


# UD.2

# ADMINISTRACIÓN DE SERVIDORES

## web

Despliegue de aplicaciones web

# ÍNDICE

1. Configuración del servidor web
  2. Módulos
  3. Instalación, configuración y uso de los módulos de Apache
  4. Host virtuales
  5. Control de acceso
  6. Autenticación y autorización
  7. El protocolo HTTPS
  8. HTTPS en Apache
- 

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

- ▶ En Apache, la configuración por defecto está en  
`/etc/apache2/httpd.conf`
  - En las distribuciones *Ubuntu*, el archivo base de configuración es  
`/etc/apache2/apache2.conf`
- ▶ Edición del archivo de configuración
  - Realizar una copia de seguridad

```
$ cd /etc/apache2/
$ sudo cp apache2.conf apache2.conf.old
```
  - Editarlo

```
$ gedit apache2.conf
```

Contiene: **Directivas** para el servidor y **comentarios**

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

- ▶ **#** Los **comentarios** empiezan por almohadilla y son ignorados por el servidor
- ▶ Las **directivas** para el servidor indican al servidor cómo actuar:
  - Dónde está el directorio que contiene los documentos web
  - Qué módulos se cargarán
  - Información sobre host virtuales

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

- ▶ Los **módulos multiproceso** (MPMs– Multiprocessing Modules) se introducen con Apache 2
  - Cada módulo MPM crea **hilos o procesos hijos** (prefork) para atender las peticiones.
  - Existen varios módulos e incluso dependen del sistema operativo

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

**\$ sudo gedit /etc/apache2/apache2.conf**

- En servidores pequeños puede ser suficiente un solo fichero de configuración con todas las directivas.
- En servidores más complejos es necesario dividir estos archivos siguiendo un criterio lógico.
- Hay includes de todo el contenido de un directorio
  - Permiten crear archivos de configuración de elementos no asimilables lógicamente a otro lugar
  - Para realizar pruebas en la configuración que sean fácilmente reversibles
  - Para añadir configuraciones asociadas a módulos concretos que pueden habilitarse y deshabilitarse

```
/etc/apache2/  
#      |-- apache2.conf  
#      |      `-- ports.conf  
#      |-- mods-enabled  
#      |      |-- *.load  
#      |      `-- *.conf  
#      |-- conf-enabled  
#      |      `-- *.conf  
#      `-- sites-enabled  
#           `-- *.conf
```

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

7

## *Directivas del archivo de configuración*

- Las directivas le indican al servidor cómo actuar.
- Todo cambio en el fichero de configuración conlleva el reinicio del servidor.

**\$ sudo apachectl restart**

- Más información:
  - Índice de directivas <http://httpd.apache.org/docs/2.4/en/mod/directives.html>
  - Guía Rápida de Referencia de Directivas  
<http://httpd.apache.org/docs/2.4/en/mod/quickreference.html>
  - Términos usados para describir las directivas  
<http://httpd.apache.org/docs/2.4/mod/directive-dict.html>



# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **ServerRoot** #ServerRoot "/etc/apache2"
  - Indica el directorio raíz de la instalación (no es el que contiene las páginas web que se especifica por DocumentRoot).
  - Aparece comentada porque es el valor por defecto.
  - Esta directiva solo se modificaría en caso de mover el servidor Apache a otra ubicación en la estructura de directorios.
  - Se puede utilizar para definir rutas relativas al directorio de instalación de Apache, usaríamos %ServerRoot% en lugar de la ruta completa.



# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **PidFile** **PidFile logs/httpd.pid**
  - Establece la ruta al archivo en el que el servidor graba su ID de proceso (pid).
  - Por defecto se coloca en **%ServerRoot%/logs**
  - Aquí, aparece definida mediante una constante, la cual se define en **/etc/apache2/envvars**
  - No se recomienda cambiar la ruta
- **TimeOut** **Timeout 300**
  - Son los segundos que se esperan las respuestas durante la comunicación.
  - Por defecto se coloca en 300 (no cambiarlo)

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **KeepAlive** KeepAlive on
  - Determina si el servidor va a permitir o no que se haga más de una petición.
  - Si se activa, conviene poner `KeepAliveTimeout 5` para que un usuario no sature el servicio.
- **MaxKeepAliveRequest** MaxKeepAliveRequest 100
  - Número de peticiones que podrá realizar cada conexión
- **KeepAliveTimeout** KeepAliveTimeout 5
  - Tiempo que el servidor esperará antes de atender una nueva petición del mismo cliente en la misma conexión.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **User y Group**

User nobody  
Group #-1

- Estas directivas indican con que usuario y grupo se lanzarán los procesos hijos que genere Apache
- El usuario **no** debe ser **root** por seguridad
- Si el servidor se lanza con un usuario distinto de root, los procesos hijos tendrán ese mismo usuario ya que no se podrá modificar, únicamente el usuario root puede cambiarlo.
- En esta distribución se utilizan dos variables globales.

- **HostnameLookups**

HostnameLookups off

- Indica al servidor si debe hacer éste una consulta DNS para cada petición.
- Por defecto está desactivada para economizar recursos.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **ErrorLog** %ServerRoot%logs/error\_log
  - Indica dónde ubicar el archivo de registro de los errores del servidor
- **LogLevel** LogLevel warn
  - Establece cuánta información se guardará en el archivo de registro de errores.
  - Los valores posibles incluyen:  
debug, info, notice, warn, error, crit, alert, emerg.
  - Por defecto tiene el valor `warn` que es suficiente para empezar.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **Directory**
  - Configura cómo se comportará y qué se permitirá en cada directorio al que tiene acceso el servidor Apache
  - Se aplica al directorio y sus subdirectorios salvo que haya otra directiva `directory` propia del subdirectorio

```
<Directory />  
    Options FollowSymLinks  
    AllowOverride None  
    Require all denied  
</Directory>
```

Estas se aplicarán al directorio raíz y a todos subdirectorios

Opciones muy restrictivas. Impiden que nadie que acceda al servidor pueda acceder a ningún directorio de la estructura de nuestro servidor.

- Ver las otras directivas Directory

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **Directory (cont.)**
  - Pueden añadirse otras directivas relativas a otros directorios.

```
<Directory /usr/share>
    AllowOverride None
    Require all granted
</Directory>
```

Directorio compartido del usuario

Directorio DocumentRoot

```
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Opciones más permisivas:

- Permite a Apache seguir enlaces lógicos
- Indica permisos, permite el acceso a todos.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Directivas del archivo de configuración*

- **AccessFileName** AccessFileName .htaccess
  - Indica nombre del archivo en el que se debe buscar las directivas de acceso de cada directorio.
  - El valor de defecto es `.htaccess` y no debe cambiarse.

- **Files**
  - Contenedor para establecer directivas para tipos de archivos

```
<FilesMatch "^\.ht">  
    Require all denied  
</FilesMatch>
```

Deniega el acceso a los ficheros que empiezan por `.ht` por cuestiones de seguridad  
Ej. `.htaccess` y `.htpasswd`



# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas del archivo de configuración: **ports.conf***

- **Listen**
  - Indica qué dirección y puerto debe escuchar el servidor las peticiones http que lleguen además de los de defecto.
  - Puerto estándar en el que el servidor recibe peticiones es 80.
  - En máquinas virtuales escuchará en la interfaz de loopback 127.0.0.1
  - Si configuramos Apache para escuchar en otro puerto, deberemos acceder a las páginas web del servidor especificando el puerto tras la dirección.
  - Si queremos que el servidor sea visto desde la máquina host deberemos configurarlo especificando la IP de la tarjeta de red que añadimos como solo anfitrión.

```
Listen 192.168.56.101:80
```

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas del archivo de configuración de los módulos habilitados:*

/etc/apache2/**mods-enabled/**

**mpm\_event.conf**

- **ifModule**

- Es un contenedor que permite establecer determinadas opciones solo si se ha cargado un módulo determinado.

