06052 SEMINARIO

CONFIGURACIÓN DE APACHE 2

Ramón M. Gómez Labrador

(ramon.gomez@eii.us.es)

Marzo de 2.006

Nota importante: El presente seminario se oferta dentro del plan de formación para personal informático de la Universidad de Sevilla para el año 2006 y toda su documentación asociada está bajo licencia Creative Commons (http://creativecommons.org/licenses/by/2.5/es).

06052 Seminario sobre Configuración de Apache 2

Índice

1. Introducción.

El servicio de hipertexto a contribuido en gran parte al auge de Internet, ya que permite ofrecer información mediante documentos que incluyen texto, gráficos, datos y enlaces o vínculos a otros documentos -que pueden estar almacenados en otros ordenadores-, creando una especie de *telaraña global* (World-Wide Web). Para ver el contenido de dichos documentos debe utilizarse un programa especial denominado navegador.

Este concepto fue creado por Tim Verners-Lee en el Centro Europeo para Física de Partículas (CERN), como un mecanismo para la comunicación e intercambio de información entre los investigadores del centro.

La WWW se basa en [1]:

- Un protocolo de transporte conocido como **HTTP** (Hipertext Transfer Protocol).
- Un lenguaje que permite crear documentos; conocido **HTML** (Hipertext Makeup Language) y sus variantes.
- Una estructura cliente/servidor, donde la información se publica en el servidor y el programa navegador (cliente) muestra los datos según el formato indicado en el documento.

Desde su creación, tanto el protocolo HTTP como el lenguaje HTML han evolucionado rápidamente, incluyendo una serie de mejoras que permiten realizar documentos más comprensibles y cómodos de leer. También han aparecido nuevos lenguajes para una comunicación más interactiva entre el servidor y el usuario (CGI, *aplets* Java, PHP) y que contribuyen a crear estilos de presentación (CSS), a crear documentos dinámicos (XML) o a generar espacios de realidad virtual (VRML).

En los navegadores es en donde ha existido una evolución más rápida, ya que actualmente pueden utilizar varios protocolos e incluso ejecutar aplicaciones. Así, un mismo programa puede usarse para ver documentos, para transferir ficheros, para leer el correo electrónico o para publicar contenido.

El paradigma más extendido para desarrollar servidores de información mediante aplicaciones de código abierto y gratuitas es el conocido como **LAMP**: Linux, Apache, MySQL y PHP/PERL/Python. Los aspectos que hacen interesante el uso de estas herramientas para la publicación de información en Internet son:

- El ahorro de costes, ya que todos los programas y sistemas son gratuitos. Por contra se requiere una cierta especialización en el personal de administración y desarrollo del servicio.
- La versatilidad de las aplicaciones permiten crear configuraciones a la medida de cada sistema, ya que todos ellos cuentan con la posibilidad de ampliaciones y módulos que complementan o añaden beneficios al sistema.
- La existencia de gran cantidad de aplicaciones y módulos adicionales y gratuitos, que ayudan a mejorar la gestión y el acceso.

- La posibilidad de acceder de forma segura a la información, definiendo distintos niveles de control.
- La estructura se adapta bastante bien para crear gestores de contenidos y servidores de aplicaciones, incluso algunos de estos programas son gratuitos y fácilmente instalables. En su contra, el sistema empieza a tener ciertos problemas cuando las bases de datos son excesivamente grandes.

Apache [ii] es el servidor para hipertextos de código abierto más utilizado y está incluido en todas las distribuciones del sistema operativo Linux.

El resto de capítulos de esta documentación se centran en la instalación, configuración y desarrollo de un servidor *web* basado en Apache.

2. Instalación.

Apache mantiene dos versiones distintas de su servidor HTTP, que utilizan una filosofía de trabajo similar. Sin embargo, la versión 2 de Apache incluye algunas mejoras, que se describen en la siguiente lista.

- Posibilidad para trabajar con multiprocesos multihilvanados en sistemas Unix con soporte POSIX.
- El proceso de compilación del servidor se asemeja al de las herramientas GNU.
- Incluye una gestión de módulos más flexible.
- Soporte para IPv6.
- Soporte para filtrado de información.
- Respuesta de errores en diferentes idiomas de forma dinámica.
- Inclusión de la biblioteca para compatibilidad con expresiones regulares del lenguaje de programación PERL (PCRE).
- Inclusión de módulos propios para interactuar con SSL, DAV, compresión de datos, cabeceras, sesiones, autentificación, etc.

El Apache Server Project ha anunciado en diciembre de 2.005 una nueva rama de trabajo conocida como versión 2.2, que añade ciertas funcionalidades de control de acceso y rendimiento sobre la rama 2.0.

El seminario se centra básicamente en la configuración de un servidor Apache 2.0; sin embargo el administrador del servicio debe realizar un análisis previo de las ventajas ofrecidas por cada versión y de los requisitos necesarios para implantar el sistema deseado.

2.1. Instalación mediante paquetes.

En caso de utilizar paquetes de programas precompilados para el sistema operativo, las distintas versiones del servidor necesitan la instalación de requisitos previos o módulos adicionales.

