



The Apache Software Foundation

Community-led development since 1999.

SERVIDOR WEB APACHE

El **servidor Apache** es un servidor web de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que alguien quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo. Además Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. *Patchy server* ("un servidor "parcheado").

El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la **Apache Software Foundation** dentro del proyecto HTTP Server (httpd).

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Las directivas que se pongan en los ficheros principales de configuración se aplicarán a todo el servidor. Si se quiere cambiar únicamente la configuración de una parte del servidor, puede cambiar el rango de acción de las directivas poniéndolas dentro de las secciones <Directory>, <DirectoryMatch>, <Files>, <FilesMatch>, <Location>, y <LocationMatch> . Estas secciones limitan el dominio de aplicación de las directivas dentro de ellas, a locales particulares dentro del sistema de ficheros o URL's particulares. Estas secciones pueden ser anidadas, para permitir un grado de selección más concreto.

Apache tiene la capacidad de servir varios sitios web diferentes al mismo tiempo. Esto se llama Hospedaje Virtual (Virtual Hosting). El dominio de aplicación de las directivas también puede ser delimitado poniéndolas dentro de <VirtualHost>, de manera que solo tendrán efecto para peticiones de un sitio web en particular.

Las directivas de configuración se agrupan en tres secciones básicas:

Directivas que controlan la forma de operación del servidor Apache como un todo, es decir, el **entorno global**.

Directivas que definen los parámetros del **servidor principal o predeterminado** que responde a las peticiones que no gestiona ningún 'virtual host'. Entendemos por 'virtual host' un subservidor con unas características propias. Estas directivas también se aplican a los servidores virtuales como valores predeterminados si éstos no especifican un valor propio específico del host virtual.

Directivas de configuración para host virtuales, que permiten peticiones Web a distintas direcciones IP o nombres de host y que se gestionan por el mismo proceso servidor Apache.

Configuración descentralizada

Apache permite una administración descentralizada de la configuración, a través de ficheros colocados dentro del árbol de páginas web. Los ficheros especiales se llaman normalmente **.htaccess**, pero se puede especificar cualquier otro nombre en la directiva `AccessFileName`. Las directivas que se pongan dentro de los ficheros **.htaccess** se aplicarán únicamente al directorio donde esté el fichero, y a todos sus subdirectorios. Los ficheros **.htaccess** siguen las mismas reglas de sintaxis que los ficheros principales de configuración. Como los ficheros **.htaccess** se leen cada vez que hay una petición de páginas, los cambios en estos ficheros comienzan a actuar inmediatamente.

Entorno global

Las directivas de esta sección afectan al comportamiento general de Apache, como el número de peticiones concurrentes que puede atender o dónde se ubican los ficheros de configuración.

Servidor principal

Las directivas de esta sección definen los valores del servidor web principal, el que responde a cualquier petición que no sea atendida por ningún host virtual. Estas directivas también se aplican a los servidores virtuales como valores predeterminados si estos no especifican un valor propio específico del host virtual.

Algunas de estas directivas dependen de módulos DSO (Objetos Dinámicos Compartidos) que existen de forma independiente del archivo binario [httpd/apache2](http://httpd.apache2). Los módulos que se deseen usar como objetos dinámicos compartidos pueden compilarse al mismo tiempo que el servidor, o pueden compilarse en otro momento y ser añadidos después usando la Herramienta de Extensión de Apache ([apxs](#) => *APache eXtenSion*)

Desde 1996, Apache, es el servidor HTTP más usado. Tuvo un papel fundamental en el desarrollo de la World Wide Web y alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años. (Estadísticas históricas y de uso diario proporcionadas por Netcraft).

1.- Ventajas:

- Modular
- Servidor Web potente, flexible y ajustado al HTTP/1.1
- Ofrece su propio API (Application Programming Interface).
- Servidor "Open Source" Código abierto

- Multi-plataforma (Windows, Linux, Unix, MAC)
- Altamente configurable y extensible.
- Popular (fácil conseguir ayuda/soporte)

2.- Tipos de instalación:

- Compilar código fuente.
- Utilizar paquetes binarios adaptados para cada Sistema operativo (usaremos este tipo).

3.- Módulos:

La arquitectura del servidor Apache es muy modular. El servidor consta de una sección *core* y diversos módulos que aportan las funcionalidades básicas de un servidor web.

LoadModule

Esta directiva corresponde al soporte de Dynamic Shared Object (Objetos Dinámicos Compartidos). Son módulos que incorporan ciertas funcionalidades que se le incorporan al servidor Apache. Para que un módulo sea funcional tienen que estar construido como un DSO e incorporar la correspondiente directiva 'LoadModule' antes de que sea utilizado. Los módulos compilados de forma estática (comprobado con `apache2 -l/httpd -l`) no es necesario incluirlos.

Algunos de estos módulos son:

- **mod_ssl** - Comunicaciones Seguras vía **TLS**.
- **mod_rewrite** - reescritura de direcciones/URLs (generalmente utilizado para transformar páginas dinámicas como php en páginas estáticas html para así engañar a los navegantes o a los motores de búsqueda en cuanto a cómo fueron desarrolladas estas páginas). El **módulo mod_rewrite** es un módulo del servidor web Apache que por defecto está instalado en todos los servicios de alojamiento web, por ejemplo, Hostinet. Este módulo permite crear direcciones URL alternativas a las dinámicas generadas por la programación de nuestro sitio web (blog, foro, portal...), de tal modo que sean más legibles y fáciles de recordar. Para hacer que funcione deberemos incluir en el archivo *.htaccess* del directorio *public_html* del alojamiento la siguiente línea de código:

RewriteEngine on

mod_dav - Soporte del protocolo **WebDAV** (RFC 2518). **WebDAV** es un grupo de trabajo del Internet Engineering Task Force. El término significa "Creación y control de

versiones distribuidos en web" (**Web Distributed Authoring and Versioning**), y se refiere al protocolo que el grupo definió.

El objetivo de WebDAV es hacer de la WWW un medio legible y editable, en línea con la visión original de Tim Berners-Lee. Este protocolo proporciona funcionalidades para crear, cambiar y mover documentos en un servidor remoto. Esto se utiliza sobre todo para permitir la edición de los documentos que sirve un servidor web, pero puede también aplicarse a sistemas de almacenamiento generales basados en web, que pueden ser accedidos desde cualquier lugar. La mayoría de los sistemas operativos modernos proporcionan soporte para WebDAV, haciendo que los ficheros de un servidor WebDAV aparezcan como almacenados en un directorio local.

https://httpd.apache.org/docs/2.4/mod/mod_dav.html

<http://www.redeszone.net/2013/03/12/webdav-que-es-y-como-functiona-manual-para-usarlo-con-otixo/>

- **mod_deflate** - Compresión transparente con el algoritmo deflate del contenido enviado al cliente.
- **mod_auth_ldap** - Permite autenticar usuarios contra un servidor **LDAP**.
- **mod_proxy_ajp** - Conector para enlazar con el **servidor Jakarta Tomcat** de páginas dinámicas en Java (servlets y JSP).