```
<IfModule mod_mime_magic.c>  ← Modulo cargado
    MIMEMagicFile conf/magic  ← Opción establecida
</IfModule>
```

- Si ponemos `<IfModule !mod_mime_magic.c>` establecería la propiedad cuando no se cargase ese módulo
- Esta directiva no aparece en la configuración principal, se ha desplazado al archivo de configuración de cada módulo.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración de los módulos habilitados:*

`/etc/apache2/mods-enabled/`

└─ `mpm_event.conf`

- **StartServers**

```
<IfModule mpm_event_module>
```

```
    StartServers                2
```

```
    MinSpareThreads      25
```

```
    MaxSpareThreads      75
```

```
    ThreadLimit          64
```

```
    ThreadsPerChild      25
```

```
    MaxRequestWorkers    150
```

```
    MaxConnectionsPerChild  0
```

```
</IfModule>
```

← Modulo cargado

← Opción establecida:

- Número inicial de los procesos de servidor al arrancar
- Rango de subprocesos de trabajo que se mantienen de reserva
- Número constante de subprocesos de cada proceso de servidor
- número máximo de subprocesos de trabajo
- número máximo de peticiones que sirve un proceso de servidor

- Dentro del archivo `/etc/apache2/mods-enabled/mpm_event.conf`
- Esta directiva establece cuantos servidores se crearán al arrancar.

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

19


*Directivas de los archivos de configuración de los módulos habilitados:*  
**/etc/apache2/mods-enabled/**

**dir.conf**

- **DirectoryIndex**

```
<IfModule mod_dir.c>  
    DirectoryIndex index.html index.cgi index.pl index.php  
    index.xhtml index.htm  
</IfModule>
```

Sucesión de archivos, se busca  
y muestra por orden



- Especifica la página por defecto que se buscará al acceder a un directorio de la jerarquía de nuestro sitio
- Si en un directorio no hay ninguno de estos archivos Apache lo crea dinámicamente de forma que liste los contenidos.
- Dentro del archivo **/etc/apache2/mods-enabled/dir.conf**

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración de los módulos habilitados:*

**/etc/apache2/mods-enabled/**

↳ **alias.conf**

- **Alias**

```
<IfModule alias_module>
  Alias /icons/ "/usr/share/apache2/icons/"
  <Directory "/usr/share/apache2/icons">
    Options FollowSymLinks
    AllowOverride None
    Require all granted
  </Directory>
</IfModule>
```

Nombre real

- Permite crear alias para archivos o directorios
- Dentro del archivo **/etc/apache2/mods-enabled/alias.conf**

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración de los módulos habilitados: /etc/apache2/mods-enabled/*

└─ **setenvif.conf**

- **BrowserMatch**

Evita problemas con navegadores que no sigan algún estándar

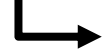
```
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0
    force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
```

- Permite modificar la respuesta dependiendo de la configuración del cliente en cuanto a navegador y plugins
- Dentro del archivo **/etc/apache2/mods-enabled/setenvif.conf**

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración de los módulos*

*habilitados: /etc/apache2/mods-enabled/*



**negotiation.conf**

- LanguagePriority

```
LanguagePriority en ca cs da de el eo es et fr he hr  
it ja ko ltz nl nn no pl pt pt-BR ru sv tr zh-CN zh-  
TW
```

- Permite establecer la prioridad de los idiomas en caso de que no se especifique uno o haya un empate en la negociación por diferentes motivos.
- Dentro del archivo `/etc/apache2/mods-enabled/negotiation.conf`



# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración habilitados:*

`/etc/apache2/conf-enabled/`

↳ `serve_cgi_bin.conf`

- ScriptAlias

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/  
<Directory "/usr/lib/cgi-bin">  
    AllowOverride None  
    Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch  
    Require all granted  
</Directory>
```

Sólo se debe incluir si  
se va a usar scripts CGI

- Indica dónde se ubicará la carpeta de los scripts CGI
- Dentro del archivo `/etc/apache2/mods-enabled/serve_envif.conf`

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración de los virtual host disponibles:* `/etc/apache2/sites-available/`

└─ `000_default.conf`

- **ServerAdmin**
  - Configura la dirección del administrador del servidor web que se mostrará al generarse una página de error
- **ServerName**
- **DocumentRoot**
  - Indica el directorio raíz donde se colocarán las páginas web.
  - ▶ Dentro del archivo `/etc/apache2/sites-available/000-default.conf`

```
ServerAdmin webmaster@localhost  
  
#ServerName www.example.com  
  
DocumentRoot /var/www/html
```

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

*Directivas de los archivos de configuración disponibles:*

`/etc/apache2/conf-available/  
└─ security.conf`

- **ServerSignature**
  - Indica, si esta activa, que al mostrarse una página generada automáticamente por el propio servidor (pag. de error, listados de directorios,...) se debe poner el nombre y la versión del servidor.

<code>ServerSignature</code>	<code>On</code>
------------------------------	-----------------

- ▶ Puede utilizarse maliciosamente por lo que conviene que aparezca deshabilitada
- ▶ Dentro del archivo `/etc/apache2/sites-available/security.conf`

# 1. CONFIGURACIÓN DEL SERVIDOR WEB

## *Registros de error:*

### ErrorLog

- Establece dónde ubicar el archivo de registro de los errores que se produzcan en el servidor.

```
ErrorLog    logs/error_log
```

### LogLevel

- Establece cuánta información se guardará en el archivo de registro de errores.

```
LogLevel    warn
```

### CustomLog

- Establece la ruta al archivo que registra las visitas a nuestro sitio web

```
#CustomLog  logs/access_log combined
```

## 2. MÓDULOS

El núcleo de Apache incluye la funcionalidad necesaria para establecer un servidor web pero existen muchos módulos adicionales que permiten añadir funciones extra.

Al instalar Apache, se habilita la opción *Dynamic Shared Object* –DSO– que permite añadir módulos dinámicamente sin necesidad de recompilar el servidor.

Los módulos compilados con el servidor no tiene por qué ser añadidos. La directiva *LoadModule* permite indicar que módulos dinámicos se cargan.

¿Qué módulos están instalados en nuestro servidor?

**sudo apachectl -l**

lista de módulos estáticos

**sudo apachectl -M**

lista de nombres de módulos  
(estática/dinámico)

Más información sobre los módulos de Apache:

- Lista de módulos <http://httpd.apache.org/docs/2.4/en/mod/>

## 2. MÓDULOS

1. Módulos relacionados con el entorno
2. Módulos de autenticación y control de acceso
3. Módulos de generación dinámica de contenidos
4. Módulos de configuración del tipo de contenido
5. Módulos para el listado de directorios
6. Módulos para la gestión de las cabeceras HTTP de las respuestas
7. Módulos de información del servidor y de registro de la actividad
8. Módulos de mapeo de URLs
9. Otros módulos

### Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

0. **core** – funcionalidades básicas del Servidor HTTP Apache que siempre están presentes.

### 1. Módulos **relacionados con el entorno**

Controlan qué parte del **entorno del servidor** (variables de entorno de Apache que modifican el comportamiento del servidor) está disponible para otros módulos o programas.

- **mod\_env** – permite pasar el valor de variables de entorno a programas CGI, perl, PHP,...
- **mod\_setenvif** – posibilita la creación de variables de entorno a partir de datos que nos envía el cliente con el protocolo HTTP
- **mod\_unique\_id** – establece un identificador único para cada petición que llega al servidor. Extensión

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```



## 2. MÓDULOS

### 2. Módulos de **autenticación y control de acceso**

Realizan la autenticación y control de acceso (para filtrar los usuarios que pueden visitar un directorio del sitio web a través de su dirección IP o del nombre de usuario).

**mod\_auth\_basic** – modulo básico de autenticación

**mod\_auth\_digest** – permite encriptar la contraseña

**mod\_authn\_file** – modulo de autenticación

**mod\_authz\_user** – modulo de autorización

**mod\_authz\_form** – a partir de apache 2.4, permite una autenticación mediante un formulario HTML.

**mod\_authz\_host** – permite la autenticación mediante IP o nombre de dominio

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

### 3. Módulos de **generación dinámica de contenidos**

Permiten delegar la atención de determinadas peticiones a diferentes scripts o programas externos.

**mod\_cgi** – permite ejecutar scripts de tipo CGI (Common Gateway Interface)

**mod\_include** – permite usar filtros SSI (Server-Side Includes)

**mod\_actions** – dependiendo del tipo MIME o en el método de la petición HTTP, permite usar diferentes scripts para procesar dichas peticiones.

**mod\_ext\_filter** – permite filtrar una respuesta mediante un programa externo antes de enviársela al cliente.

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

### 4. Módulos de configuración del tipo de contenido

Permiten al servidor detectar y negociar el tipo de contenido más adecuado para el cliente.