El siguiente cuadro de muestra la orden para instalar desde la red los paquetes necesarios para ejecutar la última versión del servidor Apache compilada para una distribución de Linux Fedora Core 4.

```
yum install httpd httpd-devel mod_ssl
```

Aunque el administrador del sistema puede modificar los puertos de comunicaciones utilizados por Apache, los servicios usados normalmente por el servidor *web*, y que deben revisarse en el archivo `/etc/services`, son los siguientes:

```
http        80/tcp      # World Wide Web HTTP
http        80/udp      # World Wide Web HTTP
https       443/tcp     # http protocol over TLS/SSL
https       443/udp     # http protocol over TLS/SSL
```

2.2. Compilación del código fuente.

Debido a la versatilidad y a la amplitud de posibilidades que Apache abre al gestor del *web*, muchos administradores utilizan los paquetes precompilados por las distribuciones de Linux. Sin embargo, aquellos que deseen tener una configuración personalizada deben descargar el código fuente, revisar detenidamente la información de instalación, hacer las instalaciones previas para los requisitos de utilidades y compilar el código.

Se recomienda, tanto para Apache 1.3 como para Apache 2.0, compilar el código para la opción de soporte para objetos compartidos dinámicamente (**DSO**), lo que permite cargar en memoria aquéllos módulos que se necesitan en un momento determinado. También se añade la posibilidad de compilar código ejecutable para componentes adicionales y cargarlos en el sistema cuando sea necesario.

El siguiente cuadro muestra un ejemplo de creación y compilación de Apache 1.3.

```
SSL_BASE="SYSTEM" \
./configure --prefix=/usr/local --logfiledir=/var/log/httpd \
            --enable-module=all --enable-shared=max \
            --with-perl=/usr/bin/perl --enable-suexec \
            --suexec-safepath=/bin:/usr/bin
make
make install
```

A continuación se muestra el cuadro de código para la compilación de la versión 2.0 de Apache, con soporte para objetos dinámicos en todos los módulos, compatibilidad con OpenSSL y con la biblioteca de compresión LibZ.

```
./configure --prefix=/usr/local/apache --enable-mods-shared=all \
            --enable-ssl --with-z
make
make install
```

Una vez configurado el servidor, el administrador puede preparar un arranque inmediato y automático ejecutando los siguientes mandatos:

```
/etc/init.d/apache start
chkconfig –a apache
```

3. Configuración.

3.1. Pasos iniciales.

Antes de comenzar a preparar la configuración del servidor, el administrador del servicio debe planificar detenidamente los requisitos y tener en cuenta los siguientes aspectos:

Módulos:	Revisar las características funcionales de cada módulo de Apache y enumerar aquellos que pueden ser cargados por el servicio.
Tipo de servidor:	Apache puede arrancarse como un servicio independiente (recomendado) o estar integrado dentro del metaservicio de red Inetd.
Propietario y grupo:	Crear -si es necesario- el grupo y la cuenta del usuario ficticio que ejecuta las peticiones al servidor. Por motivos de seguridad, no es recomendable que el usuario `root` -ni cualquier otro administrador- sea propietario de los procesos del servidor.
Directorios:	Deberán crearse los directorios donde se encontrarán las páginas de hipertexto, los gráficos e iconos más comunes, los programas ejecutables (CGI, *aplets*, etc.) y los históricos de accesos al servidor. Establecer los permisos adecuados en cada uno de ellos.
Informes y registro:	Establecer parámetros sobre el tipo de información que será almacenada en relación con los accesos al servidor. El administrador deberá revisar estos ficheros para corregir parámetros, prevenir intrusiones o evitar errores en los accesos al servidor.
Política de acceso:	Creación de usuarios y grupos propios del servidor para acceder a información reservada, posibilidad de mostrar el contenido de los directorios, permitir que los usuarios normales de la máquina puedan publicar páginas personales o ejecutar programas, restringir o permitir el acceso desde ordenadores o dominios específicos, habilitar el uso de páginas dinámicas (PHP, PERL, etc.)
Otros programas:	Compilar y configurar otros programas que pueden ayudar a la gestión del servidor o que mejoran sus capacidades de presentación (contadores, *servlets*,

	interfaces para acceso a bases de datos, creación de copias de seguridad de la información, estadísticas de accesos, páginas dinámicas, etc.). El propio servidor Apache viene equipado con varios módulos de ampliación (que permiten generar estadísticas, compatibilidad con PERL, PHP, etc. y que pueden ser añadidos durante el proceso de configuración.
Servidores virtuales:	Configuración de otros servidores virtuales que pueden definirse en el mismo ordenador. Especificar un nombre ficticio y una configuración completa para cada uno de ellos, así como los puertos de comunicaciones donde debe "escuchar" para servir los datos, o la posibilidad de estar localizados en máquinas diferentes.

Después de preparar una configuración inicial, el gestor del servidor *web* deberá iniciar los procesos que lo activan. Cada vez que se modifica cualquier parámetro de configuración, deberá de procederse con la parada y al rearranque del servicio correspondiente o bien con la recarga del fichero de configuración.

Si el gestor del sistema ha optado por preparar una configuración del servicio `httpd` basada en el Inetd, deberá añadirse la línea correspondiente en el fichero `/etc/inetd.conf` (en caso de usar el metaservicio Xinetd, debe configurarse el fichero correspondiente en el directorio `/etc/xinetd.d`).

Los dos próximos apartados ayudarán a comprender cada aspecto relativo a la configuración del servidor de hipertextos. El administrador del sistema puede obtener mayor información dirigiéndose a la documentación suministrada con el servidor Apache o en la dirección de Internet del Proyecto Apache [ii].