El servidor de base puede ser extendido con la inclusión de **módulos externos** entre los cuales se encuentran:

- **mod_cband** - Control de tráfico y limitador de ancho de banda.
- **mod_perl** - Páginas dinámicas en **Perl**.
- **mod_php** - Páginas dinámicas en **PHP**.
- **mod_python** - Páginas dinámicas en **Python**.
- **mod_ruby** - Páginas dinámicas en **Ruby**.
- **mod_aspdotnet** - Páginas dinámicas en **.NET de Microsoft (Módulo retirado)**.
- **mod_security** - Filtrado a nivel de aplicación, para seguridad.

<https://httpd.apache.org/docs/2.0/es/dso.html>

4.- Uso

Apache es usado principalmente para enviar páginas web estáticas y dinámicas en la World Wide Web. Muchas aplicaciones web están diseñadas asumiendo como ambiente de implantación a Apache, o que utilizarán características propias de este servidor web.

Apache es el servidor web en la plataforma de aplicaciones XAMPP, junto a MySQL y los lenguajes de programación PHP/Perl/Python (y ahora también Ruby) para múltiples sistemas operativos.

Este servidor web es redistribuido como parte de varios paquetes propietarios de software, incluyendo la base de datos Oracle y el IBM WebSphere application server.

Mac OS X integra apache como parte de su propio servidor web y como soporte de su servidor de aplicaciones WebObjects. Apache es incluido con Novell NetWare 6.5, donde es el servidor web por defecto, y en muchas distribuciones Linux.

Apache es usado para muchas otras tareas. Un ejemplo es la compartición de archivos desde un ordenador personal hacia Internet. Un usuario que tiene Apache instalado en su escritorio puede colocar arbitrariamente archivos en la raíz de documentos de Apache, desde donde pueden ser compartidos.

Los programadores de aplicaciones web a veces utilizan una versión local de Apache con el fin de previsualizar y probar código mientras éste es desarrollado.

Microsoft Internet Information Services (IIS) es el principal competidor de Apache, así como **Sun Java System Web Server** de Sun Microsystems y un anfitrión de otras aplicaciones como Zeus Web Server. Algunos de los sitios web más grandes del mundo están ejecutándose sobre Apache. La capa frontal (front end) del motor de búsqueda Google está basada en una versión modificada de Apache, denominada Google Web Server (GWS). Muchos proyectos de Wikimedia también se ejecutan sobre servidores web Apache.

5.- Configuración

- **Servidor principal**

- Atiende las peticiones si no se configuran servidores virtuales.

- **Servidores virtuales**

- Apache soporta sitios o servidores virtuales basados en IP, nombres y puertos. Varios servidores sobre una misma instalación de Apache.

- <VirtualHost> ... </VirtualHost>

- **Linux (Debian/Ubuntu)**

- La versión precompilada (.deb) para Debian/Ubuntu tiene habilitados servidores virtuales.

- Tiene creado y habilitado un servidor virtual por defecto (default).

- **Windows**

- La versión de Windows no tiene habilitados por defecto los servidores virtuales.

- Se utiliza el servidor “principal”.

La mayor parte de la configuración se realiza en el fichero **apache2.conf** (Ubuntu) o **httpd.conf** (Otros). Cualquier cambio en este archivo requiere reiniciar el servidor, o forzar la lectura de los archivos de configuración nuevamente.

En Windows:

Modo gráfico:

- ✓ Monitor Apache o Servicios del Sistema

Línea de comandos:


- ✓ Inicio (varias posibilidades)
net start apache2.2
httpd -k start
- ✓ Parada (varias posibilidades)
net stop apache2.2
httpd -k stop
httpd -k shutdown

En Linux:

- ✓ **Inicio** (varias posibilidades)
 - sudo service apache2 start
 - sudo /etc/init.d/ apache2 start
 - sudo apachectl start
- ✓ **Parada** (varias posibilidades)
 - sudo service apache2 stop
 - sudo /etc/init.d/ apache2 stop
 - sudo apachectl stop
- ✓ Lea de nuevo los ficheros (varias posibilidades)
 - sudo service apache2 reload
 - sudo /etc/init.d/ apache2 reload

Configuración editando ficheros de texto.

5.1.- Ficheros de Configuración en Linux:

 **Directorio: /etc/apache2**

```

alumno@ubuntuServer:~$ ls -la /etc/apache2
total 88
drwxr-xr-x  8 root root  4096 sep 25 00:07 .
drwxr-xr-x 93 root root  4096 nov  9 18:38 ..
-rw-r--r--  1 root root  7219 oct 19 17:15 apache2.conf
drwxr-xr-x  2 root root  4096 sep 24 23:38 conf-available
drwxr-xr-x  2 root root  4096 sep 24 23:38 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 24 23:38 mods-available
drwxr-xr-x  2 root root  4096 sep 24 23:38 mods-enabled
-rw-r--r--  1 root root   320 ene  7 2014 ports.conf
drwxr-xr-x  2 root root  4096 sep 24 23:38 sites-available
drwxr-xr-x  2 root root  4096 sep 24 23:38 sites-enabled
alumno@ubuntuServer:~$

```

➤ /etc/apache2/apache2.conf

- Fichero de configuración principal.
- Contienen un conjunto de directivas que determinan el comportamiento del servidor.
- Incluye (directiva include) otros ficheros de configuración.

```

# Formato general apache2.conf

# Aquí empieza la Sección 1 (directivas de configuración global)

... Directivas globales

# Aquí empezaría la sección 2 (directivas de funcionamiento del
# servidor principal )
... Directivas de funcionamiento del servidor principal (se heredan en
los servidores virtuales)
User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}

include /etc/apache2/mods-enabled/*.load
include /etc/apache2/mods-enabled/*.conf
include /etc/apache2/httpd.conf
include /etc/apache2/ports.conf

.. Directivas de logs y errores
include /etc/apache2/conf.d/

# Aquí empezaría la sección 3 (Servidores virtuales)
include /etc/apache2/sites-enabled/

```

➤ /etc/apache2/ports.conf

- Incluido en apache2.conf.
- Se definen las IPs y puertos donde escucha el servidor.

➤ /etc/apache2/envvars

Define variables de entorno.

🌈 Directorios de configuraciones globales

- Configuraciones disponibles: /etc/apache2/conf-available/
 - Configuraciones habilitadas: /etc/apache2/conf-enabled/
- Enlaces simbólicos a los ficheros de conf-available.

```

alumno@ubuntuServer:/etc/apache2/conf-enabled$ ls -la
total 8
drwxr-xr-x 2 root root 4096 sep 24 23:38 .
drwxr-xr-x 8 root root 4096 sep 25 00:07 ..
lrwxrwxrwx 1 root root 30 sep 24 23:38 charset.conf -> ../conf-available/charset.conf
lrwxrwxrwx 1 root root 44 sep 24 23:38 localized-error-pages.conf -> ../conf-available/localized-error-pages.conf
lrwxrwxrwx 1 root root 46 sep 24 23:38 other-vhosts-access-log.conf -> ../conf-available/other-vhosts-access-log.conf
lrwxrwxrwx 1 root root 31 sep 24 23:38 security.conf -> ../conf-available/security.conf
lrwxrwxrwx 1 root root 36 sep 24 23:38 serve-cgi-bin.conf -> ../conf-available/serve-cgi-bin.conf
alumno@ubuntuServer:/etc/apache2/conf-enabled$

