646	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND
apache	730	646	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND
apache	731	646	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND
apache	735	646	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND
apache	736	646	0	09:08	?	00:00:00	/usr/sbin/httpd -DFOREGROUND

→ 5 processus enfants. Le 1^{er} de la liste est le dispatcher

- d) Configurez-le pour:

- qu'il lance 7 processus enfant en mode 'prefork' au démarrage et qu'il s'assure qu'il en reste toujours au moins 3 en réserve. En cas de montée en charge, 20 processus enfants maximum seront créés pour satisfaire les requêtes.

Lorsque le serveur aura satisfait toutes les requêtes de cette montée, il se stabilisera avec 10 processus enfants.

Testez tout cela, à l'aide d'outils adéquats.

/etc/httpd/conf/conf.modules.d/00-mpm.conf

...
Loadmodule ... /prefork.so → A vérifier

/etc/httpd/conf.d/00-server.conf

```
<IfModule prefork.c>
    MinSpareServers 3
    MaxSpareServers 10
    StartServers 7
    MaxRequestWorkers 20
    ServerLimit 20
</IfModule>
```

```
# systemctl restart httpd.service<cr>
# ps -ef | grep httpd<cr>
```

...
→ 7 processus enfants ici.

Tests

Sur un terminal 2

```
# watch -n 0 'ps ax | grep httpd' <cr>
```

Sur un terminal 3

```
# watch -n 0 'free' <cr>
```

Sur le terminal 1

On va utiliser l'outil d'Apache 'ab' qui vous permet de tester les performances de votre serveur HTTP Apache (Plus d'info: voir documentation d'Apache)

```
# ab -n 10000 -c 500 http://www.mysite.be/index.html/ <cr>
```

→ sur le terminal 2, on constate que le nombre de processus enfants augmente jusqu'à 20 durant le test^(*); puis diminue au fur et à mesure que les requêtes sont traitées jusqu'à se stabiliser à 10 processus enfants.

→ sur le terminal 3, on constate une consommation mémoire de plus en plus importante au fur et à mesure que les instances de httpd sont créées (environ 4 Mo/instance et 20 Mo avec php)...

(*) La directive **MaxRequestWorkers** permet de fixer le nombre maximum de requêtes pouvant être traitées simultanément. Si la limite **MaxRequestWorkers** est atteinte, toute tentative de connexion sera normalement mise dans une file d'attente. Lorsqu'un processus enfant se libèrera suite à la fin du traitement d'une requête, la connexion en attente pourra être traitée à son tour.

Pour les serveurs non threadés (c'est à dire utilisant prefork), la directive **MaxRequestWorkers** définit alors le nombre maximum de processus enfants qui pourront être lancés simultanément pour traiter les requêtes. La valeur par défaut est 256 ; si vous l'augmentez, vous devez aussi augmenter la valeur de la directive **ServerLimit** qui définit le nombre maximum que l'on peut affecter à la directive **MaxRequestWorkers**, et ceci pour la durée de vie du processus Apache httpd..

Pour les serveur threadés et hybrides (utilisant par exemple event ou worker), **MaxRequestWorkers** définit alors le nombre total de threads qui seront disponibles pour servir les clients. La valeur par défaut est 16 x 25 (**ServerLimit** x **ThreadsPerChild**). Par conséquent, pour affecter à la directive **MaxRequestWorkers** une valeur qui requiert plus de 16 processus, vous devez aussi augmenter la valeur de la directive **ServerLimit**.