3.2. Ficheros de configuración.

A partir de la revisión 1.3.4 del servidor Apache sólo se utiliza un único fichero de configuración (`httpd.conf`), para evitar posibles redundancias o confusiones. Las versiones anteriores (hasta la 1.3.3) conservaban tres ficheros de configuración:

Fichero	Descripción
`httpd.conf`	Datos de control del servidor.
`access.conf`	Datos para el control de accesos.
`srm.conf`	Datos sobre especificación de ficheros

A partir de la versión 2.0 de Apache, se utiliza también un directorio (normalmente `conf.d`) donde se localizan los ficheros de configuración complementarios para los módulos añadidos.

Como se ha comentado, las versiones 1.3 (a partir de la 1.3.4) y 2.0 de Apache utilizan un fichero de configuración principal denominado `httpd.conf`. Las distribuciones de Linux suelen colocar este fichero en el directorio `/etc/apache`; sin embargo, cuando se compila el programa con las opciones por defecto, éste se sitúa en el directorio `${PREFIX}/etc`, siendo `${PREFIX}` el directorio de instalación del servidor.

Desde el programa de configuración pueden cargarse otros ficheros adicionales (usando la orden `include`), para distribuir lógicamente las opciones relacionadas con los módulos extra o con características especiales.

El fichero `httpd.conf`, y cualquier otro archivo llamado por éste, consta de una sintaxis basada en una serie de directivas de configuración, que se pueden clasificar en [1]:

- **Simples**: una directiva por línea.
- **Compuestas**: bloque de código que incluye una o varias directivas, tanto simples como complejas.

El formato genérico de una directiva simples es:

```
Directiva Parámetro ...
```

Y el formato del bloque de código de una directiva compuesta es el siguiente:

```
<Directiva Parámetro ...>
Directivas ...
...
</Directiva>
```

Las líneas con comentarios comienzan con el símbolo almohadilla (#). Las expresiones regulares suelen ir precedidas por el símbolo tilde (~).

Adicionalmente se distribuye otro fichero de configuración (`mime.types`) que permite especificar los tipos de documentos que serán suministrados por el `httpd`.

La siguiente tabla muestra algunas de las directivas más comunes para configurar un servidor Apache 2.0.

Directiva	Descripción
`LoadModule`	Carga el módulo correspondiente. Cada módulo habilita una serie de directivas.
`ServerType`	Tipo de ejecución del servidor, independiente o basada en `inetd`.

ServerName	Nombre completo del servidor
Port	Puerto TCP gestionado por el servidor
HostnameLookups	Búsqueda de clientes por nombre o por dirección IP. La búsqueda por nombres ralentiza la respuesta del `httpd`, es conveniente registrar los accesos por dirección IP y posteriormente revisarlos con programas estadísticos que soliciten los nombres al DNS.
User	Usuario ficticio propietario de los procesos del servidor.
Group	Grupo ficticio propietario de los procesos del servidor. Los usuarios reales que escriban páginas *web* deben pertenecer a este grupo.
ServerAdmin	Dirección de correo del administrador del administrador del servicio.
ServerRoot	Directorio de configuración.
ErrorLog	Fichero histórico de errores (referido a **ServerRoot**).
CustomLog	Otros ficheros históricos (referido a **ServerRoot**).
KeepAlive	Posibilidad de habilitar el uso de conexiones persistentes (recomendado).
StartServers	Número de procesos servidores que deben arrancarse (su función varía según el módulo usado para multiprocesos, MPM). Valor dependiente de la potencia del servidor.
MaxClients	Número máximo de procesos en ejecución o de clientes conectados (según el módulo MPM usado). También depende del tipo de servidor utilizado.
Listen	Permite la "escucha" en otros puertos para crear servidores virtuales.
<VirtualHost>	Directivas de configuración de un servidor virtual.
<Directory>	Directivas de configuración para accesos a directorios.
<Location>	Directivas de configuración de servicios asociados a URLs.
<Files>	Directivas de configuración asociadas a nombres de ficheros.
DocumentRoot	Directorio donde se encuentran los documentos principales del servidor.
UserDir	Directorio de los documentos personales de los usuarios.
DirectoryIndex	Archivo o programa que contiene el índice o la página principal

	de un directorio.
`AddIcon`	Incluye iconos que permiten identificar tipos de archivos.
`Alias`	Asocia nombres (alias) a directorios.
`ScriptAlias`	Indica los alias para directorios que incluyen programas CGI.
`Redirect`	Indica a los clientes que el documento está en una nueva URL externa. **Nota**: ver la documentación del módulo `mod_rewrite` ya que permite una mayor versatilidad para reescribir las URLs, tanto para redirecciones internas como externas.
`ErrorDocument`	Permite diseñar documentos que gestionan errores de acceso al servidor.

Asimismo, las distribuciones de Linux añaden algunas herramientas gráficas que ayudan a generar ficheros de configuración de forma más cómoda.

3.3. Configuración para acceso seguro con SSL.

SSL (Secure Socket Layer) es un protocolo creado por Netscape para la realizar una comunicación segura y codificada entre el servidor y el navegador, certificados por una autoridad competente. **TLS** (Transport Layer Security) es la evolución de SSL desarrollado por la IETF. La versión 1 de TLS se identifica con la versión 3 del protocolo SSL.

Aunque inicialmente SSL se desarrolló para trabajar con el protocolo HTTP, generando el protocolo para transmisión de hipertexto seguro (**HTTPS**), sin embargo SSL/TLS se aplica como capa de seguridad a casi la totalidad de protocolos de Internet

El módulo `mod_ssl` controla la interfaz de comunicación entre Apache y OpenSSL [iii]. Evidentemente, ambas utilidades deben estar instaladas en la máquina. La versión 2.0 de Apache incluye en su código este módulo, sin embargo la versión 1.3 requiere que éste sea descargado e instalado a parte.

