Configuración básica de Apache

Instalación de Apache

El primer paso será instalar el servidor web Apache

```
sudo apt-get install apache2
```

Para verificar que está corriendo, visita http://127.0.0.1

Ficheros de configuración de Apache

Apache se puede configurar de varias maneras:

1. En el fichero `/etc/apache2/apache2.conf`. Las directivas aquí definidas se aplican a todos los hosts virtuales activos en nuestro servidor. Ya modificamos este archivo cuando instalamos **phpMyAdmin**
2. En los ficheros de configuración de cada uno de los hosts virtuales. Para cada host virtual se pueden definir una serie de directivas en su fichero de configuración que sobrescribirán a las definidas con el método 1.
3. En un fichero llamado `.htaccess` colocado en raíz de uno de los directorios del `DocumentRoot` del host virtual que sobrescriben a las definidas mediante los métodos 1 y 2.

¿Qué método elegir? Depende del control que poseas sobre la instalación.

Si tienes acceso a modificar el archivo `apache2.conf` y es una configuración que deseas **aplicar a todos los hosts virtuales**, mejor hazla ahí. El problema surge cuando despliegas la aplicación en un hosting compartido. Seguramente no te dejarán tocar el archivo de configuración general de apache, por lo que deberás usar el método 2, preferiblemente. Si por el contrario, tienes una máquina propia (como por ejemplo en Amazon Web Services - AWS) se recomienda este método. Tiene una desventaja: cuando modificas este archivo debes reiniciar apache (`sudo service apache2 restart`), con todo lo que ello implica (nuestro servidor estará caído durante el tiempo que tarde en reiniciar). Por tanto, las modificaciones en este archivo se deben pensar con cuidado antes de que

nuestra web entre en producción ya que cualquier modificación va a suponer pérdida de conexión de todos nuestros hosts virtuales.

Si la modificación sólo es para un **host virtual o directorio dentro de éste**, como por ejemplo definir `DocumentRoot` o alguna directiva que sólo se aplica a tu host virtual, o no puedes acceder a `apache2.conf`, hazla en el fichero de configuración de tu sitio. Las modificaciones en este archivo no implican reiniciar apache; sólo basta con hacer una recarga (`sudo service apache2 reload`), por lo que el sitio no estará caído mientras está en proceso de recarga. También es verdad que algunos hostings tampoco dejan modificar este archivo.

Por último, tenemos siempre la opción de definir las directivas en un archivo `.htaccess`. Este método siempre lo podrás aplicar en la compañía de hosting ya que no es más que otro archivo de los muchos que estarán alojados en tu `DocumentRoot`. Cuando se modifica este archivo, no hay que hacer ni `reload` ni `restart` sino que cuando un cliente hace una petición de un recurso que *cuelga* del directorio donde está el archivo `.htaccess`, **apache** lo lee y aplica las directivas definidas en él. Es decir, **para cada petición debe leer este archivo!** Por lo que sólo usa esta opción en caso de necesidad.

Host virtuales

Hemos visto en el apartado anterior que en un mismo servidor podemos tener alojadas varias webs.

Cada una de estas webs se llaman **Host Virtuales**.

Vamos a crear 2 hots virtuales en nuestra instalación de **apache** de tal forma que podamos activarlos/desactivarlos a nuestro antojo. Cada uno de los hosts virtuales se definen con su propia configuración dentro de la carpeta de instalación de apache, que suele ser `/etc/apache2`

A continuación se muestra la estructura de directorios de una instalación típica:

```
root@pedrablava:/etc/apache2# ls -la
total 96
drwxr-xr-x   8 root root  4096 sep 20 16:53 .
drwxr-xr-x 149 root root 12288 sep 25 19:35 ..
-rw-r--r--   1 root root  7115 ene  7  2014 apache2.conf
drwxr-xr-x   2 root root  4096 sep 20 16:53 conf-available
drwxr-xr-x   2 root root  4096 sep 20 16:53 conf-enabled
-rw-r--r--   1 root root  1782 ene  3  2014 envvars
-rw-r--r--   1 root root 31063 ene  3  2014 magic
drwxr-xr-x   2 root root 12288 sep 20 16:57 mods-available
drwxr-xr-x   2 root root  4096 sep 20 16:57 mods-enabled
-rw-r--r--   1 root root   320 ene  7  2014 ports.conf
drwxr-xr-x   2 root root  4096 sep 20 16:53 sites-available
drwxr-xr-x   2 root root  4096 sep 20 16:53 sites-enabled
```