- **que la requête `http://www.mysite.be/pamauthor` présente la page `/usr/share/doc/pam/html/sag-author.html`**

```
/etc/httpd/conf/conf.modules.d/00-base.conf
```

```
...  
LoadModule alias_module modules/mod_alias.so  
...
```

```
/etc/httpd/conf/conf.d/00-1main.conf
```

```
...  
Alias /pamauthor  
    "/usr/share/doc/pam/html/sag-author.html"
```

```
<Directory "/usr/share/doc/pam/html">  
    Require all granted  
</Directory>
```

- **qu'il donne la possibilité aux utilisateurs *jean*, *louis* et *bernard* de disposer d'un site perso dont la page d'accueil puisse être `index.html` ou `accueil.html` (inventez leur contenu). Le site perso de *bernard* ne sera pas accessible pour le moment.**

```
# useradd jean ...          + # chmod 755 /home/jean  
# useradd louis ...         + # chmod 755 /home/louis  
# useradd bernard ...       + # chmod 755 /home/bernard
```

```
jean~$ mkdir public_html  
---jean crée son fichier index.html ---  
jean~/public_html$ cat index.html  
<html>  
<head>  
<title> Test du site perso de jean </title>  
</head>  
<body>  
<h1>Site web de jean. Ca marche !!! </h1>  
</body>  
</html>
```

```
louis~$ mkdir public_html  
---louis crée son fichier accueil.html ---  
louis~/public_html $ cat accueil.html  
<html>  
<head>  
<title> Test du site perso de louis </title>  
</head>  
<body>  
<h1>Site web de louis. Ca marche !!! </h1>  
</body>  
</html>
```

```
bernard~$ mkdir public_html
---bernard crée son fichier index.html ---
bernard~/public_html $ cat index.html
<html>
<head>
<title> Test du site perso de bernard </title>
</head>
<body>
<h1>Site web de bernard. Ca marche !!! </h1>
</body>
</html>
```

/etc/httpd/conf.modules.d/00-base.conf

```
...
LoadModule userdir_module modules/mod_userdir.so
LoadModule dir_module modules/mod_dir.so
...
```

/etc/httpd/conf.d/01-userdir.conf

```
...
<IfModule mod_userdir.c>
    UserDir disabled bernard
    UserDir public_html
</IfModule>
```

/etc/httpd/conf.d/00-server.conf

```
<IfModule dir_module>    → redéfinition du conteneur
    DirectoryIndex index.html accueil.html
</IfModule>
```

- qu'un index des entrées du dossier 'cours' soit présenté lors de la requête <http://www.mysite.be/cours>

```
# mkdir /var/www/html/cours
# cd /var/www/html/cours
# cat > f1.txt ...
# cat > f2.3.3.tar.gz ...
# cat > f2.3.2.tar.gz ...
# cat > f3.jpg ...
```

/etc/httpd/conf.modules.d/00-base.conf

```
...
LoadModule dir_module modules/mod_dir.so
```

/etc/httpd/conf.d/01-autoindex.conf

```
...
IndexOptions FancyIndexing VersionSort ...
```


e) Testez vos configurations.

```
# systemctl restart httpd.service<cr>
```

Via un navigateur:

`http://www.mysite.be/pamauthor`

→ *Redirection vers /usr/share/doc/pam/html/sag-author.html.*

`http://www.mysite.be/~jean`

→ *Page d'accueil de jean.*

`http://www.mysite.be/~louis`

→ *Page d'accueil de louis.*

`http://www.mysite.be/~bernard`

→ *Message d'erreur 404.*

`http://www.mysite.be/cours`

→ *Un index amélioré des entrées du dossier /var/www/html/cours doit apparaître.*

Exercice 3

Configurez Apache pour:

- rendre l'entrée d'un intranet (<http://www.mysite.be/intranet>) uniquement accessible par les machines de notre réseau local 10.103.0.0/16.

```
# mkdir /var/www/html/intranet

/var/www/html/intranet/index.html
<html>
<head>
<title> Test de l'intranet </title>
</head>
<body>
<h1>Site de l'intranet. Ca marche !!! </h1>
</body>
</html>

/etc/httpd/conf.d/00-1main.conf

<Directory /var/www/html/intranet>
  <RequireAny>                                → optionnel
    Require ip 10.103.0.0/16
  </RequireAny>                                → optionnel
</Directory>
```