```

📁 Directorios de configuración de módulos

- **Módulos disponibles:** /etc/apache2/mods-available/
- **Módulos habilitados:** /etc/apache2/mods-enabled/

Enlaces simbólicos a los ficheros de mods-available.

```

alumno@ubuntuServer:/etc/apache2/mods-enabled$ ls -la a*
lrwxrwxrwx 1 root root 36 sep 24 23:38 access_compat.load -> ../mods-available/access_compat.load
lrwxrwxrwx 1 root root 28 sep 24 23:38 alias.conf -> ../mods-available/alias.conf
lrwxrwxrwx 1 root root 28 sep 24 23:38 alias.load -> ../mods-available/alias.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 auth_basic.load -> ../mods-available/auth_basic.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 authn_core.load -> ../mods-available/authn_core.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 authn_file.load -> ../mods-available/authn_file.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 authz_core.load -> ../mods-available/authz_core.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 authz_host.load -> ../mods-available/authz_host.load
lrwxrwxrwx 1 root root 33 sep 24 23:38 authz_user.load -> ../mods-available/authz_user.load
lrwxrwxrwx 1 root root 32 sep 24 23:38 autoindex.conf -> ../mods-available/autoindex.conf
lrwxrwxrwx 1 root root 32 sep 24 23:38 autoindex.load -> ../mods-available/autoindex.load

```

📁 Directorios de configuración de sitios (o servidores) virtuales

- **Sitios Disponibles:** /etc/apache2/sites-available/

Contienen el fichero 000-default.conf con la configuración del servidor virtual por defecto.

- **Sitios Activos:** /etc/apache2/sites-enabled/

Contienen el fichero 000-default.conf que es un enlace al fichero del mismo nombre del directorio sites-enable.

```

alumno@ubuntuServer:/etc/apache2$ ls -la sites-enabled/
total 8
drwxr-xr-x 2 root root 4096 sep 24 23:38 .
drwxr-xr-x 8 root root 4096 sep 25 00:07 ..
lrwxrwxrwx 1 root root 35 sep 24 23:38 000-default.conf -> ../sites-available/000-default.conf
alumno@ubuntuServer:/etc/apache2$

```



```
curso1617@curso1617-VirtualBox: /etc/apache2/sites-enabled
GNU nano 2.5.3 Archivo: 000-default.conf Modificado

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html

<Directory /var/www/html>
Options Indexes
</Directory>

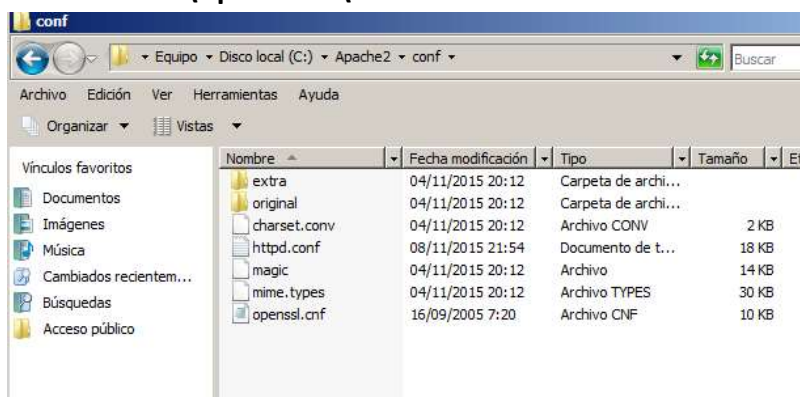
# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
```

5.2.- Ficheros de Configuración en Windows:

Directorio: C:\Apache2.2\conf



- **Fichero de configuración principal:** C:\Apache2.2\conf\httpd.conf

Contiene un conjunto de directivas que determinan el comportamiento del servidor. Puede incluir (directiva include) otros ficheros de configuración.

Las directivas no especificadas utilizan su valor por defecto.

Comentarios -> #

Muchas directivas. <http://httpd.apache.org/docs/2.4/mod>

- ✓ **Ficheros a servir por defecto:** DirectoryIndex
- ✓ **Opciones sobre directorios:** <Directory> ... </Directory>
 - Options Indexes
 - ErrorDocument
- ✓ **Directorios virtuales**
 - Directorios que están fuera del directorio raíz (DocumentRoot) del servidor.

- Alias

✓ **Redirecciones:** Redirect

6.- Algunas de las Directivas que usaremos:

➤ **ServerRoot:**

La directiva ServerRoot es el directorio raíz de la instalación, donde se almacenan los distintos ficheros que utiliza el servidor Apache. Salvo que se indiquen rutas absolutas, las rutas relativas parten de él.

➤ **ServerName**

ServerName especifica el nombre y el puerto que el servidor utiliza para identificarse. Con una correcta configuración, este valor se puede determinar automáticamente, pero es recomendable especificarlo explícitamente para evitar problemas durante el arranque.

Si no se pone un nombre DNS válido, no funcionarán las redirecciones generadas por el servidor.

Si no existe un nombre DNS registrado entonces deberemos poner la dirección IP. El acceso mediante IP siempre es posible, pero además funcionarán las redirecciones.

ServerName www.asirxx.net 80

Esta directiva tendremos que modificarla en el fichero inicial de configuración. Esta directiva también tendremos que incluirla en los VirtualHost

➤ **UseCanonicalName**

UseCanonicalName determina como Apache construye las autoreferencias de URL y las variables SERVER_NAME y SERVER_PORT.

Cuando está como "Off", Apache usará los valores suministrados por el cliente. Cuando está como "On", Apache usará la directiva ServerName. **UseCanonicalName Off**

➤ **ServerAdmin**

ServerAdmin especifica la dirección de email que el servidor incluye en cualquier mensaje de error que el servidor envía al cliente.

➤ **Timeout**

Timeout indica el número de segundos antes de que se cancele una conexión por falta de respuesta. El valor predeterminado: **Timeout 300**

➤ **KeepAlive**

KeepAlive indica si se permiten o no las conexiones persistentes, es decir más de una petición por conexión. Puede tomar los valores On u Off. El valor predeterminado es On. Por ejemplo: **KeepAlive Off**

➤ **MaxKeepAliveRequests**

MaxKeepAliveRequests indica el máximo número de peticiones que se permiten en conexiones persistentes. Un valor 0 permite un número ilimitado. Se recomienda dejar este valor elevado para obtener un mayor rendimiento. Por ejemplo: **MaxKeepAliveRequests 100**

➤ **KeepAliveTimeout**

KeepAliveTimeout indica el número de segundos de espera para la siguiente petición del mismo cliente con la misma conexión. Por ejemplo: **KeepAliveTimeout 15**

➤ **MaxClients**

MaxClients indica el máximo número de procesos hijos que se lanzan para atender conexiones simultáneas. Si llegan más peticiones que procesos estas se almacenan en una cola (listen). Por ejemplo: **MaxClients 150**

➤ **MaxRequestsPerChild**

MaxRequestsPerChild indica el número máximo de hilos por hijo. Por ejemplo: **MaxRequestsPerChild 1000**

➤ **DocumentRoot**

Indica el directorio donde se almacenan los documentos web.

Existe la posibilidad de utilizar enlaces simbólicos dentro del DocumentRoot.