El módulo `mod_ssl` incluye bastantes directivas de configuración, de ahí que sea conveniente utilizar un fichero independiente para preparar el sistema, localizado .

Directiva	Descripción
`SSLEngine`	Activación del motor SSL/.
`SSLCACertificateFile`	Indica el fichero con el certificado de la Autoridad de

	certificación.
SSLCARevocationFile	Fichero con la lista de certificados revocados (CRL) por la autoridad de certificación.
SSLCertificateFile	Fichero del certificado del servidor que puede incluir su clave privada.
SSLCertificateKeyFile	Clave privada y codificada del certificado. Este fichero debe estar bien protegido.
SSLCipherSuite	Lista de algoritmos de cifrado usados por OpenSSL.
SSLOptions	Opciones SSL para directorios.
SSLProtocol	Protocolos usados (SSLv2, SSLv3, TLS).
SSLRequire	Definición de requisitos de acceso.
SSLRequireSSL	Obligación de usar HTTPS.
SSLUserName	Variable que almacena el nombre de usuario.
SSLVerifyClient	Nivel de verificación de clientes

3.3. Configuración con PHP.

Según la definición de PHP [2], éste es un lenguaje evolucionado a partir de PERL para el desarrollo de *scripts* dinámicos, orientado a objetos, interpretado, robusto, seguro, de altas prestaciones e independiente de la arquitectura.

PHP (Preprocesador de Hipertexto) [iv] es un lenguaje escrito en C, que incluye características de C, C++, PERL, Python y Awk. Aunque se utiliza para generar de forma dinámica las páginas de información en HTML (o XML), PHP también puede usarse como lenguaje de programación local. Los programas PHP se ejecutan en el servidor -al contrario que las rutinas JavaScript, que se ejecutan en el cliente- y es una alternativa válida a los lenguajes comerciales como JSP o ASP.

Otras características adicionales del lenguaje PHP son:

- El compilador Zend puede generar ficheros ejecutables basados en PHP.
- Puede combinarse con código en JavaScript o AJAX.
- Soporte completo para el acceso a varios gestores de bases de datos.
- Generación automática del módulo para trabajar con Apache.
- Control de accesos y de registros de incidencias.
- Soporte para publicación de ficheros desde el cliente.

- Creación dinámica de gráficos (usando bibliotecas externas).
- Distintos tipos de variables, matrices, expresiones regulares, clases, etc.
- Gestión de sesiones y "*cookies*".
- Tratamiento de ficheros XML/XSLT (PHP 5).
- Tratamiento de excepciones y control de errores (PHP 5).

Las distribuciones de Linux incluyen los paquetes necesarios para instalar el módulo PHP para Apache. Dependiendo de la versión del sistema operativo, el paquete para instalar el lenguaje puede denominarse `mod_php`, `mod_php5`, `apache2-mod_php5` o venir incluido en el propio paquete del PHP.

Si el administrador prefiere compilar e instalar los fuentes, debe revisar la información de la instalación para tener preparados los requisitos necesarios. En el siguiente ejemplo se muestra los mandatos para instalar PHP 5 usando los programas Apache 2.0, OpenSSL, Berkeley DB, MyQSL; y las bibliotecas para XML, XSLT, JPG, PNG, GD, MM, FreeType y Zlib.

```
./configure --with-apxs2=/usr/local/apache/bin/apxs \
            --with-ssl=/usr/local/ssl \
            --enable-mbstring --enable-calendar \
            --with-db3=/usr/local/BerkeleyDB.3.3 --with-dom=/usr \
            --with-dom-xlst -with-xml with-gettext \
            --enable-versioning --enable-memory-limit --enable-wddx \
            --with-gd --with-jpeg-dir=/usr --with-png-dir=/usr \
            --with-freetype-dir=/usr --with-zlib \
            --with-mysql=/usr/mysql --with-mm --with-pear
make install
```

Los únicos requisitos para la configuración de Apache con PHP se describen en el siguiente cuadro de código (fichero *ServerRoot*`/conf.d/php.conf`).

```
# Cargar el módulo para PHP.
LoadModule php5_module modules/libphp5.so
# El intérprete PHP usará los ficheros con extensión .php
AddHandler php5-script .php
AddType text/html .php
# Cláusula opcional para generar los índices de forma dinámica
DirectoryIndex index.php
```

Tras recargar el fichero de configuración de Apache, el administrador del servicio puede crear un pequeño programa y cargar la página en el navegador para probar los parámetros de ejecución de PHP.

```
<? php
// test.php - prueba de ejecución de PHP
phpinfo ();
?>
```

Las directivas de configuración listadas pueden modificarse editando el fichero correspondiente, denominado `php.ini`, cuya localización se lista en la página de información.

4. Seguridad.

Los archivos y directorios que forman parte del programa servidor de hipertextos sólo deben ser accesibles por los usuarios autorizados. Sólo aquellos directorios con información pública o propios de aquellas usuarios que pueden publicar datos, deben ser accesibles por el usuario ejecutor del `httpd`.

En ningún caso, un usuario deberá tener permiso para modificar los datos de otro usuario o la información principal del servidor.

En los próximos párrafos se incluyen las recomendaciones necesarias para incrementar la seguridad en el servidor. Cada uno de ellos irá acompañado con una o varias tablas que incluyen los mandatos que permiten realizar las funciones descritas. Cada línea puede ir precedida por un uno de los siguientes "puntos indicativos" [1]:

`#`: mandatos ejecutados por el usuario administrador (root).