- que les utilisateurs 'tux', 'bill', 'louis' et 'jean' puissent s'authentifier lorsqu'ils accèdent au site <http://www.mysite.be/beta> ; en sachant que seuls 'tux' et 'bill' auront le droit d'accéder à ses ressources.

```
# mkdir /var/www/html/beta

/var/www/html/beta/index.html

<html>
<head>
<title> Test du site beta </title>
</head>
<body>
<h1>Site du site beta. Ca marche !!! </h1>
</body>
</html>

# mkdir /var/www/securite
# htpasswd -c /var/www/securite/pwd tux
# htpasswd /var/www/securite/pwd bill
# htpasswd /var/www/securite/pwd jean
# htpasswd /var/www/securite/pwd louis
```

/etc/httpd/conf.d/00-1main.conf

```
<Directory /var/www/html/beta>
  AuthName "Acces au site beta"
  AuthType Basic
  AuthUserFile /var/www/securite/pwd
  Require user tux bill
</Directory>
```

Testez vos configurations:

service httpd restart

Via un navigateur:

http://www.mysite.be/intranet

→ Ne marche que si la requête est envoyée à partir d'une machine du réseau 10.103

http://www.mysite.be/beta

→ Accès pour 'bill' et 'tux' mais pas pour 'jean' et 'louis'...

Expliquez le principe de l'authentification Basic.

https://fr.wikipedia.org/wiki/Authentification_HTTP

Client

Serveur

Requête http sans champ 'Authorization'

Demande d'authentification par la méthode "Basic"

www-authenticate : Basic realm="Acces au site beta"

Envoi des identifiants par la méthode Base64

Authorization : Basic dHV4OnR1eA==

Codage de tux:tux en Base64

Expliquez le codage Base64.

<https://fr.wikipedia.org/wiki/Base64>

Soit codé tux:tux en Base64 (chaque groupe de 3 bytes (3*8bits) est codé en (4*6bits))

7475783A747578 >> 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 1 1 1 1 0 0 0 ...

Chaque groupe de 6 bits
sert d'index dans une table.

d H V 4

Testez avec tshark que la méthode d'authentification 'Basic' n'est pas sécurisée.

Terminal 1

```
# tshark -V -i lo port http > http.sniff
```

Terminal 2

```
# lynx http://www.mysite.be/beta
```

Terminal 1

```
# ^C
```

http.sniff

```
GET /beta ...
```

```
...
```

```
WWW-Authenticate : Basic realm="Acces au site beta"
```

```
...
```

```
Authorization : Basic dHV4O...
```

```
Credentials : tux:tux
```

Testez d'autres outils permettant de (dé)coder un message chiffré par la méthode base64.

La commande base64

```
# echo -n tux:tux | base64  
dHV4O nR1eA==
```

```
# echo dHV4O nR1eA== | base64 -d  
tux:tux
```

Le décodeur en ligne: www.base64decode.org

Exercice 4

Configurez Apache pour qu'il puisse en même temps gérer de l'hébergement par ip et par nom.

- 192.168.27.10 **permettra d'héberger l'unique site** jean.mysite.be
- 192.168.27.11 **permettra d'héberger l'unique site** louis.mysite.be
- 192.168.27.2 **permettra d'héberger les sites** vh2.mysite.be **et** vh3.mysite.be. **Ce dernier ne sera accessible qu'aux users repris dans /var/www/securite/pwd.**
- 192.168.27.1 **permettra d'héberger le site** vh1.mysite.be
- **Le site maître sera toujours accessible par l'Ip 192.168.27.1 ou par www.mysite.be. Ce sera aussi le site par défaut si une requête arrive via une interface associée à aucun site.**

/etc/hosts

```
192.168.27.1    ... www.mysite.be vh1.mysite.be
192.168.27.2    ... vh2.mysite.be vh3.mysite.be
192.168.27.10   ... jean.mysite.be
192.168.27.11   ... louis.mysite.be
192.168.27.99   ... notexist.mysite.be
```