**mod\_mime** – permite que Apache determine el tipo MIME a partir de la extensión del archivo.

**mod\_mime\_magic** – permite determinar el tipo MIME a partir de un patrón de bytes que se almacena dentro del archivo. Sólo se activa cuando no es efectivo el módulo anterior.

**mod\_negotiation** – permite negociar el contenido entre cliente y servidor. El navegador indica qué es lo que puede manejar (idioma, codificación,...) y el servidor le responde de la forma más apropiada.

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status module (shared)
```

## 2. MÓDULOS

33

### 5. Módulos para el listado de directorios

Permiten configurar cómo será el listado del contenido de un directorio cuando un cliente accede a él mediante una URL de directorio y no se encuentra el archivo establecido para esos casos (indicado en la directiva *DirectoryIndex*)

**mod\_dir** – realiza dos funciones básicas: **redirecciona las URIs de directorios** (si no señalan a un fichero y no terminan en “/”) hacia la correcta URI (añade “/” al final) y **busca el archivo por defecto** para cargarlo cuando se accede a un directorio (*index.html* o lo especificado en *DirectoryIndex*)

**mod\_autoindex** – se encarga de **generar el listado** del contenido del directorio cuando se accede a él y no se encuentra el archivo correspondiente. Permite la configuración del propio listado.

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

### 6. Módulos para la **gestión de las cabeceras HTTP de las respuestas**

Las cabeceras en HTTP incluyen mucha información importante para la comunicación, este módulo permite modificar dichas cabeceras.

**mod\_asis**

**mod\_headers**

**mod\_expires**

**mod\_cern\_meta**

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

### 7. Módulos de **información del servidor y de registro de la actividad**

Proporcionan información sobre el estado del servidor y permiten configurar el registro de actividad.

**mod\_log\_config** – configura el registro de acceso de usuarios al servidor

**mod\_status** – Muestra información sobre el estado del servidor

**mod\_info** – Muestra información de configuración del servidor

**mod\_usertrack** – permite identificar usuarios y registrarlos de manera individual usando HTTP cookies. Este método permite el servicio “personalizado” a cada cliente.

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```



## 2. MÓDULOS

### 8. Módulos de **mapeo de URLs**

Manejar y modificar las URL de nuestro sitio, desde el nombre del dominio en adelante. Permitirá crear alias, tener varios sitios web en el mismo servidor y podremos rescribir las direcciones para que lleven a diferentes lugares de la estructura de archivos y directorios del servidor.

**mod\_userdir** – permite crear sitios personales para cada usuario

**\*mod\_alias** – permite crear alias o nombres simbólicos entre dos rutas de la estructura de archivos. Permite así, crear redirecciones de un archivo/directorio a otro.

**mod\_rewrite** – permite que el servidor pueda modificar a URL de la petición para que sea otra que hayamos configurado nosotros. Se utilizan patrones establecidos para filtrar URL ya no útiles (por cambio en el sistema de directorios del servidor) hacia otras activas que se correspondan

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```



## 2. MÓDULOS

37

### 8. Módulos de **mapeo de URLs**

Manejar y modificar las URL de nuestro sitio, desde el nombre del dominio en adelante. Permitirá crear alias, tener varios sitios web en el mismo servidor y podremos rescribir las direcciones para que lleven a diferentes lugares de la estructura de archivos y directorios del servidor.

**mod\_speling** – corrige pequeños errores en las URLs de las peticiones. Puede ralentizar al servidor.

- un carácter de más o de menos o cambio de un carácter por otro.
- Incorrecto uso de mayúsculas/minúsculas.

**mod\_vhost\_alias** – lo veremos cuando veamos host virtuales. permite tener varios sitios web en el mismo servidor Apache. No se usa pues crea dinámicamente los host virtuales y eso sólo se recomienda cuando se crean muchos.

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
setenvif_module (shared)
status_module (shared)
```

## 2. MÓDULOS

### 9. Otros módulos

**\*mod\_so** – permite añadir otros módulos sin la necesidad de recompilarlos. Permite el uso de DSO – *Dynamic Shared Object*– y todos los módulos restantes pueden usarse de esta manera menos este.

**mod\_imagemap** – incluye el soporte para mapas de imágenes en HTML

**mod\_proxy** – permite convertir a Apache en un servidor proxy

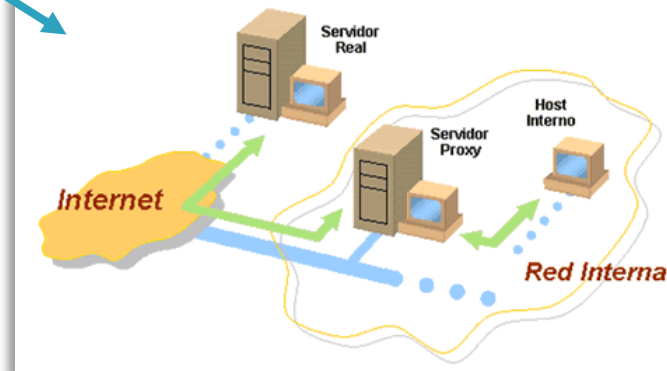
**mod\_file\_cache** – permite que determinados archivos estáticos y que no cambien se puedan almacenar en caché.

**mod\_dav** – para usar webDAV

**mod\_example\_hooks** – útil para quien quiera aprender a programar un módulo Apache (requiere conocer C)

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
```



# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## LOS ARCHIVOS DE CONFIGURACIÓN DE LA INSTALACIÓN POR PAQUETE

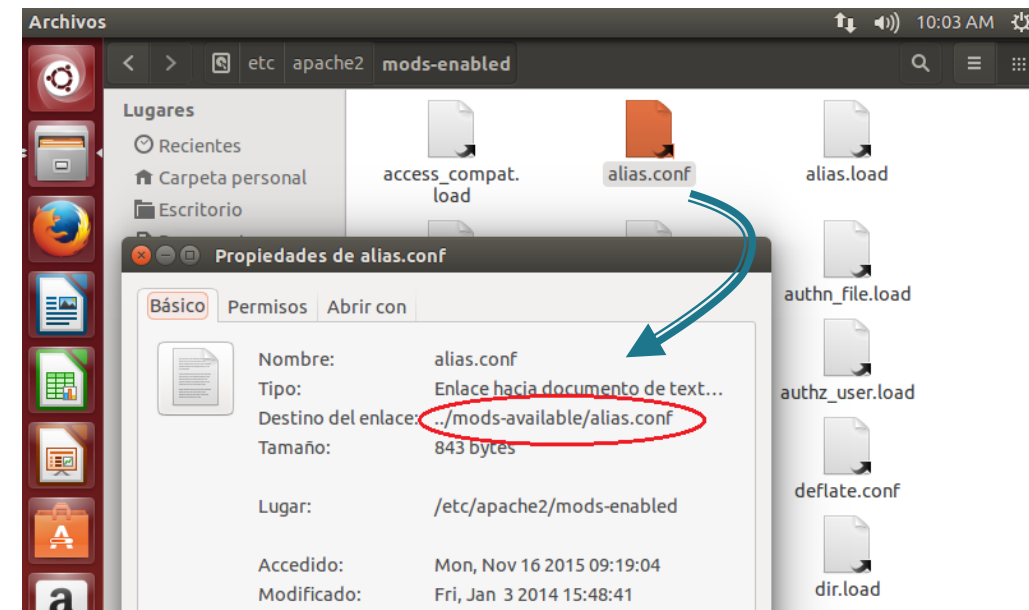
Esta instalación implica la distribución de las directivas de configuración en varios archivos. La directiva **include** permite realizar esta configuración de forma personalizada.

En nuestro caso:

```
# Include module configuration:  
IncludeOptional mods-enabled/*.load  
IncludeOptional mods-enabled/*.conf
```

Realmente, la carpeta **mod-enabled** solo contiene enlaces simbólicos a archivos de **mod-available**.

</etc/apache2/apache2.conf>



# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

The screenshot shows a file manager window titled 'mods-available' with a sidebar on the left containing various system icons. The main pane displays a list of files and folders. The 'alias.conf' and 'alias.load' files are highlighted with red boxes. A preview window for 'alias.conf' is open, showing the following configuration:

```
<IfModule alias_module>
  Alias /icons/ /usr/share/apache2/icons/"
  <Directory "/usr/share/apache2/icons">
    Options FollowSymLinks
    AllowOverride None
    Require all granted
  </Directory>
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

The path `/usr/share/apache2/icons/` is circled in red. Below it, a preview window for 'alias.load' shows the following configuration:

```
LoadModule alias_module /usr/lib/apache2/modules/mod_alias.so
```

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## LOS ARCHIVOS DE CONFIGURACIÓN DE LA INSTALACIÓN POR PAQUETE

Como ya hemos visto, los módulos que están instalados por defecto son:

Loaded Modules:

```
core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core module (shared)
```

`$ apachectl -M`

El hecho de que esté activo **so-module** indica que soporta DSO por lo que se pueden activar los módulos de manera dinámica.