DocumentRoot "/var/www/html"

Esta directiva tendremos que modificarla en el fichero inicial de configuración si queremos ubicar las páginas en un directorio distinto al predeterminado.

➤ **Listen**

Listen permite asociar Apache a una dirección y/o puerto específico además del predeterminado. Por ejemplo: **Listen 12.34.56.78:8080 Listen 80**

➤ **DirectoryIndex**

DirectoryIndex indica cómo responde el servidor cuando se solicita un directorio. Lo habitual es que haya una página que se cargue automáticamente. Por ejemplo:

DirectoryIndex index.php index.html indice.html index.html.var

Es posible aplicar esta directiva a directorios individuales para indicar índices distintos. Por ejemplo:

```
<Directory /var/www/html/varios>
DirectoryIndex pagina.html
</Directory>
```

Haría que al solicitar el directorio varios se cargara pagina.html. Evidentemente, si se pide una página concreta por su nombre es la solicitada la que se carga.

➤ **Options** Indexes All None

Options indica varias posibles opciones de comportamiento. Se pueden aplicar a un directorio concreto. El formato es: Options [+|-]option [[+|-]option] ...

donde un "+" añade la opción y un "-" quita la opción. Ciertas opciones pueden depender de algún módulo DSO.

Las opciones que tenemos son:

- **All** todas las opciones salvo MultiViews. Es el valor predeterminado
- **ExecCGI** Se permite la ejecución de scripts CGI.
- **FollowSymLinks** el servidor seguirá los enlaces simbólicos. Tener esta opción activa aumenta el rendimiento ya que el servidor no comprueba si un fichero o directorio es un enlace simbólico y es más rápido, pero en algunos casos puede presentar problemas de inseguridad.
- **Includes** se permiten incluir Server-side.
- **IncludesNOEXEC** se permiten incluir Server-side pero se deshabilitan los órdenes #exec y #exec CGI.
- **Indexes** Si una URL solicita un directorio y no existe DirectoryIndex (v.g., index.html) en ese directorio, el servidor devolverá una lista del contenido del directorio.
- **MultiViews** se permiten mostrar contenido negociado en función de diversos valores.
- **SymLinksIfOwnerMatch** Se sigue un enlace simbólico sólo si los propietarios del enlace y del destino coinciden.

➤ Include

Include inserta uno o varios ficheros de configuración. Por ejemplo

Include conf.d/*.conf

Es habitual dejar el fichero de configuración con las características globales que no se tienen que modificar en el fichero principal e incluir los ficheros que pueden estar sujetos a modificación en el directorio "/etc/apache2/conf_available".

- LoadModule
- <IfModule ... > ... </IfModule>

➤ <IfModule modulo>

Indican una serie de directivas que el servidor sólo tendrá en cuenta si el módulo indicado está activo en el servidor. Esta directiva puede contener múltiples línea y termina en un </IfModule>.

Por ejemplo:

```
<IfModule prefork.c>
directiva1
directiva3
directiva3
</IfModule>
```

➤ AccessFileName

AccessFileName especifica el nombre del fichero de configuración particular de un directorio. En este directorio se pueden incluir las directivas que queremos que se apliquen a este directorio concreto. El valor habitual es: **AccessFileName .htaccess**

➤ AllowOverride

AllowOverride controla qué directivas se pueden situar en los ficheros .htaccess.

Los valores de AllowOverride pueden ser "All", "None", o una combinación de:

- **AuthConfig**: Permitir el uso de directivas de autorización (AuthDBMGroupFile, AuthDBMUserFile, AuthGroupFile, AuthName, AuthType, AuthUserFile, Require, etc.).
- **FileInfo** Permitir el uso de directivas de control de tipo de documentos (DefaultType, ErrorDocument, ForceType, LanguagePriority, etc).

- **Indexes** Permitir el uso de directivas que controlan los índices de directorios (AddDescription, AddIcon, AddIconByEncoding, AddIconByType, DefaultIcon, DirectoryIndex, FancyIndexing, HeaderName, IndexIgnore, IndexOptions, ReadmeName, etc.).
- **Limit** Permitir el uso de directivas de acceso de hosts (Allow, Deny y Order).
- **Options** Permitir el uso los valores de la directiva Options.

Por ejemplo:

Options FileInfo AuthConfig Limit o bien **AllowOverride None**

Hay que tener en cuenta que si AllowOverride tiene un valor distinto a "None", el servidor web tendrá que buscar este fichero en todos los directorios que haya hasta llegar al fichero o directorio solicitado para aplicar la configuración. Esto puede influir en el rendimiento del servidor web. Por este motivo es aconsejable poner de forma global.

AllowOverride None

Y después activarlo en el directorio concreto que queramos que tenga una configuración específica.

Si permitimos a otros usuarios poner documentos en el servidor web deberemos controlar la directiva AllowOverride para determinar qué configuraciones permitimos que el usuario incluya en los ficheros .htaccess para evitar problemas de seguridad. Por ejemplo, la opción FollowSymLinks permite poner enlaces simbólicos en el servidor web para acceder a ficheros externos al árbol web (especificado mediante la directiva DocumentRoot. Normalmente para aumentar el rendimiento del servidor tendremos esta directiva activa para evitar que el servidor tenga que comprobar si un fichero o directorio es un enlace simbólico antes de leerlo; pero si esta directiva está activa un usuario podría poner un enlace, por ejemplo, al fichero /etc/passwd y hacerlo público a través de la web, cosa nada aconsejable.

➤ **Allow**

Allow afecta a los hosts que pueden acceder a una determinada área del servidor. El acceso se puede controlar por nombre, IP, rango de IP u otras características que se puedan almacenar en variables de entorno.

El primer argumento de esta directiva es siempre "from". Los siguientes argumentos pueden tener diferentes formas:

All se permite el acceso a todos los hosts salvo lo especificado en Deny y Order que se verá más adelante.

Para permitir el acceso a un grupo concreto de hosts podremos especificar:

- Un nombre de dominio:

Ejemplo: Allow from asirxx.net

- Una dirección IP

Ejemplo: Allow from 192.168.1.3

- Una dirección IP parcial

Ejemplo: Allow from 192.168.1

- Una dirección de red/máscara

Ejemplo: Allow from 10.12.0.0/255.255.0.0

- Una dirección de red/numero

Ejemplo: Allow from 10.12.0.0/16

Uso de una variable de entorno como por ejemplo User-Agent (navegador), Referer (procedencia del enlace), u otra cabecera de la petición HTTP.

➤ Deny

Deny afecta a los hosts que no pueden acceder a una determinada área del servidor. El resto es idéntico a Allow.

Order

La directiva Order controla el orden en que se evalúan las directivas Allow y Deny:

Order Deny,Allow

Primero se evalúa Deny. Se permite acceso a cualquier host que primero no esté indicado en Deny o que sí lo esté en Allow. ***El acceso se garantiza por defecto.***

Order Allow,Deny

Primero se evalúa Allow. Se deniega acceso a cualquier host que primero no esté indicado en Allow o que sí lo esté en Deny. ***El acceso se deniega por defecto.***

Los valores se separan por comas y sin espacios.

