Parte I: Configuración del servidor web apache en linux-CentOS 6.0.

1. **Instalación del servidor web apache.**
2. **Archivos de configuración de apache.**
3. **Configuración del servidor.**
4. **Archivos log de apache.**
5. **Directorios virtuales. Directiva Alias.**
6. **Acceso a páginas web personales.**
7. **Configuración de Hosts Virtuales. Directivas de configuración.**
8. **Autenticación y autorización. Acceso a páginas privadas.**
9. **Servidor apache seguro.**

1. Instalación del servidor web Apache en centOS 6.

Apache es un servidor HTTP de código abierto y libre que funciona en Linux, sistemas operativos derivados de Unix™, Windows, Novell Netware y otras plataformas. Ha desempeñado un papel muy importante en el crecimiento de la red mundial, y continua siendo el servidor HTTP más implantado entre los distintos servidores que ofertan servicios web en Internet. (http://www.apache.org)

Entre las características más significativas destacamos:

- Es modular
- Permite crear servidores virtuales
- Permite crear servidores seguros https
- Permite crear sitios privados
- Permite crear sitios de usuario

Apache es un servicio que solo necesita instalar e iniciar. El servidor web apache en CentOS-Redhat-Fedora es el servidor o demonio: **httpd.** Debemos comprobar si está instalado e instalarlo, en su caso:

yum –y install httpd

Normalmente Apache ya incluye soporte para **PHP**/**MySQL**, **Perl**, **Python** y **SSL**/**TLS**, pero puede instalarse ejecutando:: **yum -y install php php-mysql mod_perl mod_python mod_ssl**

Para añadir el servicio a los servicios que se inician al arrancar el sistema, ejecutamos:
chkconfig httpd on

Para iniciar el servicio por primera vez, solo se necesita utilizar:
service httpd start

Para reiniciar el servicio, considerando que se interrumpirán todas las conexiones establecidas en ese momento, utilizar:
service httpd restart

Si el servicio ya esta trabajando, también se puede utilizar **reload** a fin de que Apache vuelva a leer y cargar la configuración sin interrumpir el servicio, y las conexiones establecidas.
service httpd reload

Para detener el servicio, solo se necesita utilizar:
service httpd stop

También habrá que abrir el servicio web (**puerto 80**) en el cortafuegos.

2. Archivos de configuración de apache.

El archivo de configuración principal de apache es: **httpd.conf** y se encuentra en el directorio: **/etc/httpd/conf/**

Además de este fichero existen unos **directorios importantes** para la configuración del servidor.

- **/var/www/html/**: directorio raíz de documentos (**DocumentRoot**) , donde se alojarán todos los documentos HTML y aplicaciones web.
- **/etc/httpd/** directorio principal del servidor (**ServerRoot**) donde residen los archivos y directorios principales de configuración del servidor. Cualquier ruta relativa que aparezca en el archivo de configuración de apache hace referencia a **/etc/httpd/**, por ejemplo en la directiva **ErroLog logs/error_log** logs será un enlace simbólico que se encuentra en /etc/httpd/ (y que nos lleva a /var/log).
- **cgi-bin/**: contiene scripts o programas CGI, su función principal es la de añadir dinamismo a los documentos web, se ejecutan en el servidor. Los programas **cgi** pueden estar compilados en diferentes lenguajes de programación. El más popular es **perl.**
- **conf/**: contiene ficheros de configuración del servidor.
- **conf.d/**: contiene archivos de configuración relacionados con otros módulos o aplicaciones como por ejemplo **php**, **ssl, Squid** ... Cualquier ajuste que se requiera realizar para una funcionalidad adicional, se puede realizar sin tocar el archivo principal de configuración, utilizando cualquier archivo con extension *.**conf** dentro del directorio /**etc/httpd/conf.d/**
- **modules** es un enlace simbólico al directorio **/usr/lib/httpd/modules** donde se encuentran los módulos (librerías ya compiladas) que permiten dar mayor funcionalidad al servidor web. Los módulos son archivos de extensión **.so** por ejemplo: **mod_ssl.so** es el módulo que debe incluirse en el servidor para dar soporte seguro (SSL) al servidor.
- **logs/**: es un enlace que nos lleva a **/var/log/httpd** a los archivos de registro de acceso y error: **/var/log/httpd/access_log** y **/var/log/httpd//error_log** (accesos con error).
- **icons/**: imágenes que utiliza Apache.