`$`: mandatos ejecutados por otro usuario.

El gestor de la máquina debe ser el propietario de los directorios y de los archivos ejecutables, históricos y de configuración del Apache.

```
# cd DirectorioWeb
# chown root     . bin conf logs
# chgrp GrupoWeb . bin conf logs
# chmod 2750     . bin conf logs
```

Los ficheros y directorios con información global del servidor deben ser accesibles sólo para lectura por el usuario propietario de los procesos del servidor (cláusula **`User`**). Sólo en casos especiales -como en los archivos históricos o en algunos documentos generados por programas CGI- éste usuario tendrá permisos de escritura.

```
# chown User  logs/*
# chgrp Group logs/*
# chmod 640   logs/*
```

Los ficheros y directorios que contienen información general del servidor -incluidos en el directorio indicado en la cláusula **`DocumentRoot`**-, deberán tener permiso de lectura para el grupo ejecutor de los procesos.

```
# chown Propietario DocumentRoot
# chgrp Group       DocumentRoot
# chmod 2750        DocumentRoot
$ chmod 640         FicheroWeb
$ chmod 2750        DirectorioWeb
```

Si los usuarios del ordenador también pueden publicar datos en el servidor, deberán tener en sus cuentas un directorio que coincida con el indicado en la cláusula **`UserDir`** y un fichero dentro de dicho directorio con el nombre indicado en la cláusula

DirectoryIndex. Todos los ficheros y directorios publicados deben tener permiso de lectura par el grupo correspondiente a la cláusula **Group**.

```
$ cd
$ mkdir UserDir
$ chmod 711 .
$ chmod 755 UserDir
$ vi UserDir/DirectoryIndex
$ chmod 644 UserDir/DirectoryIndex
```

Nota: para restringir aún más el acceso a la información publicado pueden usarse los permisos 711 para el directorio de las páginas del usuario.

Los programas CGI deben ser ejecutables por el usuario indicado por la cláusula **User** o por el grupo de la cláusula **Group**. Sin embargo, no es conveniente que dicho usuario tenga permiso de modificación en estos ficheros. El usuario `root` sólo será propietario de ficheros ejecutables en casos estrictamente necesarios.

```
# chown root    ScriptAlias
# chgrp Group   ScriptAlias
# chmod 2750    ScriptAlias
# chown Usuario ScriptAlias/FicheroCGI
# chmod 750     ScriptAlias/FicheroCGI
```

El servidor Apache incluye también varias directivas que permiten restringir o verificar el acceso a determinados documentos.

Las subdirectivas incluidas en las cláusulas **<Directory>**, **<Files>** y **<Location>** -y las equivalentes para expresiones regulares **<DirectoryMatch>**, **<FilesMatch>** y **<LocationMatch>**- establecen los permisos básicos para acceder a directorios, ficheros y URLs específicos, respectivamente. La siguiente tabla muestra las directivas permitidas.

Subdirectiva	Descripción
Options	Opciones de acceso.
All	Todas las opciones excepto **MultiViews** (valor por omisión).
ExecCGI	Se permite la ejecución de programas CGI.
FollowSymLinks	Permite seguir enlaces simbólicos (ignorado en **<Location>**.
Includes	Permite incluir ficheros y programas en documentos HTML.
IncludesNOEXEC	Permite incluir ficheros, pero no programas.
Indexes	Genera automáticamente un listado del directorio si

	en él no existe el fichero especificado en la directiva **DirectoryIndex**.
MultiViews	Soporta la negociación de contenidos especificada en el protocolo HTTP/1.1.
SymLinksIfOwnerMatch	Se siguen los enlaces simbólicos si el origen y el destino del enlace son del mismo usuario (ignorado en **<Location>**.
AllowOverride	Indica qué conjunto de opciones pueden solaparse mediante el fichero para la configuración de accesos al directorio (`.htaccess`).
AuthConfig	Configuración para autentificar usuarios con permiso de acceso.
<Limit>	Directivas aplicadas a determinados métodos de acceso del protocolo HTTP.
Order	Orden de preferencia para las cláusulas de permito y denegación.
Allow	Permite el acceso a los nombres, direcciones IP o dominios indicados.
Deny	Deniega el acceso a los nombres, direcciones IP o dominios especificados.
Require	Indique qué usuarios o grupos tienen permiso de acceso.

El orden para el procesamiento de los permisos es:

1. Cláusulas de **<Directory>** sin expresiones regulares y las opciones solapadas por el fichero de accesos al directorio (`.htaccess`).
2. Cláusulas de **<DirectoryMatch>** y **<Directory>** con expresiones regulares.
3. Cláusulas de **<Files>** y **<FilesMatch>**.
4. Cláusulas de **<Location>** y **<LocationMatch>**.
5. Las directivas anteriores incluidas en la sección **<VirtualHost>** se aplican en último lugar.

La autentificación de usuarios y grupos se realiza a través de una serie de ficheros que deben estar fuera del árbol de directorios de documentos y únicamente con permisos de lectura para el usuario ejecutor de los procesos del servidor (directiva **User**).

El fichero típico para el control de las claves de los usuarios definidos para acceso al *web* tiene el siguiente formato:

```
Usuario:Clave
```

El fichero para la definición de grupos de usuarios consta de líneas con el siguiente formato:

```
Grupo: Usuario1 Usuario2 ...
```

La siguiente tabla muestra las directivas de configuración que gestionan los ficheros de usuarios y grupos en Apache 2.0.