Los módulos instalados y referenciados desde los archivos de `mods-available` se localizan en `/usr/lib/apache2/modules`

`$ cd /usr/lib/apache2/modules`

`$ ls -la`




# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## LOS ARCHIVOS DE CONFIGURACIÓN DE LA INSTALACIÓN POR PAQUETE

Hay otros módulos adicionales que no se incluyen en la instalación estándar de Apache.

**\$ sudo apt-cache search libapache2-mod**



```
libapache2-mod-apparmor - changehat AppArmor library as an Apache module
libapache2-mod-auth-mysql - Apache 2 module for MySQL authentication
libapache2-mod-auth-pgsql - Module for Apache2 which provides PostgreSQL
authentication
libapache2-mod-macro - Transitional package for apache2-bin subversion-dbg -
Debug symbols for Apache Subversion
libapache2-mod-auth-plain - Módulo para Apache2 que provee de autenticación en
texto plano
libapache2-mod-perl2 - Integración de perl con el servidor web Apache 2
libapache2-mod-perl2-dev - Integración de perl con el servidor web Apache 2 -
```

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## INSTALACIÓN

Para instalar un módulo en Apache hay que utilizar la directiva **LoadModule** del módulo `mod_so`.

**LoadModule module filename**

Ruta y nombre del archivo .so donde se encuentra

Nombre del módulo

Por ejemplo:



The screenshot shows a text editor window titled 'alias.load'. Inside, the line `LoadModule alias_module /usr/lib/apache2/modules/mod_alias.so` is displayed. The word `alias_module` is circled in blue, and the entire path `/usr/lib/apache2/modules/mod_alias.so` is also circled in blue. Above the screenshot, two blue arrows point from the text 'Nombre del módulo' to the circled `alias_module`, and from the text 'Ruta y nombre del archivo .so donde se encuentra' to the circled path.

Después será necesario reiniciar Apache **\$ apachectl restart**, comando que es equivalente a **\$ apache2 restart**

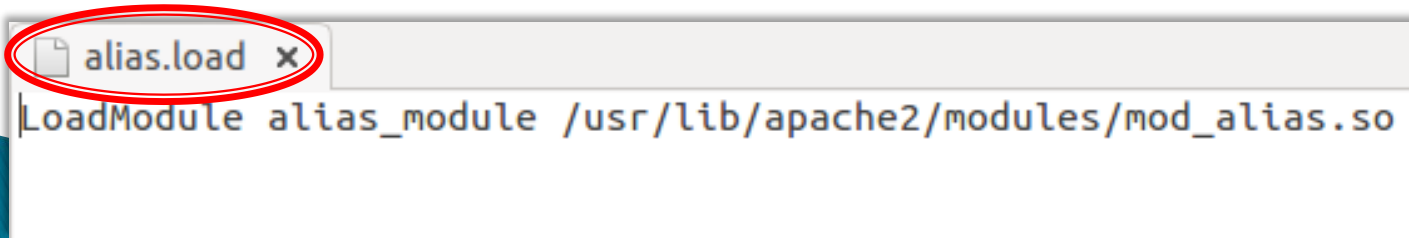


# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

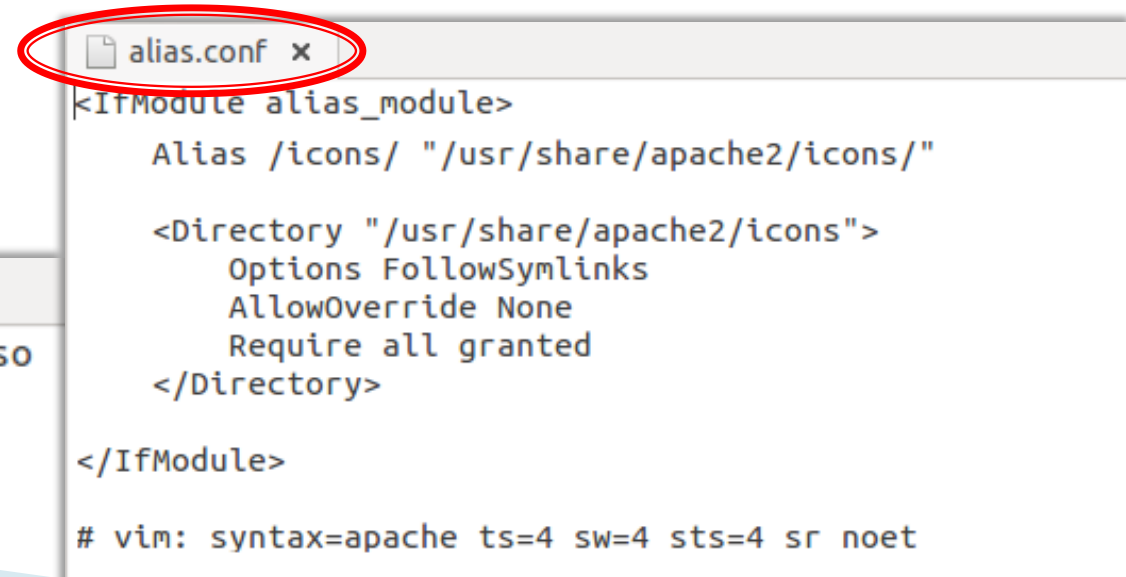
## INSTALACIÓN

En nuestra instalación se organiza en diferentes archivos localizados dentro de `/etc/apache2/mods-available`