3. Configuración del servidor Web Apache.

El servidor lee la información del fichero de configuración: **/etc/httpd/conf(httpd.conf.**

Las directivas de configuración se agrupan en tres secciones:

- Configuración global.
- Configuración del servidor principal.
- Configuración de los Servidores Virtuales.

Configuración global (Global Enviroment)

En esta sección se definen las directivas que controlan el funcionamiento general de Apache, como pueden ser el número de peticiones concurrentes que puede manejar, donde puede encontrar los ficheros de configuración...

- **ServerRoot**:

  Define el directorio donde reside toda la información de configuración y registro que necesita el servidor, en nuestro caso **/etc/httpd/**

- **PidFile**:

  Indica el fichero en el cual el servidor registrará el identificador de proceso (PID) con el que se lanza.

- **ScoreBoardFile**:

  Fichero utilizado para almacenar información interna del proceso servidor. Se requiere para algunas arquitecturas para comunicar entre los procesos hijos y el padre.

- **Include conf.d/*.conf**
  Que se carguen todos los archivos de configuración que se encuentran en el directorio **/etc/httpd/conf.d/** por ejemplo, entre otros, se incluye el archivo de configuración para php: **/etc/httpd/conf.d/php.conf**

- **Timeout**:

  Número de segundos tras los cuales el servidor cierra la conexión.

- **KeepAlive**:

  Si se permiten o no conexiones persistentes. Si se pone On una vez realizada la conexión con un cliente se procesan todas las peticiones de ese mismo cliente sin tenerse que conectar de nuevo

- **MaxKeepAliveRequest**:

  El número máximo de peticiones permitidas durante una conexión persistente.

- **MinSpareServers**:

  Define el número de procesos servidores hijo desocupados que no atienden una petición.

- **MaxSpareServers**:

  Define el nº máximo de procesos servidor hijo desocupados que no manejan una petición. Si existieran mas de lo que define la directiva, el proceso padre mataría los procesos que exceden.

- **StartServers**:

  Número de procesos servidor hijo que serán creados cuando arranca Apache por primera vez, por defecto se arrancan 8, (cada servidor hijo atiende a un cliente)

- **Maxclients**:

  Define el número de peticiones simultáneas que apache puede soportar. Como máximo se crean este nº de procesos servidores hijo.

- **MaxRequestPerChild**:

  Define el nº de peticiones que cada proceso hijo tiene permitido procesar antes

de morir, es el nº máximo de peticiones que un apache hijo va a atender de un cliente, una vez realizada la conexión

- **ThreadsPerChild**:

  Nº de hilos concurrentes que el servidor permitirá utilizar. Este valor representa el nº máximo de conexiones que el servidor puede manejar a la vez.

- **Listen**:

  Esta directiva configura el servidor Apache para que escuche en más de una dirección IP o en más de un puerto.

- **LoadModule**:

  Esta directiva enlaza dentro del servidor la librería o fichero objeto (**/usr/lib/httpd/modules**) nombrado cuando se arranca el servidor y añade la estructura del modulo a la lista de módulos activos.

Configuración del servidor principal (Main Server Configuration).

Estas directivas definen los parámetros del servidor principal, el cual responde a las peticiones que no son manejadas por un host virtual. También proporciona parámetros por defecto para todos los hosts virtuales.

- **Port**:

  Define el puerto en el cual escucha el servidor (0 - 65535). El **puerto estándar** para el protocolo HTTP es el **80**.

- **User/Group**:

  Definen el usuario y el grupo con el que el servidor contestará las peticiones (usuario y grupo: **apache)**. El servidor inicialmente se ejecuta como root.

- **ServerAdmin**:

  Define la dirección de correo que el servidor incluirá en cualquier mensaje de error que devuelva al cliente.

- **ServerName**:

  Define el nombre de host del servidor (en **/etc/hosts** o que pueda ser resuelto por un servidor **DNS**).

- **DocumentRoot**:

  Define el directorio desde el cual el servidor servirá los documentos, normalmente **/var/www/html** o **/usr/local/apache/htdocs)**.

- **Directory**:

  <Directory></Directory> se utilizan para encerrar un grupo de directivas que se aplicarán al directorio en cuestión y sus subdirectorios. El parámetro directorio, puede ser una trayectoria completa o un metacaracter.