Todos los hosts se definen dentro de la carpeta `sites-available` y los que están activos están en la carpeta `sites-enabled`.

```
root@pedrablava:/etc/apache2/sites-available# ls -la
total 20
drwxr-xr-x 2 root root 4096 sep 20 16:53 .
drwxr-xr-x 8 root root 4096 sep 20 16:53 ..
-rw-r--r-- 1 root root 1332 ene  7  2014 000-default.conf
-rw-r--r-- 1 root root 6437 ene  7  2014 default-ssl.conf
```

```
root@pedrablava:/etc/apache2/sites-enabled# ls -la
total 8
drwxr-xr-x 2 root root 4096 sep 20 16:53 .
drwxr-xr-x 8 root root 4096 sep 20 16:53 ..
lrwxrwxrwx 1 root root   35 sep 20 16:53 000-default.conf -> ../sites-available/
000-default.conf
```

Por tanto, podemos tener definidos muchos sites en `sites-available` pero no tenemos porqué tenerlos todos activos a la vez.

Para crear un host virtual (suponiendo que se llama `website1`), debemos crear un archivo llamado `website1.conf` (el nombre del archivo de configuración no importa) dentro se `sites-available` y editar su configuración. Por ejemplo, una configuración mínima sería:

```
<VirtualHost website1.local:80>
    ServerName www.website1.local
    ServerAlias website1.local
    DocumentRoot /home/victor/Documentos/webs/website1
    <Directory "/home/victor/Documentos/webs/website1/">
        AllowOverride All
        Require all granted
        Allow from all
    </Directory>
</VirtualHost>
```

Figure 1: 1537281856382

La directiva `ServerName` sirve para establecer el nombre del host virtual, mientras que `ServerAlias` sirve para especificar un nombre alternativo. Si tenemos más de un virtual host, `ServerName` o `ServerAlias` deben coincidir con el nombre del host virtual, en este caso **website1.local**.

`DocumentRoot` establece el directorio donde van a residir los documentos pertenecientes al host virtual. Las directivas para un directorio concreto se especifican dentro de `Directory` (ya las veremos).

Para hacer pruebas en nuestro ordenador, hemos de modificar las rutas en nuestro equipo para que pueda asociar estos nombres (ficticios) con la ip local. Debemos editar el fichero `hosts` para que nos devuelva la dirección del bucle local (127.0.0.1) cuando el navegador pida la url www.website1.local

Este fichero está en `/etc/hosts`

```
127.0.0.1   localhost
127.0.0.1   website1.local
127.0.0.1   www.website1.local
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

El siguiente paso será activar (*enable*) dicho host virtual. Para ello usamos el comando `a2ensite nombre-de-archivo-de-configuración`

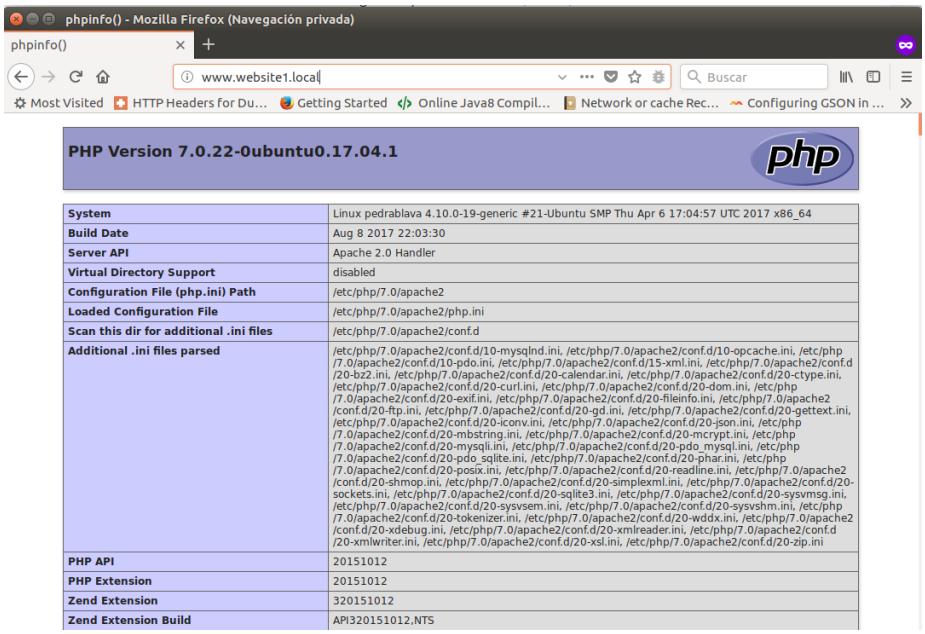
Por ejemplo, para activar el sitio **website1**, ejecutaríamos `a2ensite website1.conf`

También debemos crear la estructura de directorios del sitio. Para ello hemos de crear un directorio (`mkdir`) en la ruta que hemos definido en `DocumentRoot` del archivo de configuración. Como este directorio está vacío, vamos a crear un archivo `index.php` para que sea devuelto por defecto.