→ Pour pouvoir tester en local sans DNS

/etc/sysconfig/network-scripts/ifcfg-enp0s3:0

```
...
IPADDR=192.168.27.1
NETMASK=255.255.255.0
ONBOOT=yes
NAME=enp0s3:0
DEVICE=enp0s3:0
```

→ Faire pareil pour enp0s3:1 enp0s3:2 enp0s3:3 enp0s3:4 en adaptant les variables IPADDR, NAME et DEVICE correctement

```
# mkdir /var/www/vh/jean
# mkdir /var/www/vh/louis
# mkdir /var/www/vh/vh1
# mkdir /var/www/vh/vh2
# mkdir /var/www/vh/vh3
```

→ Pour des raisons de sécurité, les hôtes virtuels sont hébergés dans un dossier différent de celui du site principal.

En effet, si les 'DocumentRoot' les hôtes virtuels étaient hébergés dans /var/www/html, on pourrait recevoir la page d'accueil d'un de ceux-ci via l'URL `http://www.mysite.be/<dossier_de_l'hôte_virtuel>` !!! o:(((

/var/www/vh/jean/index.html

```
<html>
<head>
<title> Test du virtualhost de jean (par IP) </title>
</head>
<body>
<h1>Site du virtualhost de jean (par IP). Ca marche !!! </h1>
</body>
</html>
```

/var/www/vh/louis/index.html

```
<html>
<head>
<title> Test du virtualhost de louis (par IP) </title>
</head>
<body>
<h1>Site du virtualhost de louis (par IP). Ca marche !!! </h1>
</body>
</html>
```

/var/www/vh/vh1/index.html

```
<html>
<head>
<title> Test du virtualhost de vh1 (par nom) </title>
</head>
<body>
<h1>Site du virtualhost de vh1 (par nom). Ca marche !!! </h1>
</body>
</html>
```

/var/www/vh/vh2/index.html

```
<html>
<head>
<title> Test du virtualhost de vh2 (par nom) </title>
</head>
<body>
<h1>Site du virtualhost de vh2 (par nom). Ca marche !!! </h1>
</body>
</html>
```

/var/www/vh/vh3/index.html

```
<html>
<head>
<title> Test du virtualhost de vh3 (par nom) </title>
</head>
<body>
<h1>Site du virtualhost de vh3 (par nom). Ca marche !!! </h1>
</body>
</html>
```

/etc/httpd/conf.d/00-server.conf

...

Par Ip

```
<VirtualHost 192.168.27.10>
    ServerName jean.mysite.be
    DocumentRoot /var/www/vh/jean
</VirtualHost>

<VirtualHost 192.168.27.11>
    ServerName louis.mysite.be
    DocumentRoot /var/www/vh/louis
</VirtualHost>
```

Par Noms

```
<VirtualHost 192.168.27.1>
    ServerName www.mysite.be
    DocumentRoot /var/www/html
</VirtualHost>
```

*Pour pouvoir toujours accéder au site maître via http://192.168.27.1
En 1ère position sinon on touche vh1 (voir théorie)*

*Si ce conteneur est absent, alors vh1 sera automatiquement hébergé
par Ip et http://192.168.27.1 ou http://www.mysite.be touchera vh1.*

```
<VirtualHost 192.168.27.1>
    ServerName vh1.mysite.be
    DocumentRoot /var/www/vh/vh1
</VirtualHost>
```

```
<VirtualHost 192.168.27.2>
    ServerName vh2.mysite.be
    DocumentRoot /var/www/vh/vh2
    <Directory /var/www/vh/vh2>
        AuthName "Acces au site vh2"
        AuthType Basic
        AuthUserFile /var/www/securite/pwd
        Require valid-user
    </Directory>
</VirtualHost>
```

*Peut être placée avec les autres
conteneurs <Directory ...>*

```
<VirtualHost 192.168.27.2>
    ServerName vh3.mysite.be
    DocumentRoot /var/www/vh/vh3
</VirtualHost>
```

Testez vos configurations:

```
# systemctl stop httpd.service
```

```
# httpd -S | more
```

...