  Options: [+|-] opcion:

  Cuando varias directivas son aplicables a un directorio se toma la más específica, o sea no se mezclan a no ser que vayan precedidas de + o -, en cuyo caso se añaden o se eliminan. Por ejemplo en la declaración siguiente:

  <Directory /var/www/html/documentos>

  Options Indexes FollowSymLinks

</Directory>

<Directory /var/www/html/documentos/linux>

Options Includes ExecCGI

</Directory>

Las opciones aplicadas al directorio /var/www/html/documentos/linux son solamente Includes ExecCGI, en cambio si se pone:

<Directory /var/www/html/documentos>

Options Indexes FollowSymLinks

<(Directory>

<Directory /var/www/html/documentos/linux>

Options + Includes + ExecCGI -FollowSymLinks

</Directory>

Las opciones serían Indexes, Includes y ExecCgi.

Otras opciones son: **ExecCGI , FollowSymLinks, Includes, IncludesNOEXEC, Indexes, MultiViews…**

Indexes: Cuando se hace una solicitud al servidor y no se puede servir una de las páginas especificadas en **DirectoryIndex** se visualiza un listado con el contenido del directorio.

FollowSymLinks: Que se sigan los enlaces de la página

MultiViews. Permite visualizar diferentes contenidos web

Order allow,deny orden en el que se validarán las 2 siguientes cláusulas

Allow from all (equipos a los que se permite acceder al servidor se puede especificar una IP, una lista de IP o una dirección de red, por ejemplo: **10.168.53.0/24** o **10.168.53.**)**.** Se pueden especificar, también, nombres de equipos o de dominios, si pueden ser resueltos vía DNS.

Deny from all (equipos a los que NO se permite acceder al servidor)

Si se especifican dentro de un directorio se validan las dos (**Allow** y **Deny**, en el orden que indique la directiva **order**), sirve para permitir o denegar a ciertas máquinas (redes, subredes…) el acceso al directorio

AllowOverride "valor":

Se utiliza si se quiere proteger el directorio con control de acceso a usuarios autorizados. Las opciones para el control de acceso se suelen guardar en un archivo de nombre **.htaccess.** Cuando el servidor encuentra un fichero **.htaccess** (definido por la directiva **AccessFileName**) necesita conocer qué directivas hay declaradas en él, para poder activarlas.

El parámetro **"valor"** puede ser definido a **none** y en tal caso el servidor no leerá el fichero **.htaccess** puede ser definido a **All**, de forma que activa todas las directivas especificadas en **.htaccess**

El parámetro "valor" también puede ser definido a: **AuthConfig, FileInfo, Indexes, Limit, Options.**

- **UserDir** directorio:

  Habilita y define el nombre del directorio (**public_html** o se puede especificar otro) que será añadido al directorio HOME del usuario($HOME/public_html) donde cada usuario puede colocar documentos web públicos si se recibe una petición: **http://nombre-servidor/~usuario**.

- **DirectoyIndex** nombre-página-de-inicio …:

  Establece una lista con los nombres de ficheros típicos, usados como páginas de inicio de un directorio, cuando en una solicitud no se indique el nombre de la página a visualizar, se servirán cualquiera de las páginas de inicio que aquí se especifiquen por su orden (index.html default.html index.htm index.php index.php3 …). Por ejemplo: http://servidor/mis_paginas/

- **AccessFileName** fichero:

  El nombre del fichero a buscar en cada directorio para información de control de acceso, normalmente **.htaccess**

- **Alias** url directorio:

  Esta directiva permite que los documentos sean almacenados en un sistema de ficheros diferente al definido por la directiva DocumentRoot, con ella se definen los llamados **directorios virtuales.** Por ejemplo:

  Alias /error/ "/var/www/error/"

  define que cuando se haga una solicitud http://en la que se incluya la ruta /**error/**, el servidor buscará las páginas en el directorio: **/var/www/error** por ejemplo: (http://servidor/error/)

- **Redirect** url url-nueva

  Permite indicar que un documento web se encuentra localizado en otro directorio o en otro servidor.