- ▶ Archivo **XXXX.load** contiene la instrucción para cargar al módulo.
- ▶ Archivo **XXXX.conf** contiene la configuración para dicho módulo.



```
alias.load x
LoadModule alias_module /usr/lib/apache2/modules/mod_alias.so
```



```
alias.conf x
<IfModule alias_module>
    Alias /icons/ "/usr/share/apache2/icons/"

    <Directory "/usr/share/apache2/icons">
        Options FollowSymlinks
        AllowOverride None
        Require all granted
    </Directory>
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## INSTALACIÓN

Así, el directorio `/etc/apache2/mods-available` contiene todos los módulos incluidos en esta versión.

Por otro lado, `/etc/apache2/mods-enabled` determina los módulos que están activos en el servidor mediante un enlace simbólico a los módulos del directorio anterior.

- ▶ Para **activar** un módulo:

**\$ sudo a2enmod spelling**

Informa de lo que sucede y de que hay que reiniciar el servidor.

- ▶ Para **desactivar** un módulo:

**\$ sudo a2dismod spelling**

Informa de lo que sucede y de que hay que reiniciar el servidor.

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## USO Y CONFIGURACIÓN

Cada módulo en Apache2 tiene un uso y una configuración diferentes por lo que se recomienda acudir a la [documentación oficial de módulos](#).

Siguiendo con `mod_speling` (modulo que corrige errores en las URLs)

- Directiva **CheckSpelling** activa/desactiva el módulo
- Directiva **CheckCaseOnly** hace que solo se corrijan los errores de mayúsculas/minúsculas.

### CheckSpelling Directive

<b>Description:</b>	Enables the spelling module
<b>Syntax:</b>	CheckSpelling on off
<b>Default:</b>	CheckSpelling Off
<b>Context:</b>	server config, virtual host, directory, .htaccess
<b>Override:</b>	Options
<b>Status:</b>	Extension
<b>Module:</b>	mod_speling
<b>Compatibility:</b>	CheckSpelling was available as a separately available module for Apache 1.1, but was limited to miscapitalizations. As of Apache 1.3, it is part of the Apache distribution. Prior to Apache 1.3.2, the CheckSpelling directive was only available in the "server" and "virtual host" contexts.

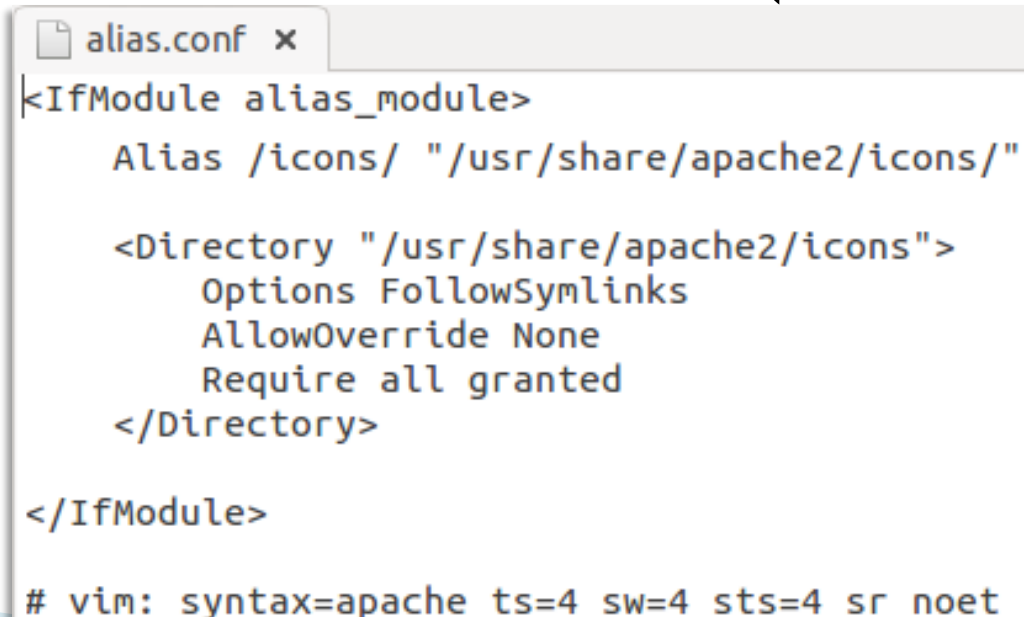
### CheckCaseOnly Directive

<b>Description:</b>	Limits the action of the speling module to case corrections
<b>Syntax:</b>	CheckCaseOnly on off
<b>Default:</b>	CheckCaseOnly Off
<b>Context:</b>	server config, virtual host, directory, .htaccess
<b>Override:</b>	Options
<b>Status:</b>	Extension
<b>Module:</b>	mod_speling

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## USO Y CONFIGURACIÓN

Los archivos de configuración asociados a cada módulo (**XXXX.conf**) incluyen directivas de configuración que únicamente se aplicarán si se carga el módulo determinado (mediante la directiva **ifModule**).



```
alias.conf x
<IfModule alias_module>
    Alias /icons/ "/usr/share/apache2/icons/"

    <Directory "/usr/share/apache2/icons">
        Options FollowSymLinks
        AllowOverride None
        Require all granted
    </Directory>

</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## mod\_status Y mod\_info

Estos módulos permiten obtener información útil sobre el servidor.

**mod\_status** está activado por defecto. Se puede ver y modificar la configuración en el directorio `/etc/apache2/mods-available/`.

**\$ sudo gedit status.conf**

- Dentro de las directivas que contiene, nos fijamos en la directiva **Require** que determina desde dónde podemos consultar el estado de la máquina.

```
<Location /server-status>
    SetHandler server-status
    Require local
    #Require ip 192.0.2.0/24
</Location>
```

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

**mod\_status** Y **mod\_info**

**mod\_info** no está activado por defecto.

**\$ sudo a2enmod info** (para activarlo)

Se puede ver y modificar la configuración en  
`/etc/apache2/mods-available/`

**\$ sudo gedit info.conf**

- Consulta esta información desde la dirección de la máquina que hayamos configurado en la directiva **Require** anterior

<http://192.0.2.0/server-info>

```
<Location /server-status>  
    SetHandler server-status  
    Require local  
    #Require ip 192.0.2.0/24  
</Location>
```



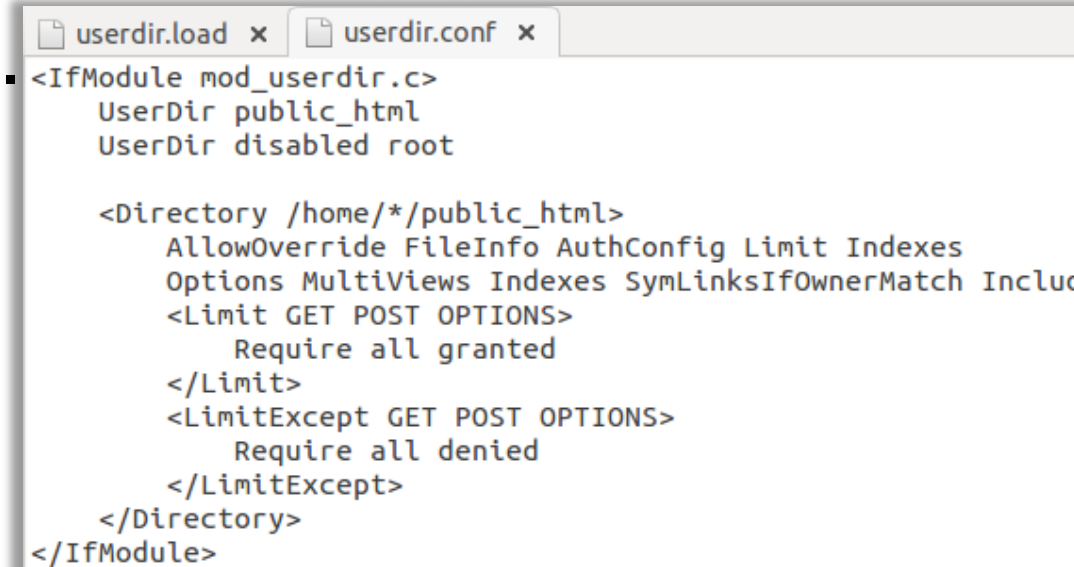
# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## DIRECTORIOS PERSONALES DE USUARIOS

Puede ser interesante que cada usuario tenga un espacio propio para almacenar sus propias páginas web. Esto puede cargar de trabajo al administrador por lo que Apache da una alternativa automatizada mediante `mod_userdir`.

Si lo activamos, cada usuario tendrá el espacio al que accederá mediante

<http://sitioejemplo.com/~nombreusuario>

A screenshot of a text editor window showing the configuration for the mod\_userdir module in an Apache configuration file. The window has two tabs: 'userdir.load' and 'userdir.conf'. The 'userdir.conf' tab is active, displaying the following configuration code:

```
<IfModule mod_userdir.c>
    UserDir public_html
    UserDir disabled root

    <Directory /home/*/public_html>
        AllowOverride FileInfo AuthConfig Limit Indexes
        Options MultiViews Indexes SymLinksIfOwnerMatch Include
        <Limit GET POST OPTIONS>
            Require all granted
        </Limit>
        <LimitExcept GET POST OPTIONS>
            Require all denied
        </LimitExcept>
    </Directory>
</IfModule>
```



# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## DIRECTORIOS PERSONALES DE USUARIOS

### ► UserDir:

- Para que cada usuario tenga su propia carpeta se utiliza la directiva **UserDir public\_html**. Crea en el servidor rutas para cada usuario (p.e. para un usuario identificado como Sergio se tendría la ruta *www.servidordeprueba.es/~Sergio*) y cada usuario tendrá en su carpeta personal un subdirectorio *public\_html* para publicar lo que quiera.

**UserDir disabled** Los usuarios no tienen su propio espacio

- Cuando se habilitan los directorios de usuario se recomienda deshabilitar el del root

**UserDir disabled root**

# 3. INSTALACIÓN, CONFIGURACIÓN Y USO DE LOS MÓDULOS DE APACHE

## MÓDULO MIME

Los tipo MIME sirven para identificar los tipos de archivos.