Directiva	Descripción
AuthType	Declara cuál es el tipo de autentificación que va a utilizarse.
AuthName	Indica el nombre de autentificación enviado al programa cliente.
AuthUserFile	Establece el camino para el fichero de usuarios para la autentificación básica (módulo `mod_auth`).
AuthDigestFile	Establece el camino para el fichero de usuarios para la autentificación MD5 (módulo `mod_digest`). El administrador del servidor puede usar el programa **`htdigest`** (situado en el directorio *`ServerRoot`*`/bin`) para gestionar este tipo de ficheros. El algoritmo para codificación MD5 es más seguro que el usado por el método básico.
AuthGruopFile	Indica el camino para el fichero de grupos (módulo **`mod_auth`**).

El servidor Apache también soporta la autentificación mediante ficheros DB compatibles con las bibliotecas de bases de datos de Berkeley, a través de un servidor LDAP o mediante usuarios anónimos (equivalentes a los utilizados en el protocolo FTP).

Tanto el proceso de autentificación como el de cualquier otro tipo de entrada de datos sensibles debe realizarse mediante conexiones seguras usando HTTPS.

La versión 2.2 de Apache incorpora un nuevo conjunto de módulos que separan el control de la autentificación (`mod_authn_*`) y la autorización de acceso (`mod_authz_*`).

5. Ejemplos.

5.1. Ejemplo de configuración básica.

En este ejemplo vamos a comentar las características principales para configurar y arrancar un servidor Apache. Asimismo, accederemos a la página que nos confirmará el correcto funcionamiento del programa y verificaremos este hecho revisando los procesos que están ejecutándose en la máquina.