En el siguiente ejemplo, todos los hosts del dominio asirxxx.net tienen acceso pero ningún otro.

```
Order Deny,Allow
Deny from all
Allow from asirxx.net
```

En el siguiente ejemplo todos los hosts del dominio asirxx.net tienen acceso salvo los host que estén en el subdominio paloma.asirxx.net. El resto de hosts que no están en el dominio asirxx.net tampoco tendrían acceso.

```
Order Allow,Deny
Allow from asirxx.net
Deny from paloma.asirxx.net
```

Por otro lado, si cambiamos Order en el anterior ejemplo y lo ponemos como Deny,Allow, todos los hosts tendrían acceso. Esto sucede porque Allow se evalúa en segundo lugar y permitiría acceso a paloma.asirxx.net y el acceso por defecto es aceptar.

La presencia de una directiva Order puede afectar al acceso a una parte del servidor incluso en ausencia de directivas Allow y Deny por los efectos de los valores predeterminados. Por ejemplo:

```
<Directory /www>
  Order Allow,Deny
</Directory>
```

Denegaría todo el acceso al directorio /www ya que el estado por defecto sería Deny.

➤ Files

Files permite especificar una configuración concreta para el acceso a ficheros concretos.

➤ FilesMatch

FilesMatch permite especificar una configuración concreta para el acceso a ficheros concretos basándose en expresiones regulares. La configuración es similar al LocationMatch.

Por ejemplo:

```
<FilesMatch "^\.ht">
  Order allow,deny
  Deny from all
</FilesMatch>
```


impediría que alguien pudiera consultar desde un navegador cualquier fichero que comenzara por .ht, para evitar hacer visibles los ficheros .htaccess y .htpasswd.

Por ejemplo, si en ciertos directorios tuviéramos algunos scripts cuyo nombre termina en .sh y no quisiéramos que nadie los pudiera ver a través del servidor web pondríamos:

```
<FilesMatch "^.*\.sh">  
    Order allow,deny  
    Deny from all  
</Files>
```

➤ Alias

Alias permite albergar ficheros fuera del directorio especificado en DocumentRoot.

Alias directorio_relativo directorio_absoluto

Por ejemplo: Alias /manual "/var/www/manual"

Haría que siempre que pongamos <http://192.168.1.17/manual> el contenido que se sirva sea el de /var/www/manual independientemente de donde esté definido el DocumentRoot.

También:

```
Alias /icons/ "/var/www/icons/"  
<Directory "/var/www/icons">  
    Options Indexes MultiViews  
    AllowOverride None  
    Order allow,deny  
    Allow from all  
</Directory>
```

Con esta directiva podemos evitar tener activo FollowSymLinks y poder acceder a directorio ajenos al árbol web especificado en DocumentRoot.

➤ AliasMatch

AliasMatch es idéntico a Alias salvo que podemos especificar expresiones regulares en lugar de un valor fijo.

```
Alias /pag.*/ "/var/paginas/"  
<Directory "/var/www/icons">  
    Options -Indexes  
    DirectoryIndex indice.html
```

```
AllowOverride None
Order allow,deny
Allow from all
</Directory>
```

En este caso una petición dirigida a un directorio que empiece por pag automáticamente se dirige al directorio "/var/paginas".

➤ ErrorLog

ErrorLog indica la ubicación del fichero de registro de errores en las consultas. Es conveniente especificar un fichero de registro en cada VirtualHost con el nombre asociado a ese servidor. De esta forma podemos separar los registros de los distintos servidores que tengamos.

```
##
ErrorLog ${APACHE_LOG_DIR}/error.log
##
```

Si consultamos en el fichero de variables de entorno envvars, veremos la ruta exacta donde se ubica el fichero de errores:

```
alumno@ubuntuServer:/etc/apache2$ cat envvars | grep APACHE_LOG
export APACHE_LOG_DIR=/var/log/apache2$SUFFIX
alumno@ubuntuServer:/etc/apache2$
```

➤ LogLevel

LogLevel Controla el número de mensajes registrados en error.log.

Puede ser: debug, info, notice, warn, error, crit, alert, emerg. Se recomienda un nivel de error medio, por ejemplo, **LogLevel warn**.

```
##
# LogLevel: Control the severity of messages logged to the error_log.
# Available values: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the log level for particular modules, e.g.
# "LogLevel info ssl:warn"
##
LogLevel warn
```

➤ Redirect

Redirect permite indicar al cliente que un documento ha cambiado de dirección web.

Por ejemplo:

Redirect permanent /cursoactual <http://www.palomatica.info/moodle23>

➤ **AddIcon**

AddIcon indica al servidor qué imagen mostrar según el tipo de fichero en los listados generados automáticamente. Sólo es válida para los directorios con la opción de FancyIndexing activa, es decir en el directorio tenemos IndexOptions FancyIndexing.

➤ **DefaultIcon**

DefaultIcon es el icono que se muestra para ficheros sin icono asociado.

```
DefaultIcon /icons/unknown.gif
```

➤ **AddDescription**

AddDescription permite poner una breve descripción tras el fichero, solo cuando está activo FancyIndexing, decir en el directorio tenemos IndexOptions FancyIndexing.

El formato es: AddDescription "descripcion" tipofich

```
AddDescription "documento comprimido con GZIP" .gz
AddDescription "archivo tar" .tar
AddDescription "archivo tar comprimido" .tgz
AddDescription "archivo comprimido con ZIP" .zip
```

➤ **DefaultLanguage**

DefaultLanguage se utiliza para especificar el idioma predeterminado cuando se negocian los contenidos con el cliente. Por ejemplo, DefaultLanguage es

➤ **AddLanguage**

Añade un nuevo idioma.

➤ **AddDefaultCharset**

AddDefaultCharset especifica un conjunto de caracteres por defecto para las páginas devueltas. El valor predeterminado es iso-8859-1 (latin1).

```
AddDefaultCharset ISO-8859-1

#AddDefaultCharset UTF-8
```

➤ AddType

AddType permite añadir o modificar la configuración MIME de un tipo concreto. Por ejemplo:

```
AddType application/x-tar .tgz
```

De esta forma cuando el servidor lea un fichero de este tipo para enviarlo a un cliente previamente el envía la cabecera adecuada para que el navegador cliente sepa qué hacer con él.

➤ ErrorDocument

Permite personalizar las respuestas de los mensajes de error en tres formas distintas, un texto simple, una redirección a una página local o una redirección externa. El primer campo es el código de error correspondiente, por ejemplo 404 es el código de documento inexistente. El segundo la acción que se realiza.

El administrador Web puede de modificar estos mensajes de error para facilitar el acceso a los clientes, es decir cuando alguien solicita alguna página que no existe puede mostrar una página donde haya enlaces a los contenidos que se albergan. Un servidor bien configurado debería tener páginas de error bien configuradas para evitar que aparecieran los mensajes de error estándar.

Vemos algunos ejemplos:

```
ErrorDocument 500 "El servidor ha fallado."
```

```
ErrorDocument 404 /noencontrado.html
```

El fichero noencontrado.html se codificará dentro del DocumentRoot. También se podría crear un directorio dentro de éste y situar allí el fichero.

7.- Configuraciones de directorios

Cada directorio al que Apache tenga acceso se puede configurar para satisfacer los servicios y características que se permiten o no en dicho directorio y en sus subdirectorios.