- ▶ **TypesConf** – Indica dónde estará el archivo que describe los tipo MIME. (mime.types)

**TypesConfig** /etc/mime.types

- ▶ **DefaultType** – Establece el tipo mime para todos aquellos archivos a los que no se les pueda asignar uno mediante su extensión. módulos que están activos

**DefaultType** text/plain

Podríamos especificar más, si el grueso de archivos del servidor son páginas html o xml , pondríamos `text/html` o si el tráfico de archivos es de tipo binario (fotografías, programas,...), pondríamos `application/octet-stream`

- ▶ **AddEncoding** – especifica un tipo concreto de codificación para determinadas extensiones de archivos

**AddEncoding** x-compress .Z

**AddEncoding** x-gzip .gz

- ▶ **AddLanguage** – Permite asociar extensiones a idiomas determinados para el contenido

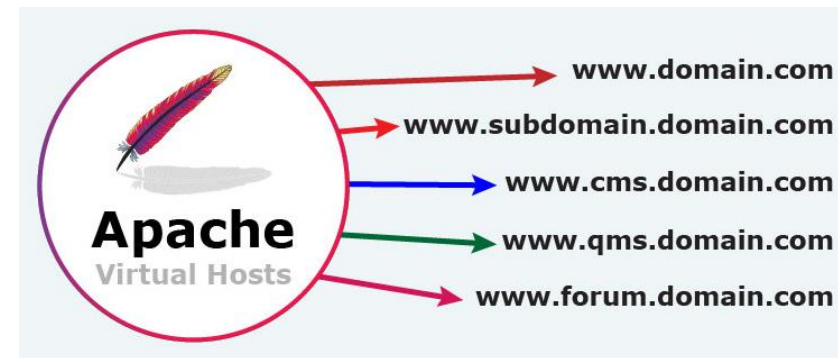
# 4. HOST VIRTUALES

## HOST VIRTUALES

- ▶ Permiten servir varios sitios web en un mismo servidor.
- ▶ El cliente no diferencia si son sitios web diferentes (de diferentes servidores) o son sitios servidos por la misma máquina (el mismo servidor).
- ▶ Se pueden gestionar varios dominios DNS en el mismo Apache, uno será el sitio principal y el resto serán host virtuales.

## Ventajas:

- ▶ Aprovechar el hardware existente
- ▶ Aprovechar las IP públicas
- ▶ Permite heredar la configuración del sitio principal, así solo se modifican las directivas que cambien



El dominio principal sería *www.domain.com* y los otros cuatro serían host virtuales.

# 4. HOST VIRTUALES

## HOST VIRTUALES

- ▶ Los host virtuales se alojan en dos directorios:
  - En `/etc/apache2/sites-available` los host **disponibles**
  - En `/etc/apache2/sites-enabled` los host **activos** (son enlaces simbólicos)

Nombre	Tamaño	Tipo	Modificado
conf-available	5 elementos	Carpeta	oct 22
conf-enabled	5 elementos	Carpeta	oct 22
mods-available	128 elementos	Carpeta	oct 22
mods-enabled	27 elementos	Carpeta	nov 16
sites-available	2 elementos	Carpeta	oct 22
sites-enabled	1 elemento	Carpeta	oct 22
apache2.conf	7,1 kB	Texto	7 ene 2014
apache2.conf.1	7,1 kB	Texto	nov 19
apache2.conf.old	7,1 kB	Desconocido	nov 3
000-vhosts.conf	1,9 kB	Texto	7 ene 2014

etc apache2 sites-available

Nombre	Tamaño	Tipo	Modificado
000-default.conf	1,3 kB	Texto	7 ene 2014
default-ssl.conf	6,4 kB	Texto	7 ene 2014

etc apache2 sites-enabled						
Nombre		Tamaño	Tipo	Modificado		
	000-default.conf	1,3 kB	Enlace hacia Texto	7 ene 2014		

# 4. HOST VIRTUALES

## SITIO POR DEFECTO

- Configuración del servidor virtual por defecto:

```
$ gedit /etc/apache2/sites-available/000-  
default.conf
```

- En el archivo de configuración principal, *apache2.conf*, se configura el directorio padre:

```
<Directory /var/www/>  
    Options Indexes FollowSymLinks  
    AllowOverride None  
    Require all granted  
</Directory>
```

Esta configuración se hereda en todos los subdirectorios salvo que se especifique una configuración concreta para el subdirectorio, en particular se hereda en `/var/www/html/`

# 4. HOST VIRTUALES

```
<Directory /var/www/>
```

```
Options Indexes FollowSymLinks
```

```
AllowOverride None
```

```
Require all granted
```

```
</Directory>
```

**Options** → directiva que controla las funcionalidades del servidor que están disponibles en un directorio particular.

- **Indexes** → Si se solicita una URL de un directorio y no hay *DirectoryIndex* (por ejemplo, index.html) en ese directorio, entonces mod\_autoindex devolverá una lista formateada del directorio. Con **-Indexes** se deshabilita el listado de contenidos del servidor
- **FollowSymLinks** → Indica que el servidor puede seguir enlaces simbólicos dentro de ese directorio

Las posibles directivas de los ficheros .htaccess no anularán lo estipulado por esta configuración.

**Require** → Comprueba si un usuario autenticado está autorizado por la autoridad del sitio.  
En este caso, “se permite el acceso incondicionalmente”.



# 4. HOST VIRTUALES

## MODIFICANDO MENSAJES DE ERROR

- ▶ Apache se puede configurar para que en caso de un error haga una de estas cuatro cosas:
  1. salida de un mensaje sencillo del error codificado (**defecto**)
  2. salida de un mensaje personalizado
  3. redirigir internamente a una ruta o URL local para gestionar el problema/error
  4. redirigir a una URL externa para manejar el problema/error
- ▶ La directiva **ErrorDocument** permite establecer mensajes personalizados de error para diferentes situaciones identificadas por códigos de error.
- ▶ Esta directiva se usa tantas veces como sea necesario.
- ▶ Se establece de forma independiente para cada sitio virtual.
- ▶ Los códigos de error los puedes consultar en

<http://www.askapache.com/htaccess/apache-status-code-headers-errordocument.html>

```
<Directory "/web/docs">  
    ErrorDocument 404 por defecto  
</Directory>
```



## 4. HOST VIRTUALES

### ALIAS A OTROS DIRECTORIOS

- Dentro de la estructura “real” de nuestro sitio (la del directorio raíz del sitio) se pueden incluir otros directorios del servidor mediante la definición de un alias (similar a la definición de enlaces simbólicos a otras carpetas o archivos) con la directiva **Alias**.
- Si creas un alias a un directorio exterior al raíz del sitio web, debes definir los permisos de acceso a ese directorio.

```
Alias "/image" "/ftp/pub/image"  
<Directory "/ftp/pub/image">  
    Require all granted  
</Directory>
```

# 4. HOST VIRTUALES

## REDIRECCIONES

- ▶ Permite redirigir las llamadas a una dirección web para que se procesen en otro punto.
  - Si se ha cambiado la estructura del sitio web y se han reubicado ciertas páginas.
  - Si se ha dividido el sitio para facilitar la gestión o mantenimiento.
- ▶ La directiva **Redirect** permite asociar una dirección absoluta o relativa a otra.

```
# Redirecciona hacia una URL de un host diferente
Redirect "/service" http://foo2.example.com/service
# Redirecciona hacia una URL del mismo host
Redirect "/one" "/two"
```

# 4. HOST VIRTUALES

## CREACIÓN DE HOST VIRTUALES

► Hay tres formas diferentes de crear host virtuales:

1. Basados en **nombres**

- Se configura para que múltiples dominios DNS apunten a una máquina Apache.
- Es el más habitual y es el **único que veremos**.

2. Basados en **IP**:

- Requiere configurar las direcciones IP de cada sitio en Apache.
- El servidor físico tendrá varias direcciones IP, una para cada sitio.
- Basados en **puertos** (es una extensión de las anteriores):
- Cada sitio se atiende en la misma IP o nombre pero en distintos puertos

3. **Varios servidores** principales :

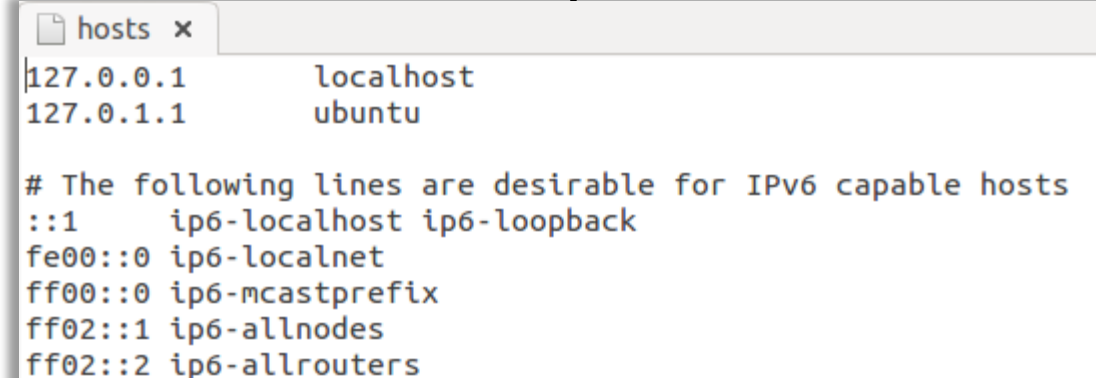
- Se mantienen varias configuraciones en el servidor.
- Solo es recomendable cuando cada sitio requiere de una configuración diferente al otro.