La tabla siguiente presenta el contenido, debidamente comentado, del archivo principal de configuración.

```
# httpd.conf - fichero de configuración global de Apache.
#
# En primer lugar, se especifican los módulos que serán cargados estática y
# dinámicamente. La línea que comienza con una almohadilla (#) indica
# comentario, o sea, dicho módulo no se carga.
# Por ejemplo, no se cargan los módulos para usar Apache como proxy, impidiendo
# el uso de las cláusulas de directivas <Proxy>.
# Por otro lado, el administrador ha considerado interesante cargar el módulo
# mod_env -que permite pasar variables de entorno a programas CGI- y sí podrá
# utilizar las directivas PassEnv o SetEnv.
#
LoadModule access_module modules/mod_access.so
LoadModule auth_module modules/mod_auth.so
LoadModule auth_anon_module modules/mod_auth_anon.so
LoadModule auth_dbm_module modules/mod_auth_dbm.so
LoadModule auth_digest_module modules/mod_auth_digest.so
LoadModule ldap_module modules/mod_ldap.so
LoadModule auth_ldap_module modules/mod_auth_ldap.so
LoadModule include_module modules/mod_include.so
LoadModule log_config_module modules/mod_log_config.so
LoadModule logio_module modules/mod_logio.so
LoadModule env_module modules/mod_env.so
LoadModule mime_magic_module modules/mod_mime_magic.so
LoadModule cern_meta_module modules/mod_cern_meta.so
LoadModule expires_module modules/mod_expires.so
LoadModule deflate_module modules/mod_deflate.so
LoadModule headers_module modules/mod_headers.so
LoadModule usertrack_module modules/mod_usertrack.so
LoadModule setenvif_module modules/mod_setenvif.so
LoadModule mime_module modules/mod_mime.so
LoadModule dav_module modules/mod_dav.so
LoadModule status_module modules/mod_status.so
LoadModule autoindex_module modules/mod_autoindex.so
LoadModule asis_module modules/mod_asis.so
LoadModule info_module modules/mod_info.so
LoadModule dav_fs_module modules/mod_dav_fs.so
LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule dir_module modules/mod_dir.so
LoadModule actions_module modules/mod_actions.so
LoadModule speling_module modules/mod_speling.so
LoadModule userdir_module modules/mod_userdir.so
```

```
LoadModule alias_module modules/mod_alias.so
LoadModule rewrite_module modules/mod_rewrite.so
#LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
#LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule cache_module modules/mod_cache.so
LoadModule suexec_module modules/mod_suexec.so
LoadModule disk_cache_module modules/mod_disk_cache.so
LoadModule file_cache_module modules/mod_file_cache.so
LoadModule mem_cache_module modules/mod_mem_cache.so
LoadModule cgi_module modules/mod_cgi.so
#
# Limitaciones de rendimiento para evitar saturaciones, indicando número máximo
# de procesos y de peticiones por proceso..
<IfModule prefork.c>
StartServers       8
MinSpareServers    5
MaxSpareServers   20
ServerLimit      256
MaxClients       256
MaxRequestsPerChild  4000
</IfModule>
#
# Servidor independiente del inetd, gestionando el puerto 80.
# El supervisor (root) es el único usuario que puede iniciar puertos TCP con
# valor menor a 1024. Sin embargo -por  razones de seguridad- no es conveniente
# que los procesos del httpd sean ejecutados por root. Por ello se definen
# usuario y grupo ficticios que ejecutan dichos procesos.
#
ServerType standalone
Port 80
User nobody
Group nobody
#
# Para mejorar el tiempo de respuesta, no se busca el nombre completo del
# ordenador cliente, sólo se registra su dirección IP.
HostnameLookups off
#
# Nombre completo del servidor y dirección de correo del administrador de
# Apache.
ServerName www.ejemplo.com
UseCanonicalName on
ServerAdmin info@www.ejemplo.com
#
# Directorio de configuración.
ServerRoot /etc/httpd
#
# Localización del fichero de errores (relativa a ServerRoot) y nivel de
# anotaciones.
ErrorLog logs/error_log
LogLevel warn
#
# Formatos de los ficheros históricos de accesos (combinado o normal), de
# referencias y de clientes.
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
#
# En este caso sólo se utiliza un fichero histórico de accesos con formato
# común.
CustomLog logs/access_log common
```

```
#CustomLog logs/referer_log referer
#CustomLog logs/agent_log agent
#CustomLog logs/access_log combined
#
# Ficheros especiales (PID del proceso principal, información interna, bloqueos
# para uso con NFS).
PidFile /var/run/httpd.pid
ScoreBoardFile /var/run/httpd.scoreboard
#LockFile /var/lock/httpd.lock
#
# Tiempo de espera normal.
Timeout 300
#
# Máximo número y tiempo de espera para comunicaciones persistentes.
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
#
# Número de procesos servidores normales y de repuesto usados para atender
# múltiples peticiones.
StartServers 10
MinSpareServers 8
MaxSpareServers 20
#
# Número máximo de clientes que pueden atenderse a la vez y número de
# peticiones que puede atender cada proceso hijo.
MaxClients 150
MaxRequestsPerChild 100
#
# Gestión de acceso.
#
# Protege todo el árbol de directorio, excepto aquellos que se indiquen en
# otros grupo </Directory>.
<Directory />
  Options None
  AllowOverride None
</Directory>
## Directorio púbico (indicado por la directiva DocumentRoot).
<Directory /home/httpd/html>
#
#       Opciones aplicables al árbol de directorios del servidor. No podrán
#       ser modificadas por ficheros .htaccess.
  Options Indexes Includes FollowSymLinks
  AllowOverride None
#       Acceso global a esta información.
  Order allow,deny
  Allow from all
</Directory>
#
# Directorio de programas (indicado por ScriptAlias).
<Directory /home/httpd/cgi-bin>
  AllowOverride None
  Options ExecCGI
</Directory>
#
# URL que muestra un informe sobre el estado del servidor, con acceso único
#  desde las máquinas del dominio .musho.es.
<Location /server-status>
  SetHandler server-status
  order deny,allow
  deny from all
  allow from .musho.es
</Location>
```

```
#
# Registra los intentos de acceso al servidor a través de un fallo en las
# primeras versiones de Apache. El gestor debe crear el programa error_phf.cgi,
# que registre los datos del posible agresor.
<Location /cgi-bin/phf*>
  deny from all
  ErrorDocument 403 /cgi-bin/error_phf.cgi
</Location>
#
# Directorio raíz de los documentos del servidor.
DocumentRoot /home/httpd/html
#
# Directorio raíz de los documentos de cada usuario (relativo a su $HOME).
UserDir public_html
#
# Índice HTML de un directorio.
DirectoryIndex index.html index.shtml index.cgi
#
# Genera índices con los archivos de un directorio si no existe ninguno de los
# archivos indicados en DirectoryIndex.
FancyIndexing on
#
# Cuando el índice de un directorio se genera automáticamente, estas directivas
# permiten asociar iconos a grupos de ficheros, según su codificación, su tipo
# o su extensión.
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
# ...
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^
DefaultIcon /icons/unknown.gif
#
# Archivos descriptivo y de cabecera del directorio.
ReadmeName README
HeaderName HEADER
# Archivos que se ignoran al generar el índice. Es recomendable añadir una
# referencia al fichero .htaccess.
IndexIgnore .??* *~ *# HEADER* README* RCS .htaccess
#
# Archivo modificador de las opciones de acceso al directorio.
AccessFileName .htaccess
#
# Codificación de los grupos de ficheros y tipo de fichero por omisión.
TypesConfig /etc/mime.types
AddEncoding x-compress Z
AddEncoding x-gzip gz
DefaultType text/plain
#
# Codificación y prioridad de sufijos de nombres de archivos para indicar el
# idioma de cada documento.
AddLanguage es .es
AddLanguage en .en
```

```
AddLanguage fr .fr
AddLanguage de .de
LanguagePriority es en fr de
#
# Establece los nuevos URLs de documentos que han sido movidos de lugar.
Redirect permanent /datos http://nuevo.servidor.es/datos/
Redirect temp      /pruebas /DirTemporal
#
# Alias para directorios específicos de iconos y de programas.
Alias /icons/ /home/httpd/icons/
ScriptAlias /cgi-bin/ /home/httpd/cgi-bin/
#
# Asociación de tipos de ficheros con acciones específicas de Apache (ficheros
# HTML o mapas analizados por el servidor, programas CGI, etc.).
AddHandler cgi-script .cgi
#
# Personalizar las respuestas de errores del servidor mediante textos,
# documentos HTML fijos o programas CGI.
#ErrorDocument 404 "Dcomento no encontrado.
#ErrorDocument 404 /error/NoHallado.html
ErrorDocument 404 /cgi-bin/error.pl
#
# Estas directivas corrigen algunos errores de comunicación con ciertas versiones
de algunos programas clientes.
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
```

Una vez realizada la configuración completa del servidor, éste debe arrancarse, usando su *script* para la gestión de servicios. En nuestro caso, ejecutar:

```
/etc/init.d/httpd start
```

Cada vez que el administrador deba realizar cambios en la configuración del servidor tendrá que pararlo y volverlo a arrancar o recargar el fichero de configuración. Entonces, ya pueden crearse los programas de mantenimiento y las páginas de información, que serán visualizadas con cualquier programa navegador.