  Redirect /curso /juegos

  Redirect /clases http://www.servidor.net/clases

- **ErrorDocument**. Especifica: o un mensaje corto de error para cada código de error, o dónde se encuentran las páginas de error, por defecto. Las páginas de error por defecto se encuentran en **/var/www/error/** Si ocurre un problema o error cuando un cliente intenta conectarse al Servidor Apache HTTP, la acción por defecto es mostrar el mensaje corto o una página de error ya definida en /var/www/error/, por ejemplo, 404 Not Found, o 403 Forbidden (sin permisos). También se puede redirigir un código de error a una página web que hayamos creado, por ejemplo, un archivo llamado 404.html, copiado a *DocumentRoot*/../error/404.html. y definir la directiva:

  ErrorDocument 404 /error/404.html

- **ErrorLog log/access_log**

  Define dónde se encuentra el archivo de registro de errores de apache

- **CustomLog log/access_log combined**

  Define dónde se encuentra el archivo de registro de solicitudes con éxito que ha tenido el servidor apache

- **ScriptAlias** url directorio:

  Tiene el mismo comportamiento que la directiva Alias, excepto que además define el directorio para que pueda contener scripts CGI (aportan dinamismo a

las páginas web).

Configuración de los Servidores Virtuales (VirtualHost).
Se explica en el punto nº 7.

4. Archivos log de apache

Por defecto, apache utiliza dos archivos de registro: **access_log** y **error_log** que están almacenados en la carpeta **/var/log/httpd**.

En el archivo **/var/log/httpd/access_log**, apache va registrando todos los accesos que los PCs hacen al servidor web y en cada línea de dicho archivo va almacenando la IP, la fecha y la hora, el comando HTTP enviado por el cliente, la url solicitada y la versión del navegador y el sistema operativo. Analizando este archivo podemos ver las veces que se ha descargado una página o un archivo, o las IPs más activas. Este archivo de registro es utilizado por los programas que presentan estadísticas de acceso al servidor web.

En el archivo **/var/log/httpd/error_log**, apache registra todas las incidencias o errores que se van produciendo. Ejemplo, cuando un cliente solicita una página inexistente o cuando un cliente intenta entrar en una carpeta prohibida o protegida. Si estamos configurando algo en apache (carpetas privadas, carpetas seguras, servidores web virtuales, alias, etc...) y no funciona, una buena idea es hacer pruebas y examinar el archivo error_log ya que nos puede dar pistas para encontrar la solución a nuestro problema.

5. Directorios virtuales. Directiva Alias.

Consiste en almacenar o alojar las páginas web en un sistema de fichero diferente al definido por la directiva DocumentRoot, por ejemplo en otro directorio, disco e incluso equipo diferente. Se define con la directiva **Alias**

Si, por ejemplo, quisiéramos alojar contenido web en un directorio localizado en **/var/contenidos/varios/** y que queremos visualizar como el directorio **/varios/** en Apache, lo primero será crear el directorio:

mkddir –p /var/contenidos/varios

y añadir un documento índice en el interior (index.html, index.php, etc)
Configuraremos la directiva Alias en el archivo de configuración de apache:

Alias /varios /var/contenidos/varios

Y la configuración del directorio virtual creado:

```
<Directory "/var/contenidos/varios">
Options Indexes Includes FollowSymLinks
AllowOverride all
</Directory>
```

En el ejemplo anterior, el parámetro **Indexes** indica que se deberá mostrar el índice de contenido del directorio.
El parámetro **FollowSymLinks** posibilita poder colocar enlaces simbólicos dentro del directorio los cuales se seguirán.
El parámetro **Includes** especifica que se permite la utilización de los SSI

(Server Side Includes) que posibilitan utilizar funciones como autenticación.
El parámetro **AllowOverride all** posibilita utilizar archivos .**htaccess**.

Reiniciamos o recargamos Apache y accedemos al directorio con cualquier navegador de red: ***http:/nombre-o-IP-del-servidor-web/varios/***

6. Acceso a páginas web personales.

Cada usuario del sistema puede disponer de un espacio web que se almacenará dentro de su carpeta home en una carpeta llamada 'public_html' (/home/usuario/public_html). Si dicha carpeta no existe, el propio usuario puede crearla y copiar dentro de ella su página web. Los permisos recomendados son 755 para que el 'grupo' y el 'resto' de usuarios tengan acceso de lectura y así se puedan visualizar las páginas.

Para acceder vía web a la página de un usuario, desde un navegador debemos acceder directamente con la dirección IP o el nombre del servidor a:

http://nombreo-IP-del-servidor/~login-usuario/

El caracter '**~**' se obtiene con Alt Gr + 4 sirve para indicar a apache que debe servir la página desde el home del usuario, dentro de la carpeta **public_html** donde se encontrarán sus páginas personales. Ejemplo, si hemos creado un usuario pepe y éste ha creado la carpeta /home/pepe/public_html y ha copiado en ella su página web, desde cualquier PC de la red podremos acceder a dicha carpeta yendo a la dirección http://ip-del-servidor/~pepe/. **~pepe** es sustituído por: **/home/pepe/public_html/** para la localización de la página (para que la página aparezca automáticamente, es necesario haber creado un archivo llamado index.html).