# 4. HOST VIRTUALES

## CREACIÓN DE HOST VIRTUALES

- ▶ Las directivas para crear los host virtuales están en el módulo **core**.
- ▶ Todos los nombres de dominio asociados a todos los host virtuales **deben** apuntar a la dirección o direcciones IP del servidor web.
  - La empresa con la que se contrata el dominio se encarga de hacerlo
  - Si publicamos nosotros el sitio, debemos configurar un servidor DNS.
- ▶ Otra opción para probarlo es editar los archivos host y añadir los alias necesarios:

**\$ sudo gedit /etc/host**



A screenshot of a text editor window titled 'hosts x' showing the contents of the /etc/hosts file. The file contains two entries for localhost: 127.0.0.1 and 127.0.1.1, both pointing to 'localhost' and 'ubuntu' respectively. Below these, there is a comment and several lines for IPv6 addresses and their corresponding hostnames.

```
hosts x
127.0.0.1      localhost
127.0.1.1      ubuntu

# 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
```

# 4. HOST VIRTUALES

## CREACIÓN DE HOST VIRTUALES

1. Creamos los **registros DNS** para que el nombre de dominio apunte a nuestra máquina o los añadimos en el fichero host.
2. Creamos un **directorio** para cada sitio de forma que los archivos estén separados. Crear, también, un archivo index.html en cada uno para que se cargue al consultar el sitio.

```
$ sudo mkdir /var/www/ejemplo.es
```

Así accederemos a las páginas mediante <http://localhost/ejemplo.es> pero queremos hacerlo escribiendo <http://ejemplo.es>

3. Escribimos los datos del host virtual en la configuración de Apache, es decir, creamos un host virtual en `/etc/apache2/sites-available`.

```
$ sudo gedit /etc/apache2/sites-available/ejemplo.es
```

# 4. HOST VIRTUALES

## CREACIÓN DE HOST VIRTUALES

```
$ sudo gedit /etc/apache2/sites-available/ejemplo.es
```

Se pueden incluir:

- Directivas de directorios en el caso de tener diferentes configuraciones.
- Configuraciones de registro de errores independientes para cada sitio.

```
ErrorLog ${APACHE_LOG_DIR}/error-ej.log
```

```
CustomLog ${APACHE_LOG_DIR}/access-ej.log combined
```

```
<VirtualHost 10.0.2.15:80>
```

```
    ServerName ejemplo.es
```

```
    ServerAlias www.ejemplo.es
```

```
    ServerAdmin alguien@ejemplo.es
```

```
    DocumentRoot /var/www/ejemplo.es
```

```
    #
```

```
    # Aquí pueden ir otras directivas
```

```
    # Por defecto hereda las del archivo principal
```

```
    #
```

```
</VirtualHost>
```

# 4. HOST VIRTUALES

## CREACIÓN DE HOST VIRTUALES

4. Activamos el nuevo host.

```
$ sudo a2ensite ejemplo.es
```

5. Recargamos los sitios de Apache

```
$ service apache2 reload
```

◦ Si da error:

```
Reloading web server config apache2
apache2: Could not reliably determine the server's fully qualified
domain name, using 127.0.1.1 for ServerName
```

- Accedemos al archivo httpd.conf y añadimos:  
o el nombre completo del servidor.

```
ServerName LocalHost
```

Ya podemos acceder mediante  
<http://ejemplo.es>

6. Desactivar el sitio virtual

```
$ sudo a2dissite default
```



# 5. CONTROL DE ACCESO

- ▶ El control de acceso forma parte de la **seguridad del sistema** y está íntimamente relacionado con la autorización y la autenticación.
- ▶ Se basa en el filtrado de acceso a determinados recursos del sistema.
- ▶ En Apache se utilizan los métodos: `mod_auth_core` y `mod_authz_host` para realizar esta función.
- ▶ Uno de los métodos para gestionar el control de acceso es el *método basado en dirección* utiliza el módulo `mod_authz_host` y las direcciones de las máquinas que quieran acceder. (activado por defecto)
- ▶ La directiva Require se utiliza para la autorización
  - Saber qué usuarios tienen permitido el acceso y cuáles no
  - Permitir o denegar el acceso (`granted` o `denied` respectivamente)

# 5. CONTROL DE ACCESO

## EJEMPLOS DE FILTROS

- ▶ Filtra una máquina concreta:

```
Require ip 10.1.2.3
Require ip 192.168.1.104 192.168.1.205
```

- ▶ Filtra un rango de direcciones:

```
Require ip 10.1
Require ip 10 172.20 192.168.2
```

```
Require ip 10.1.0.0/255.255.0.0
```

- ▶ Filtro con una máscara genérica:

```
Require ip 10.1.0.0/16
```

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

## AUTENTICACIÓN

- Consiste en comprobar que uno es quien dice ser

## AUTORIZACIÓN

- Consiste en comprobar que alguien tiene permiso para acceder a un lugar o recurso.
- ▶ El proceso de autorización suele implicar la autenticación
  - Solicita la autenticación mediante un **mensaje 401** de http que no va encriptado. La respuesta, con usuario y contraseña, tampoco.
  - Posteriormente se utilizará un protocolo seguro: **https**
  - **mod\_authn\_core** centraliza la tarea de autenticación
  - **mod\_authz\_core** es el núcleo de la gestión de autorización

## 6. AUTENTICACIÓN Y AUTORIZACIÓN

Protección mediante contraseña de un directorio del servidor:

- ▶ Crear un directorio

- si no está en nuestro sitio tendremos que crear un alias

```
mkdir /var/www/privado
```

```
mkdir /var/secreto
```

- ▶ Configurar el sitio

- se añade al sitio en el que se va a gestionar el control de acceso.

```
gedit /etc/apache2/sites-available/000-default.conf
```

## 6. AUTENTICACIÓN Y AUTORIZACIÓN

- se configura el directorio (en este caso en 000-default.conf)

Mensaje para el usuario

Tipo de autenticación

Archivo donde se guardan  
las contraseñas

Requisito, acceso con  
usuario válido

```
<Directory "/var/www/privado">  
  AuthName "Acceso privado: Introduzca  
    su usuario y contraseña"  
  AuthType Basic  
  AuthUserFile /var/secreto/.miembros  
  Require valid-user  
</Directory>
```

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

- Crear el archivo de contraseñas
  - Si no está instalado el comando htpasswd, se instala  
`sudo apt-get install apache2-utils`

```
htpasswd -c /var/secreto/.miembros pepe
```

Opción para crear el archivo solo la primera vez que se añade un miembro

Ruta especificada en el archivo de configuración

Nombre de usuario que queremos crear

Pide que se defina una contraseña y se confirme.

Acuérdate de **reiniciar** apache



# 6. AUTENTICACIÓN Y AUTORIZACIÓN

## Ficheros **.htaccess**

- ▶ Permitir el uso de los ficheros htaccess (desde el servidor o sitio virtual)
  - Cambiar la directiva `AllowOverride None` a **`AllowOverride AuthConfig`**
  - Si se hace para el sitio virtual de defecto:

```
<Directory "/var/www/">  
    Options Indexes FollowSymLinks MultiViews  
    AllowOverride AuthConfig  
    Require all granted  
</Directory>
```

Podemos modificar las directivas de autorización mediante un fichero **.htaccess**

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

## Ficheros **.htaccess** (cont)

- ▶ Podemos crear directorios y configurar su control de acceso en cada uno de ellos

```
mkdir /var/www/ficheros/  
cd /var/www/ficheros
```

- ▶ Crear dentro un fichero **.htaccess** con este contenido:

```
AuthName "Seccion Privada: Prueba de .htaccess"  
AuthType Basic  
AuthUserFile /var/secreto/.miembros  
Require valid-user
```

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

## Ficheros **.htaccess** (cont)

- Hemos sacado la configuración del *archivo de configuración* de apache
- No requiere el reinicio del servidor
- Los usuarios y contraseñas se crean como antes
- Se mejora la seguridad definiendo permisos de acceso a estos ficheros .htaccess (el usuario de apache debe tener acceso)

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

Ejemplos:

1. Permitir el acceso solo a algunos usuarios.  
`require user pepe`
2. Permitir el acceso a un grupo de usuarios

```
<Directory "/var/www/ventas">  
    AuthName "Seccion Privada: Introduzca su usuario y contraseña."  
    AuthType Basic  
    AuthUserFile /var/secreto/.miembros-ventas  
    Require valid-user  
</Directory>
```

```
<Directory "/var/www/finanzas">  
    AuthName "Seccion Privada: Introduzca su usuario y contraseña."  
    AuthType Basic  
    AuthUserFile /var/secreto/.miembros-finanzas  
    Require valid-user  
</Directory>
```

Los usuarios están en diferentes ficheros de autenticación.