→ *Info. intéressantes concernant la configuration des virtualhosts.*

...

```
# systemctl start httpd.service
```

Via un navigateur:

`http://louis.mysite.be` *ou* `http://192.168.27.11`

→ *On accède au virtualhost louis*

`http://jean.mysite.be` *ou* `http://192.168.27.10`

→ *On accède au virtualhost jean*

`http://vh1.mysite.be`

→ *On accède au virtualhost vh1*

`http://vh2.mysite.be` *ou* `http://192.168.27.2`

→ *On accède au virtualhost vh2 à condition de s'être authentifié sous tux, bill, louis ou jean*

`http://vh3.mysite.be`

→ *On accède au virtualhost vh3*

`http://www.mysite.be` *ou* `http://192.168.27.1`

→ *On accède au site maître*

`http://notexist.mysite.be`

→ *On accède au site maître*

`http://www.mysite.be/intranet`

→ *Ne marche que si la requête est envoyée à partir d'une machine du réseau 10.103*

`http://www.mysite.be/beta`

→ *Accès pour 'bill' et 'tux' mais pas pour 'jean' et 'louis'...*

Exercice 5

Continuez la configuration d'Apache pour qu'il réponde aux exigences suivantes:

	Site maître	Sites virtuels
Journal des erreurs		
Localisation	logs/error_log	log/error_log
Niveau de criticité	debug	warn
Journal des accès		
Localisation	logs/access_log	logs/access_log.prefixe
Format	combined	common

Qu'est-ce que le format combiné ? → Voir l'aide sur le site d'Apache

Ce format est identique au Common Log Format, avec deux champs supplémentaires. Chacun de ces deux champs utilise la directive commençant par le caractère "%" `%{header}i`, où *header* peut être n'importe quel en-tête de requête HTTP.

/etc/httpd/conf.d/00-server.conf

Pour tout sauf les sites virtuels

ErrorLog "logs/error_log"

LogLevel debug

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\"
\"%{User-agent}i\"" combined

CustomLog log/access_log combined

Pour les sites virtuels

Par Ip

<VirtualHost 192.168.27.10>

ServerName jean.mysite.be

DocumentRoot /var/www/vh/jean

ErrorLog "logs/error_log"

LogLevel warn

LogFormat "%h %l %u %t \"%r\" %>s %b" common

CustomLog log/access_log.jean common

</VirtualHost>

<VirtualHost 192.168.27.11>

ServerName louis.mysite.be

DocumentRoot /var/www/vh/louis

ErrorLog "logs/error_log"

LogLevel warn

LogFormat "%h %l %u %t \"%r\" %>s %b" common

CustomLog log/access_log.louis common

</VirtualHost>

Par Noms

```
<VirtualHost 192.168.27.1>
  ServerName www.mysite.be
  DocumentRoot /var/www/html

  ErrorLog "logs/error_log"
  LogLevel debug
  LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\"
                                                    \"%{User-agent}i\"" combined
  CustomLog log/access_log combined
</VirtualHost>
```

```
<VirtualHost 192.168.27.1>
  ServerName vh1.mysite.be
  DocumentRoot /var/www/vh/vh1

  ErrorLog "logs/error_log"
  LogLevel warn
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog log/access_log.vh1 common
</VirtualHost>
```

```
<VirtualHost 192.168.27.2>
  ServerName vh2.mysite.be
  DocumentRoot /var/www/vh/vh2
  <Directory /var/www/vh/vh2>
    AuthName "Acces au site vh2"
    AuthType Basic
    AuthUserFile /var/www/securite/pwd
    Require valid-user
  </Directory>

  ErrorLog "logs/error_log"
  LogLevel warn
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog log/access_log.vh2 common
</VirtualHost>
```

Peut être placée avec les autres conteneurs <Directory ...>

```
<VirtualHost 192.168.27.2>
  ServerName vh3.mysite.be
  DocumentRoot /var/www/vh/vh3

  ErrorLog "logs/error_log"
  LogLevel warn
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog log/access_log.vh3 common
</VirtualHost>
```