```
<?php
phpinfo();
?>
```

> Si queremos desactivar un sitio se utiliza el comando `a2dissite nombre-de-archivo-de configuración`

Si todo ha ido bien, al visitar la url www.website1.local nos devolverá la información de la configuración de php



Permisos

Es muy importante establecer los permisos de los directorios correctamente o puedes encontrarte con el siguiente error cuando accedas a tu host virtual.

Figure 2: 1537284433352

En sistemas Unix, los permisos de un archivo o directorio se definen para el propietario, el grupo al que pertenece el mismo y otros usuarios.

Por ejemplo:

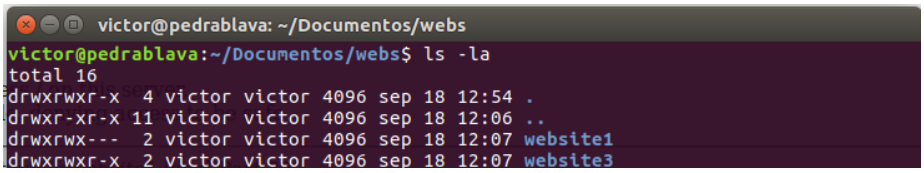


Figure 3: 1537285319233

Por ejemplo, en el directorio `website1` que pertenece al usuario **victor** y grupo **victor**, indica **drwxrwx**—:

- es un directorio porque empieza por **d**
- El usuario `victor` tiene permisos de lectura (read), escritura (write) y ejecución (x). Lo sabemos con los tres siguientes caracteres d**rwx**rwx—
- El grupo `victor` al que pertenece el directorio tiene permisos de lectura (read), escritura (write) y ejecución (x). Lo sabemos con los tres siguientes caracteres drwx**rwx**—
- Otros usuarios (que no sean el propietario ni pertenezcan a dicho grupo) no tienen ningún permiso drwxrwx—. Cuando aparece un guión significa que no tiene permiso.

Cuando un cliente hace una petición a un recurso de nuestro servidor lo hace como un usuario sin autenticar en el sistema, y **apache** actúa de su parte intentando acceder a dicho recurso con sus credenciales propias, que suelen ser usuario `www-data` y grupo `www-data`.

Por tanto, en el caso del directorio `website1`, no va a poder acceder porque ni es el propietario ni pertenece al grupo. Hemos de asegurarnos de que sí tiene permisos, por lo menos de lectura.

Si cambiamos los permisos, mediante:

```
chmod -R 775 website1
```

Ahora sí que puede acceder porque los permisos para *otros* están definidos como de **lectura** y de **ejecución**.

Si además, nuestra web tiene la opción de subir archivos de cualquier tipo, también debemos darle permiso de escritura.

> **Cuidado:**
>
> Lo normal es que en un servidor real, los archivos de `DocumentRoot` *cuelguen* de `/var/www/html` y los permisos ahí están definidos como de acceso para cualquier usuario.

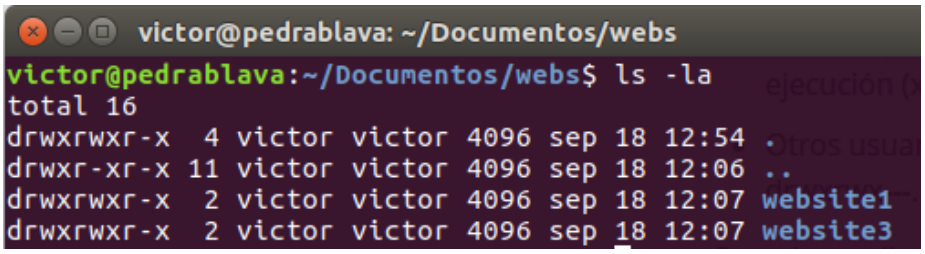


Figure 4: 1537286094413

> Sin embargo, si las prácticas las vamos haciendo en un directorio que esté en nuestro `home`, **apache** va a seguir sin poder acceder, a no ser que definamos la directiva `Require all granted` para `DocumentRoot` dentro del archivo de configuración de nuestro host virtual.
>
> **Nunca pero nunca**, des permisos 777 a tu servidor en producción. Si hace falta que **apache** acceda en modo escritura a tu `DocumentRoot` piensa en cambiar el grupo al que pertenece **apache** o usa cualquier otra técnica.
>
> Más información en https://askubuntu.com/questions/325498/apache-cant-access-folders-in-my-home-directory y en https://serverfault.com/questions/357108/what-permissions-should-my-website-files-folders-have-on-a-linux-webserver

¿Cómo cambiar los permisos en Linux?

Para cambiar los permisos se usa el comando `chmod`.

Por ejemplo, `chmod 745 file1`. Pero, ¿qué significa ese número? El primer dígito (**7**) especifica los permisos del *usuario*, el segundo dígito (**4**) especifica los permisos del *grupo* y el último dígito (**5**) especifica los permisos para *otros*.

Y ¿de dónde salen esos dígitos? Pues de una técnica que se usa mucho en informática y que se llama máscara de bits. Veamos cómo funciona:

- Permiso de lectura (r) tiene el valor 4
- Permiso de escritura (w) tiene el valor 2
- Permiso de ejecución (x) tiene el valor 1

Por ejemplo, si queremos dar permisos de lectura y escritura usaremos el valor **6**, resultante de sumar **4 + 2**, que en binario es **110**

Tarea 1

1. Cread dos hosts virtuales usando el método descrito en este documento (website1 y website2).
2. Una vez creados, cread un documento `markdown` con las capturas de pantalla de los directorios `sites-available` y `sites-enabled`, así como de los directorios creados dentro del `DocumentRoot` definido en cada uno de ellos.
3. Así mismo debéis adjuntar la configuración de vuestros sitios en apache así como el archivo `/etc/hosts`
4. Finalmente, adjuntad una captura de pantalla del navegador pidiendo esos dos sitios.

Utilización de directorios virtuales

Los directorios virtuales son directorios que se encuentran fuera del directorio raíz del servidor (`DocumentRoot`). Si no se definen directorios virtuales, todos los documentos accesibles desde Internet en vuestro sitio deberán *colgar* del directorio definido en `DocumentRoot`.

Existen dos formas de crearlos:

1. Utilizando un **Alias**.
2. Creando un **enlace simbólico** (*symlink*) dentro del directorio raíz (se debe configurar con la opción `FollowSymLinks`) que apunte a otro directorio.

Vamos a realizar la práctica sólo con la primera opción dado que la segunda, además de modificar el archivo de configuración exige la creación desde el sistema operativo de enlaces simbólicos entre el directorio físico y el directorio virtual.

Primero debemos crear el directorio físico, mediante la orden `mkdir`

El directorio debéis crearlo en `/usr/share/wiki`. Lo creamos en este directorio para no tener problemas de permisos.

Dentro de este directorio crear una página básica **index.html**

Finalmente creamos el alias dentro del archivo de configuración del sitio, añadiremos una sentencia `Alias` para crear un directorio virtual denominado `/wiki` que referencie a `/usr/share/wiki`

Tarea 2

Configurad vuestro servidor apache de esta manera y adjuntad las capturas de pantalla tanto del archivo de configuración de apache como del navegador apuntado a dicho alias.

```
<VirtualHost website1.local:80>
.......
    Alias /wiki /usr/share/wiki
    <Directory /usr/share/wiki/>
        DirectoryIndex index.html
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
.......
</VirtualHost>
```

Figure 5: 1537282410116

Redirect

Utilizando la directiva `Redirect` podemos crear una redirección de un recurso hacia una URL. Por ejemplo, vamos a crear una redirección de `/ieselcaminas` hacia la URL http://www.ieselcaminas.org . Para ello modificaremos el archivo de configuración:

```
Redirect /ieselcaminas http://www.ieselcaminas.org
```

Figure 6: 1537282492432

Tarea 3

> Realiza esta modificación en tu archivo de configuración de apache. Adjunta el contenido de tu archivo de configuración de apache y una captura de pantalla de Firebug donde se vea el `Status Code 302`

Error 404

Siguiendo la información que puedes encontrar en este enlace, configura tu instalación de apache para que muestre una página html con un mensaje de error personalizado.

Tarea 4

> Realiza esta modificación en tu archivo de configuración.
> Crea un documento `markdown` con esta información y adjunta una

captura de pantalla de firebug mostrando esta página y en la que se vea el `status code 404`