# 6. AUTENTICACIÓN Y AUTORIZACIÓN

Ejemplos:

## 3. Permitir el acceso a un grupo de usuarios

```
<Directory "/var/www/ventas">  
  AuthName "Seccion Privada: Introduzca su usuario y contraseña."  
  AuthType Basic  
  AuthUserFile /var/secreto/.miembros  
  AuthGroupFile /var/secreto/.grupos  
  Require group ventas  
</Directory>
```

```
<Directory "/var/www/finanzas">  
  AuthName "Seccion Privada: Introduzca su usuario y contraseña."  
  AuthType Basic  
  AuthUserFile /var/secreto/.miembros  
  AuthGroupFile /var/secreto/.grupos  
  Require group finanzas  
</Directory>
```

Los usuarios están en el mismo fichero de autenticación pero hay otro fichero para la autorización, en el que se les asigna grupos.

El archivo .grupos tendrá el contenido:  
ventas: pferrer  
finanzas: ezapata

# 7. El protocolo https

## HTTPS –HyperText Tranfer Protocol Secure

- ▶ Añade seguridad a http al introducir una nueva capa de protocolo entre la capa de transporte y la capa de aplicación, el protocolo **SSL** –Secure Socket Layer– o el **TLS** –Transfer Layer Secure–.
- ▶ Proporciona autenticación de los usuarios y encriptación de la información transmitida.
- ▶ Un sitio HTTPS debería tener todos sus contenidos protegidos por este protocolo para evitar ataques o robos de información a través de partes inseguras
- ▶ HTTPS es un poco menos eficiente que HTTP por lo que según el volumen de información que se sirva el servicio web puede bajar el rendimiento.
- ▶ HTTPS utiliza el **puerto 443** por defecto, por lo que si se utiliza no se necesita especificar en la barra de navegación.

## 8. https en Apache

- ▶ Apache utiliza OpenSSL que implementa TLS
- ▶ Necesita activar el módulo `mod_ssl`

```
sudo a2enmod ssl  
service apache2 restart
```

- ▶ En el archivo `ports.conf`

```
gedit /etc/apache2/ports.conf
```



- ▶ Al activar el módulo `mod_ssl` se añade la orden:

```
NameVirtualHost *:80  
Listen 80  
  
<ifModule mod_ssl.c>  
    Listen 443  
</IfModule>
```

Ahora el servidor escucha tanto en el puerto 80 (http) como en el 443 (https)

```
Listen 80  
  
<ifModule ssl_module>  
    Listen 443  
</IfModule>  
  
<ifModule mod_gnutls.c>  
    Listen 443  
</IfModule>
```



## 8. https en Apache

- ▶ **SITIO SEGURO** por defecto.
  - Es el otro sitio por defecto de la distribución de Apache.
  - `Default-ssl` sitio configurado con escucha por conexión segura

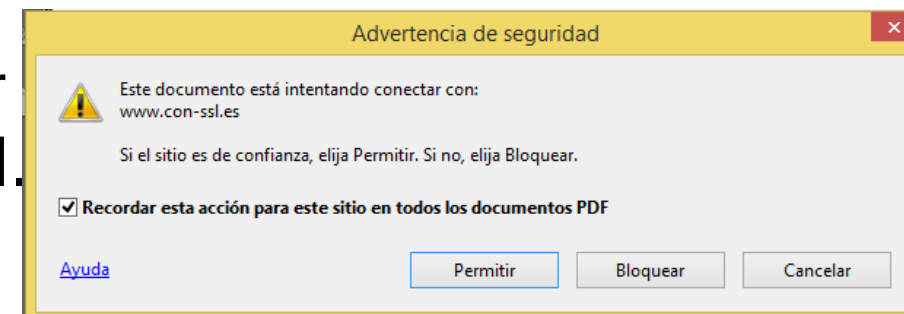
```
sudo a2ensite default-ssl  
service apache2 reload
```

Ahora podemos conectarnos de forma segura tanto al servidor como al sitio seguro por defecto.

- La configuración del sitio con ssl es

```
SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem  
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

- Apache crea un certificado autofirmado para el sitio por defecto, así, al acceder por https mostrará una excepción de seguridad.



## 8. https en Apache

- ▶ Si habilitamos un sitio con https no se debería tener activo el equivalente http. (como ocurre con los dos por defecto)
  - Deshabilitamos el default:

```
sudo a2dissite default
service apache2 reload
```

Ahora **no** podemos conectarnos http al sitio por defecto

- ▶ **CREACION DE UN SITIO SEGURO**
  - Revertimos la situación, para crear un sitio con http y otro con https:

```
sudo a2dissite default-ssl
sudo a2ensite default
service apache2 reload
```

- Para poder usar SSL en Apache debemos tener un certificado.

## 8. https en Apache

### ► CREACIÓN DE UN SITIO SEGURO (cont.)

#### 1. Adquirir un certificado de una CA\* o crear uno autofirmado.

- Configurar el DNS con un nombre de dominio y configurarlo como un host virtual por nombre → `www.con-ssl.es`
- Generar una clave privada

```
openssl genrsa -out clavepru.key 2048
```

Longitud de la clave

- Generar una clave privada con contraseña (puede ser latoso)

```
openssl genrsa -des3 -out clavepru.key 2048
```

- Generar una solicitud de certificado

```
openssl req -new -key clavepru.key -out petitionpru.csr
```

Solicita información de la empresa

(\*)Autoridad certificadora

## 8. https en Apache

### ► CREACIÓN DE UN SITIO SEGURO (cont.)

#### 2. Crear certificado autofirmado

- Lo creamos con formato X.509 con validez de un año:

```
openssl x509 -req -days 365 -in petitionpru.csr -signkey  
clavepru.key -out certificadopru.crt
```

Nos dará el visto bueno

- Colocar los ficheros creados en su destino adecuado

```
sudo mv clavepru.key /etc/ssl/private  
sudo mv certificadopru.crt /etc/ssl/certs
```

- Crear un directorio para el contenido del sitio seguro

```
mkdir /var/www/con-ssl/
```

# 8. https en Apache

## ► CREACIÓN DE UN SITIO SEGURO (cont.)

### 3. Crear el sitio virtual por nombre

```
<IfModule mod_ssl.c>
    NameVirtualHost 192.168.x.y:443
    <VirtualHost 192.168.x.y:443>
        ServerName con-ssl.es
        ServerAlias www.con-ssl.es
        ServerAdmin alguien@con-ssl.es
        DocumentRoot /var/www/con-ssl
        <Directory /var/www/con-ssl>
            DirectoryIndex index.html
            options -Indexes
            AllowOverride None
            Require all granted
        </Directory>
```

## 8. https en Apache

```
ErrorLog ${APACHE_LOG_DIR}/error_con_ssl.log
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/con_ssl_access.log combined
#-----Parte de SSL-----
SSLEngine on
SSLCertificateFile /etc/ssl/certs/certificadopru.crt
SSLCertificateKeyFile /etc/ssl/private/clavepru.key
#Para compatibilizar con ciertas versiones de Microsoft IE
BrowserMatch "MSIE [2-6]" \ nokeepalive ssl-unclean-shutdown \
    downgrade-1.0 force-response-1.0
BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown
</VirtualHost>
</IfModule>
```

- ▶ Habilitamos el sitio y recargamos Apache

```
sudo a2ensite con-ssl.es
service apache2 reload
```