Para comprobar el efecto real sobre la máquina, podemos observar los procesos del Apache que están siendo ejecutados en el ordenador.

```
ps -ef | grep httpd
nobody     301     1  0 Mar12 ?        18:25:12 httpd
nobody     302   301  0 Mar12 ?        18:25:13 httpd
nobody     303   301  0 Mar12 ?        18:25:13 httpd
nobody     304   301  0 Mar12 ?        18:25:13 httpd
nobody     305   301  0 Mar12 ?        18:25:13 httpd
nobody     306   301  0 Mar12 ?        18:25:13 httpd
nobody     307   301  0 Mar12 ?        18:25:13 httpd
nobody     308   301  0 Mar12 ?        18:25:13 httpd
nobody     309   301  0 Mar12 ?        18:25:13 httpd
nobody     310   301  0 Mar12 ?        18:25:13 httpd
```

5.2. Ejemplo de configuración con SSL.

El siguiente ejemplo describe la configuración de un servidor virtual seguro que usa el puerto 443 (HTTPS). El fichero de configuración del módulo `mod_ssl` se localiza normalmente en *ServerRoot*`/conf.d/ssl.conf`).

```
# ssl.conf
#
# Las directivas de configuración se activan sólo si está soportado el uso de
# SSL.
<IfDefine SSL>
#
# Escuchar en el puerto HHTPS.
Listen 443
# Añadir los tipos de archivos para la descarga de certificados y CRL.
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
# Control del diálogo para obtener claves.
SSLPassPhraseDialog  builtin
# Configuración del fichero y tiempo de validez para el caché de sesiones.
SSLSessionCache         dbm:/opt/apache/logs/ssl_scache
SSLSessionCacheTimeout  300
# Fichero para el semáforo de exclusión mutua (conexión entre procesos).
SSLMutex  file:/var/log/httpd/ssl_mutex
# Generador de números pseudo-aleatorios.
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
#
# Configuración del servidor virtual para HTTPS.
<VirtualHost _default_:443>
#
# Configuración general del servicio.
DocumentRoot "/var/www/htdocs"
ServerName www.ejemplo.com:443
ServerAdmin admin@ejemplo.com
ErrorLog /var/log/httpd/ssl_error_log
TransferLog /var/log/httpd/ssl_access_log
# Activar SSL.
SSLEngine on
# Algotirmos de cifrado soportados.
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
# Certificado y clave privada del servidor (directorios de OpenSSL en Fedora 4).
SSLCertificateFile /etc/pki/ssl/certs/server.crt
SSLCertificateKeyFile /etc/pki/ssl/private/server.key
# Certificado del servidor y de la autoridad.
SSLCertificateChainFile /etc/pki/ssl/certs/ca.crt
# Certificado de la autoridad y CRL.
SSLCACertificateFile /etc/pki/ssl/certs/ca-bundle.crt
# Se requiere la verificación del certificado del cliente.
SSLVerifyClient require
SSLVerifyDepth  10
# Control de acceso general para certificados de la empresa, con restricciones
# horarias y de dirección de red.
<Location />
SSLRequire (    %{SSL_CIPHER} !~ m/^(EXP|NULL)/ \
            and %{SSL_CLIENT_S_DN_O} eq "Ejemplo S.A." \
            and %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
            and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 21 ) \
            or %{REMOTE_ADDR} =~ m/^193\.50\.40\.[0-9]+$/
```

```
</Location>
#
# Se exportan las variables de de SSL para su uso en programas CGI.
<Files ~ "\.(cgi|shtml|phtml|php3?)$">
    SSLOptions +StdEnvVars
</Files>
<Directory "/usr/local/apache/cgi-bin">
    SSLOptions +StdEnvVars
</Directory>
# Características especiales de conexión para el navegador Internet Explorer.
SetEnvIf User-Agent ".*MSIE.*" \
         nokeepalive ssl-unclean-shutdown \
         downgrade-1.0 force-response-1.0
# Definición del formato del fichero para registro de incidencias.
CustomLog /opt/apache/logs/ssl_request_log \
          "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>
</IfDefine>
```

Una vez configurado el servidor, debe utilizarse la siguiente orden para arrancar el servicio con soporte para SSL.

```
/etc/init.d/apache startssl
```

5.3. Ejemplo de configuración con PHP.

El presente ejemplo muestra las directivas para modificar el comportamiento de PHP para usar la herramienta eGroupWare, personalizando parámetros del fichero de configuración `php.ini`.

```
<Directory "/var/www/html/egroupware">
    php_value memory_limit 32M
    php_value max_execution_time 60
    php_flag register_globals Off
    php_flag magic_quotes_gpc Off
    php_flag magic_quotes_runtime Off
    php_value upload_max_filesize 16M
    php_flag session.use_trans_sid Off
    php_value session.gc_probability 1
    php_value session.gc_divisor 10
    php_value mbstring.func_overload 7
</Directory>
```

6. Referencias.

1. R. M. Gómez Labrador: *"Servicios Internet en Linux"*. Centro de Formación y Perfeccionamiento del PAS (Universidad de Sevilla), 1.999.
2. A. Vasudevan: *"PHP HOW-TO, v27.6"*. Linux Documentation Project, 2002

i. Centro de Formación y Perfeccionamiento del P.A.S. de la Universidad de Sevilla: http://www.forpas.us.es/

ii. Apache HTTP Server Project: http://httpd.apache.org/

iii. OpenSSL. The Open Source toolkit for SSL/TLS: http://www.openssl.org/

iv. PHP Hypertext Preprocessor: http://www.php.net/