Testez votre nouvelle configuration.

```
# systemctl restart httpd.service
```

Via un navigateur:

```
http://www.mysite.be  
→ access_log mis à jour avec l'agent
```

```
http://louis.mysite.be  
→ access_log mis à jour sans l'agent
```

```
http://jean.mysite.be  
→ access_log mis à jour sans l'agent
```

```
http://vh1.mysite.be  
→ access_log mis à jour sans l'agent  
(.*)\.gif$" "http://autre.example.com$1.jpg"
```

```
http://vh2.mysite.be  
→ access_log mis à jour sans l'agent mais avec l'identifiant
```

```
http://vh3.mysite.be  
→ access_log mis à jour sans l'agent
```

```
# cd /var/log/httpd  
# > error_log  
# systemctl restart httpd.service  
# cat error_log
```

→ contient toutes une série d'événements de n'importe quel niveau de criticité.

Exercice 6

Continuez la configuration d'Apache pour que:

- La page d'accueil de `www.mysite.be` offre en plus la possibilité d'accéder à un secret via un lien "Cliquez ici pour voir le secret".

Modification du code html de la page d'accueil du site web de base

```
/var/www/html/index.html
<html>
<head>
<title> Test du site web de base </title>
</head>
<body>
<h1>Site web de base. Ca marche !!! </h1>
<h3>
<a href=https://secure.mysite.be> Cliquez ici pour voir le secret </a>
</h3>
</body>
</html>
```

- Un clic sur ce lien nous redirige vers la page d'accueil de `secure.mysite.be` d'un hôte virtuel SSL hébergé par Ip sur `192.168.27.100`. Cette page sera enregistrée dans `/var/www/sec/websecure` et contiendra le secret.

Création du code html de la page d'accueil du site web sécurité

```
# mkdir -p /var/www/sec/websecure

/var/www/sec/websecure/index.html
<html>
<head>
<title> Test du site web securise</title>
</head>
<body>
<h1>Cette page a été reçue entièrement chiffrée... </h1>
</body>
</html>
```

Mise à jour de /etc/hosts

```
...
192.168.27.100  secure.mysite.be
```

Création d'une interface /etc/sysconfig/network-scripts/ifcfg-enp0s3:5

```
...
IPADDR=192.168.27.100
NETMASK=255.255.255.0
ONBOOT=yes
NAME=enp0s5:0
DEVICE=enp0s5:0

+ # shutdown -r now
```

Configuration de /etc/httpd/conf/01-ssl.conf

```
...
<VirtualHost 192.168.27.100:443>
  ServerName secure.mysite.be
  DocumentRoot /var/www/sec/websecure
  SSLEngine on
  SSLCertificateFile /etc/pki/tls/certs/secure.mysite.be.crt
  SSLCertificateKeyFile /etc/pki/tls/private/secure.mysite.be.key
</VirtualHost>
```

Installation des modules openssl et mod_ssl (openssl est certainement déjà installé)

```
# yum install openssl mod_ssl -y
```

Implémentation des clés et des certificats

Création de la clé privée du serveur

```
# cd /tmp
# openssl genrsa -out secure.mysite.be.key 2048
```

Création du CSR

```
# openssl req -new -key secure.mysite.be.key -out secure.mysite.be.csr
```

```
...
Remplir tous les champs d'information
concernant le formulaire.
```

```
...
```

Création du certificat auto-signé

```
# openssl x509 -in secure.mysite.be.csr -out secure.mysite.be.crt
    -req -signkey secure.mysite.be.key -days 3650
```

Copie des clés et certificats au bon endroit

```
# cp secure.mysite.be.crt /etc/pki/tls/certs
# cp secure.mysite.be.csr /etc/pki/tls/private
# cp secure.mysite.be.key /etc/pki/tls/private
```

Suppression des clés et certificats de /tmp

```
# rm -f /tmp/secure.mysite.be*
```

Relancer Apache

```
# systemctl restart httpd.service
```

Soyons curieux

```
# httpd -S | more
```

→ On voit qu'Apache peut recevoir des requêtes destinées au port 443 (https) sur l'Ip 192.168.27.100 . Le site associé (secure.mysite.be) est configuré dans 01-ssl.conf.