Valor predeterminado

Primero configuramos el predeterminado con un conjunto de características muy restrictivas:

```
<Directory />
```

```
Options FollowSymLinks
AllowOverride None
</Directory>
```

Esta configuración garantiza el mayor rendimiento ya que el servidor, por un lado no tiene que comprobar si un fichero es un enlace simbólico y por otro no tiene que leer ficheros .htaccess.

Si queremos modificar estas características, lo haremos más adelante para directorios concretos.

Directorio raíz

El valor de directorio tendría que ser el mismo que indicamos en la directiva DocumentRoot.

```
<Directory "/var/www/html">
Options Indexes FollowSymLinks
AllowOverride None
Order allow,deny
Allow from all
</Directory>
```

Directorios restringidos

Podemos restringir el acceso a ciertos directorios mediante clave. Para esto tendremos que configurar el directorio adecuadamente. Por ejemplo, supongamos que para acceder al directorio /var/www/html/**privado** el navegador solicite una contraseña.

Podemos hacerlo de dos modos, configurando la autenticación en el fichero de configuración apache2.conf o indicándole a apache que busque en ese directorio un fichero .htaccess. Esto lo haremos indicando AllowOverride AuthConfig en el fichero apache2.conf:

```
<Directory /var/www/html/privado>
AllowOverride AuthConfig
</Directory>
```

En este caso hay que crear un fichero llamado .htaccess en el directorio /var/www/html/privado con el siguiente contenido:

```
AuthType Basic
AuthName "Acceso restringido"
AuthUserFile /etc/apache2/.htpasswdB
require valid-user
```

El fichero `.htpasswd` es conveniente que esté fuera del árbol de directorios accesibles por los clientes del servidor web.

Por último creamos el fichero `.htpasswd` a la vez que damos de alta un usuario:

```
htpasswd -c /etc/apache2/.htpasswd usuario
```

Podemos dar de alta más usuarios ejecutando:

```
htpasswd /etc/apache2/.htpasswd usuario2
```

En este caso sin la opción `-c` que se utiliza para crear el fichero. Con la opción `-D` se elimina un usuario.

8.- Módulos

Cada módulo tiene unas Funcionalidades.

Tipos:

- **Módulos estáticos** que se añaden cuando se compila Apache. Un módulo estático se incluye en el binario de Apache en tiempo de compilación, por lo que siempre está disponible.
- Módulos que se cargan dinámicamente cuando se inicia el servidor.

Ventajas de los **Módulos dinámicos** (Dynamic Shared Object):

- Servidor más flexible.
- Más sencillo el prototipado y desarrollo de módulos.

Desventajas DSO:

- Servidor es más lento en el arranque.
- Servidor más lento en funcionamiento.

Para ver los módulos en Linux:

```
alumno@ubuntuServer:/usr/lib/apache2/modules$ apache2ctl -M
```

Directivas:

- **LoadModule**

Permite cargar módulos dinámicos.

- **<IfModule nombre_modulo> ... </IfModule>**

Especificar directivas que se tendrán en cuenta si el módulo está cargado.

8.1.- Módulos disponibles en Linux:

◦ `/usr/lib/apache2/modules`

```
aluno@ubuntuServer:~$ ls -la /usr/lib/apache2/modules
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_access_compat.so
-rw-r--r-- 1 root root 10248 jul 24 19:27 mod_actions.so
-rw-r--r-- 1 root root 14344 jul 24 19:27 mod_alias.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_allowmethods.so
-rw-r--r-- 1 root root 10168 jul 24 19:27 mod_asis.so
-rw-r--r-- 1 root root 14352 jul 24 19:27 mod_auth_basic.so
-rw-r--r-- 1 root root 34832 jul 24 19:27 mod_auth_digest.so
-rw-r--r-- 1 root root 26640 jul 24 19:27 mod_auth_form.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authn_anon.so
-rw-r--r-- 1 root root 14352 jul 24 19:27 mod_authn_core.so
-rw-r--r-- 1 root root 14352 jul 24 19:27 mod_authn_dbd.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authn_dbm.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authn_file.so
-rw-r--r-- 1 root root 18480 jul 24 19:27 mod_authn_socache.so
-rw-r--r-- 1 root root 51248 jul 24 19:27 mod_authnz_ldap.so
-rw-r--r-- 1 root root 22544 jul 24 19:27 mod_authz_core.so
-rw-r--r-- 1 root root 14352 jul 24 19:27 mod_authz_dbd.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authz_dbm.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authz_groupfile.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authz_host.so
-rw-r--r-- 1 root root 10256 jul 24 19:27 mod_authz_owner.so
-rw-r--r-- 1 root root 6160 jul 24 19:27 mod_authz_user.so
-rw-r--r-- 1 root root 38928 jul 24 19:27 mod_autoindex.so
```

Directorios y ficheros de configuración:

Módulos disponibles: [/etc/apache2/mods-available/](#)

Ficheros **.load**

Para cargar un módulo.

Ficheros **.conf**

Configuración básica para iniciar el módulo.

Módulos habilitados: [/etc/apache2/mods-enabled/](#)

Enlaces simbólicos a los ficheros de mods-available.

Módulos a cargar al iniciar Apache.

Comandos:

➤ Habilitar un módulo: **a2enmod** nombre_modulo

Crea un enlace simbólico en mods_enabled.

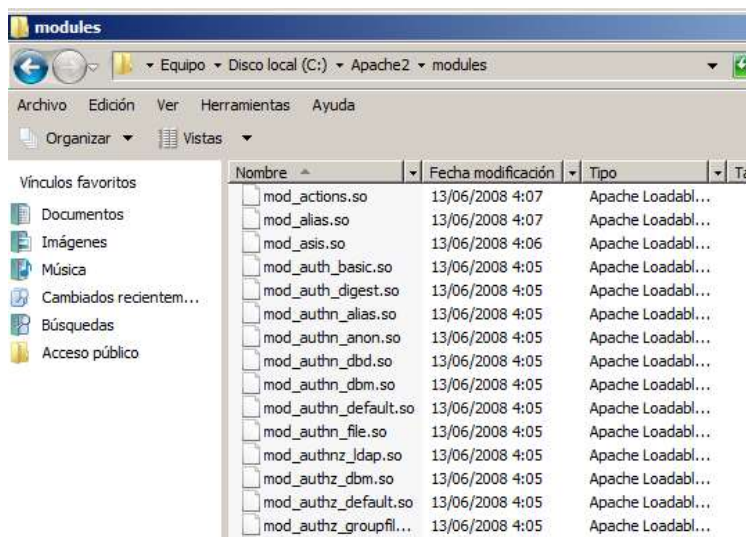
➤ Deshabilitar un módulo: **a2dismod** nombre_modulo

Borra el enlace simbólico de mods_enabled.

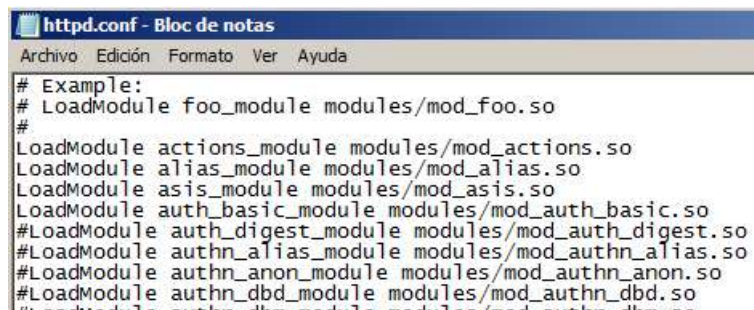
✓ Hay que **reiniciar** Apache2 al habilitar /deshabilitar módulos.

8.2.- Módulos disponibles en Windows

◦ `C:\Apache2.2\modules`



Habilitar/deshabilitar módulos en httpd.conf



9.- Control de acceso a recursos: ficheros, directorios, URLs, ...

➤ Control de acceso por host (IP/nombre_dominio)

Módulo mod_authz_host.

Order Deny,Allow

El **acceso está permitido por defecto**. Las directivas Deny se evalúan antes que las directivas Allow. Cualquier cliente que “no case” con una directiva Deny a Allow tendrá permitido el acceso. Si el cliente “casa” al mismo tiempo en una directiva Allow y otra Deny, tendrá permitido el acceso por que las directivas Allow se evalúan las últimas.

Order Allow,Deny

El **acceso está denegado por defecto**. Las directivas Allow se evalúan antes que las directivas Deny. Cualquier cliente que “no case” con una directiva Deny a Allow tendrá denegado el acceso. Si el cliente “casa” al mismo tiempo en una directiva Allow y otra Deny, tendrá denegado el acceso por que las directivas Deny se evalúan las últimas.

Order Deny,Allow	1) Acceso permitido por defecto
Deny from all	2) Todos los hosts so denegados
Allow from daw.org	3) Se permite el acceso a los hosts de dominio *.daw.org

Resultado: **Solo los host de *.daw.org son permitidos.**

Order Allow, Deny	1) Acceso denegado por defecto
Allow from daw.org	2) Se permite el acceso a los hosts de dominio *.daw.org
Deny from bbdd.daw.org *.bbdd.daw.org	3) Se deniega el acceso a los hosts de dominio *.bbdd.daw.org

Resultado: **Los hosts de *.daw.org son permitidos execepto los de *.bbdd.daw.org.**
¿Qué ocurre si se cambia el orden a **Order Deny,Allow** ?

Order Allow,Deny
Allow from 200.200.100.0/24
Deny from www.daw.org

¿Resultado?

Order Allow,Deny
Allow from 192.168.0.0/16
Deny from all

¿Resultado?

Se deniega a todos el acceso.

Otros ejemplos:

Configuration1.1:

Order deny,allow
Deny from all

Configuration1.2:

Require all denied

⇒ En estos ejemplos se deniega el acceso.

configuration2.1:

Order allow,deny
Allow from all

Configuration2.2:

Require all granted

⇒ En estos ejemplos se permite el acceso.

configuration3.1:

Order Deny,Allow

Deny from all

Allow from daw253.net

Configuration3.2:

Require host daw253.net

⇒ En estos ejemplos, todos los hosts del dominio daw253.net tienen permitido el acceso, el resto de hosts lo tienen denegado.

<http://httpd.apache.org/docs/2.4/upgrading.html>

<http://httpd.apache.org/docs/2.4/mod>

<http://httpd.apache.org/docs/2.4/es/howto/access.html>

10.- Autenticación y Autorización

Autenticación: Proceso para verificar que alguien es realmente quien dice ser.

Autorización: Acceder a algo que quiere.

<http://httpd.apache.org/docs/2.4/es/howto/auth.html>

➤ Tipos de autenticación

- **Basic:** Módulo mod_auth_basic.

La contraseña es enviada por el cliente en texto plano.

Autenticación y autorización sobre fichero de texto (htpasswd).

mod_authn_file

mod_authz_user

1) Crear fichero con usuarios/contraseñas **Htpasswd**

- Hay que instalar el paquete **apache2-utils**

- <http://httpd.apache.org/docs/2.4/es/programs/htpasswd.html>

```
# La primera vez que se invoca el comando se
# utiliza a opción -c para crear el fichero
htpasswd -c /etc/apache2/passwd profesor1

# Añade un nuevo usuario al fichero
htpasswd /etc/apache2/passwd profesor2

# Borrar un nuevo usuario al fichero
htpasswd -D /etc/apache2/passwd profesor1
```

2) Definir directivas

```
<Directory /var/www/html/profesor>
  Options Indexes FollowSymLinks
  AllowOverride None
  AuthType Basic
  AuthName "Acceso restringido"
  AuthUserFile /etc/apache2/passwd
  <RequireAll>
    Require user profesor1 profesor2
    <RequireAny>
      Require ip 127.0.0.1
      Require ip 192.168.1.16
    </RequireAny>
  </RequireAll>
</Directory>
```

```
<Directory "/var/www/privado">
AuthType Basic
AuthName "Directorio privado"
AuthUserFile /etc/apache2/passwd/.htpasswd
Require valid-user
</Directory>
```

AuthType, AuthName, AuthUserFile, AuthGroupFile, Require, ...

- **AuthName:** nombre del **dominio de autenticación**. Define el conjunto de recursos que estarán sujetos a los mismos requisitos de autenticación. También es el texto que aparecerá en la ventana que pide el usuario y la clave.
- **AuthType:** tipo de autenticación.
 - Basic:** la contraseña se negocia sin encriptar
 - Digest:** la contraseña se negocia encriptada
- **AuthUserFile:** ubicación del archivo de texto que contendrá los nombres de usuario y contraseñas usadas en la autenticación HTTP básica. Se suele llamar **.htpasswd**. Previamente hay que crear el directorio **/etc/apache2/passwd/**.
- **Require:** usuarios que tienen acceso a los recursos especificados. Opciones disponibles:
 - **valid-user:** cualquier usuario incluido en el archivo de contraseñas **.htpasswd**.
 - **user <lista de usuarios>:** lista de usuarios de **.htpasswd**, separados por espacios, que pueden acceder.
- **Satisfy:** al utilizar esta directiva determina si se deben cumplir todos los requisitos (**All**) o cualquiera (**Any**).

Para crear usuarios para el método de autenticación Básico se utiliza la orden **htpasswd**.

#htpasswd-c /etc/apache2/passwd/.htpasswd nombre_usuario

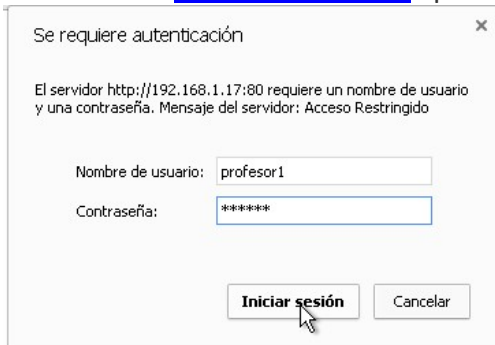
La opción **-c** permite crear el archivo **.htpasswd** con el primer usuario dado de alta, que además no tiene porque ser un usuario existente en el sistema.

Los permisos del archivo **.htpasswd** deben ser **644**, es decir lectura y escritura para el dueño, que es root y lectura para el grupo y los otros.