Test à l'aide d'un navigateur

http://www.mysite.be → On a la page d'accueil de www.mysite.be

```
...
```

On clique sur le lien

→ On reçoit un message disant que la connexion n'est pas certifiée

→ Si on "Comprends les risques"... et on "Ajoute une exception non-permanente"

→ On a la page d'accueil du site sécurisé secure.mysite.be

- . **Importez le certificat dans Firefox de façon à ne plus avoir à "Comprendre les risques" lors de l'accès au site SSL.**

Edition

- Préférences
 - Avancé
 - Chiffrement
 - Afficher les certificats
 - Autorité
 - Importer
 - Ouvrir /etc/pki/tls/certs/secure.mysite.be.crt
 - Confirmer votre confiance pour identifier les sites web + OK

*Le certificat apparaît sous le nom de l'institution donnée lors de la création du CSR.
Les accès ultérieurs au site SSL se feront dorénavant sans avoir besoin à chaque fois de "Comprendre les risques".*

- . **Tracez à l'aide d'un renifleur toute la conversion de `www.mysite.be` jusqu'à `secure.mysite.be`.**

Capture filter sur lo: tcp port 80 or tcp port 443

- **les requêtes `http://vh3.mysite.be` et `https://vh3.mysite.be` nous offrent, de manière sécurisée, la page d'accueil du site.**

- . **générez un nouveau certificat auto-signé pour ce site**

Idem que pour le site `secure.mysite.be` mais à adapter pour `vh3.mysite.be` ...

- . **les 2 requêtes seront envoyées sur l'ip `192.168.27.2`**

/etc/httpd/conf.d/00-1main.conf

...

```
<VirtualHost 192.168.27.2>
  ServerName vh3.mysite.be
  DocumentRoot /var/www/vh/vh3
  Redirect permanent "/" "https://vh3.mysite.be"
  ErrorLog "logs/error_log"
  LogLevel warn
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog log/access_log.vh3 common
</VirtualHost>
```

```

/etc/httpd/conf/01-ssl.conf
...
<VirtualHost 192.168.27.2:443>
  ServerName vh3.mysite.be
  DocumentRoot /var/www/vh/vh3

  ErrorLog "logs/error_log"
  LogLevel warn
  LogFormat "%h %l %u %t \"%r\" %>s %b" common
  CustomLog log/access_log.vh3 common

  SSLEngine on
  SSLCertificateFile /etc/pki/tls/certs/vh3.mysite.be.crt
  SSLCertificateKeyFile /etc/pki/tls/private/vh3.mysite.be.key
</VirtualHost>

+

# systemctl restart httpd.service

```

. vérifiez par les logs de access_log.vh3 que:

http://vh3.mysite.be **génère 2 logs. Pourquoi ?**

```

192.168.27.1 - - [10/Sep/2017:12:14:23 +0200] "GET / HTTP/1.0" 301 229
192.168.27.1 - - [10/Sep/2017:12:14:28 +0200] "GET / HTTP/1.0" 200 116

```

La 1ère ligne provient des logs configurés dans 00-server.conf lors de la demande de la page d'accueil redirigée par le Redirect permanent
→ status: 301

La 2ème ligne provient des logs configurés dans 01-ssl.conf lors de la demande de la page d'accueil qui est bien envoyée (chiffrée) → status : 200

https://vh3.mysite.be **génère 1 log. Pourquoi ?**

```

192.168.27.1 - - [10/Sep/2017:13:14:37 +0200] "GET / HTTP/1.0" 200 116

```

Ici, on accède directement au virtual host SSL (plus d'accès par http!). Cette unique ligne provient donc des logs configurés dans 01-ssl.conf lors de la demande de la page d'accueil qui est bien envoyée (chiffrée)
→ status : 200