Para seguir dando de alta usuarios no hay que poner el argumento **-c** de lo contrario creará siempre de nuevo el archivo con sólo el último usuario incorporado.

Los usuarios creados para Apache2 no tienen porqué estar dados de alta en el sistema, y si existen no tienen porqué tener la misma contraseña.

Al ir a la URL <http://IP/privado/> aparece la ventana:



Las directivas `<RequireAll>`, `<RequireAny>` and `<RequireNone>` se pueden combinar con la directiva `Require` para controlar qué lógica de autorización se utilizará http://httpd.apache.org/docs/2.4/mod/mod_authz_core.html

- **Digest:** Módulo `mod_auth_digest`.

La contraseña se envía cifrada (¡¡ cifrado débil, no es seguro !!) por el cliente.

- Autenticación y autorización sobre fichero de texto (`htdigest`).

1) Crear fichero con usuarios/contraseñas asociados a un dominio (realm).

<http://httpd.apache.org/docs/2.4/es/programs/htdigest.html>

```
# La primera vez que se invoca el comando se
# utiliza a opción -c para crear el fichero
htdigest -c /etc/apache2/passwd informatica admin1

# Añade un nuevo usuario al fichero
Htdigest /etc/apache2/passwd informatica admin2

# Borrar un nuevo usuario al fichero
htdigest -D /etc/apache2/passwd informatica admin1
```

2) Definir directivas

`AuthType`, `AuthName`, `AuthDigestProvider`, `AuthUserFile`, `AuthGroupFile`, `Require`, El módulo que controla este método de autenticación es **`mod_auth_digest`**. Tiene la ventaja de que el login y la contraseña van cifradas del navegador web al servidor. Por el contrario, tiene el inconveniente de que no está soportado por todos los navegadores web.

Lo primero que hay que hacer es activar dicho módulo. Para ello:

```
#a2enmod auth_digest
#/etc/init.d/apache2 force-reload
```

Utiliza **MD5** (Message Digest Authentication) para generar un **hash** que es el que se transmite o envía al servidor.

En el archivo `/etc/apache2/apache2.conf` habrá que añadir un bloque `<Directory>...</Directory>` por cada directorio que queramos proteger:

```
<Directory "/var/www/privado">
AuthName "Directorio privado"
AuthType Digest
AuthDigestDomain http://servidor.asirxx.net/privado/
```

Donde:

AuthName: indica el nombre del dominio de autenticación (realm).

AuthType: indica que el método a usar es 'Digest'.>

AuthDigestProvider: indica el soporte utilizado para la autenticación. Por defecto es file (archivo).

AuthDigestDomain: dominio protegido con autenticación digest.

AuthUserFile: indica donde se encuentra el archivo de contraseñas que ahora llamamos **.htdigest**.

La creación de usuarios en el método de autenticación Digest requiere la orden **htdigest**.

```
#htdigest /etc/apache2/passwd/.htdigest zona_privada nom_usuario
```

El parámetro '**zona_privada**' debe coincidir exactamente con el nombre del dominio de autenticación dado en la directiva **AuthName** ya que, cuando se crea un usuario, se hace incluyéndolo a un dominio de autenticación concreto.

Si la directiva **Require** indica 'valid-user', se consideran usuarios válidos sólo los que pertenecen al dominio de autenticación dado en **AuthName** y las contraseñas sólo pueden utilizarse en este dominio.

En el ejemplo añadimos el usuario *usuario1* al archivo de contraseñas **/etc/apache2/passwd/.htdigest**. Si se utiliza el archivo **.htdigest** por primera vez y no existe, hay que incluir la opción **-c**:

```
#htdigest -c /etc/apache2/passwd/.htdigest "Directorio privado" usuario1
```

En el archivo de configuración **/etc/apache2/apache2.conf** hay que añadir un bloque **<Directory>...</Directory>** para el directorio que queremos proteger:

```
Alias /privado /var/www/privado
<Directory "/var/www/privado">
AuthType digest
AuthName "Directorio privado"
AuthUserFile /etc/apache2/passwd/.htdigest
Require user usuario1
</Directory>
```

11.- Ficheros que permiten la configuración personalizada de directorios.

➤ Fichero de configuración de Apache.

```
Alias /blog /home/profesor/blog
<Directory /home/profesor/blog>
    AllowOverride All
</Directory>
```

➤ Fichero **.htaccess** dentro de un directorio.

```
Options Indexes
AuthType Digest
AuthName "informatica"
AuthDigestProvider file
AuthUserFile /home/profesor/blog/.htdigest
Require user blog
```

Cada vez que se produce una petición:

- El servidor busca en la ruta del recurso que ha solicitado el cliente un fichero con el nombre .htaccess.
- Aplica sobre el directorio las directivas definidas.

En la configuración del servidor hay que permitir el uso de estos ficheros.

<http://httpd.apache.org/docs/2.4/es/howto/htaccess.html>

Ej. Defina la siguiente directiva a nivel del servidor principal para que los ficheros que empiecen con .ht no sea visibles por los clientes.

◦ Windows

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

◦ Linux

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

No se deben usar a menos que no se tenga acceso al archivo de configuración del servidor (Ej.: Servidor de hosting)

- Eficiencia.
- Seguridad.

El nombre .htaccess se puede cambiar con la directiva AccessFileName.

12.- Ficheros de registros (logs)

En Linux:

- Errores
/var/log/apache2/error.log
- Accesos
/var/log/apache2/access.log

En Windows:

- Errores

C:\Apache2.2\log/error.log

- Accesos

C:\Apache2.2\log/access.log

Existen múltiples directivas para personalizar logs.

- ErrorLog
- LogLevel
- CustomLog
- LogFormat

- Apache tiene que tener permisos para escribir en el directorio donde se guardan los ficheros de logs.

<http://httpd.apache.org/docs/2.4/en/logs.html>

13.- Monitorización: Módulos para monitorizar el servidor

- mod_status

http://httpd.apache.org/docs/2.4/mod/mod_status.html

- mod_info

http://httpd.apache.org/docs/2.4/mod/mod_info.html

- Analizadores de logs

Webalizer (<http://www.webalizer.org/>)

Awstats (<http://www.awstats.org/>)

Visitors (<http://www.hping.org/visitors/>)

Analog (<http://www.analog.cx/>)

14.- Licencia

La licencia de software bajo la cual el software de la fundación Apache es distribuido es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto. La Licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

Ver Portal Software libre. https://es.wikipedia.org/wiki/Portal:Software_libre

Referencias

1. https://es.wikipedia.org/wiki/Servidor_HTTP_Apache
2. Servicios de Red e Internet. Álvaro García Sánchez, Luis Enamorado Sarmiento, Javier Sanz Rodríguez. Editorial Garceta.

3. <http://www.w3c.org>
4. Sitio web del Proyecto Apache: <https://httpd.apache.org/>
5. Instalar y configurar Apache HTTP server:
<http://www.librebyte.net/apache/instalar-y-configurar-apache-http-server/>
6. Netcraft: <http://news.netcraft.com/>
7. <http://www.apache.org>
8. Documentación: <http://httpd.apache.org/docs/2.2>
9. Módulos: <https://modules.apache.org/>
10. <http://www.bdat.net/documentos/apache/x1121.html>