Esta característica se puede utilizar gracias al módulo **mod_userdir.so** de apache. En el archivo de configuración de apache tenemos que indicar con la cláusula: **UserDi**r y detrás el directorio del cual colgarán los documentos web del usuario, normalmente se deja:**public_html**

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

Podemos habiltar esta posibilidad solo para algunos usuarios (por ejemplo pepe y juan) configuramos, de esta manera:

```
UserDir disabled

UserDir enable  pepe juan
```

En este caso, sólo podrán configurar páginas personales en sus directorios home, los usuarios:pepe y juan, pues sólo ellos tienen activada esta posibilidad.

7. Hosts Virtuales.

Esta directiva define los parámetros para la configuración de diferentes sitios web alojados en un mismo equipo o servidor.

Un servidor Web puede simular diferentes sitios web utilizando diferentes nombres de hosts (alias) o diferentes IP's dentro del mismo equipo. En esta sección se define el comportamiento específico de cada servidor virtual alojado en el servidor, se pueden incluir prácticamente todas las directivas de apache.

- **<VirtualHost> </VirtualHost>**

  <VirtualHost> y </VirtualHost> se utilizan para agrupar un conjunto de directivas que se aplicarán solo a un host virtual. Dentro de este par de directivas se pueden incluir cualquier otra que haga referencia a este host particular.

- **NameVirtualHost**: (para hosts virtuales basados en nombres).

  Especifica la dirección IP a resolver cuando se utiliza un sistema basado en nombres de host virtuales.

- **ServerName**:

  Especifica el nombre de host que aparecerá en la cabecera de la petición.

Los beneficios de utilizar el soporte de host virtuales permite definir prácticamente un número de servidores ilimitado, de fácil configuración y que no requiere hardware adicional.

Es fácil mantener varios sitios web con diferentes nombres de dominio, tan solo hay que definir varios alias (CNAME en el DNS) o en el archivo **/etc/hosts** si no tenemos DNS, para ese host. En este ejemplo definimos 2 sitios web diferentes: **www.domain.tld** y **www.otherdomain.tld** para la mismo servidor (IP: **111.22.33.44**).

```
NameVirtualHost 111.22.33.44:80

<VirtualHost 111.22.33.44>
        ServerName www.domain.tld
        DocumentRoot /www/domain
</VirtualHost>

<VirtualHost 111.22.33.44>
        ServerName www.otherdomain.tld
        DocumentRoot /www/otherdomain
        ServerAdmin webmaster@otherdomain.tld
        ErrorLog /var/www/ otherdomain/logs/error_log
        CustomLog /var/www/ otherdomain/logs/access_log combined
</VirtualHost>
```

Con la configuración anterior, cualquier acceso con un navegador web a la dirección IP o un nombre distinto de los definidos en el bloque <VirtualHost> dará como resultado que Apache nos dé acceso al primero de los servidores virtuales definidos.

8. Autenticación y autorización. Acceso a páginas privadas

La autentificación se basa en el principio de que el cliente envía su nombre y su password al servidor, un módulo de Apache chequea si las credenciales son correctas, y si lo son devuelve la página solicitada. Si el usuario no tiene permitido el acceso o el password no es válido, el servidor Apache devuelve un **código de estado (Acceso no autorizado).**

Control de acceso a usuarios: Acceso Restringido con autenticación básica (nombre+ Password).

Para poder definir este tipo de autenticación, es necesario seguir dos pasos:

1. Crear un fichero que contenga los usuarios y los passwords.

   htpasswd -c /etc/httpd/conf/db_users (u otra ruta fuera de DocumentoRoot)

2. Incluir un fichero llamado **.htaccess** con las opciones de control de acceso, dentro del directorio que queremos proteger.
3. Definir, en las opciones **<Directory……>** del directorio a proteger, la opción **AllowOverride** con el valor (**all** o **AuthConfig**) ya que le informa al servidor de si ese directorio contiene o no un archivo **.htaccess** que debe leer. El valor por defecto para la directiva **AllowOverride** es **none (AllowOverride none**) en este caso el servidor apache no leerá el archivo .htaccess (si lo hubiera), el directorio no se protege y tendrá una acceso anónimo permitido.

Directivas de Configuración para el archivo .htaccess

El directorio o directorios a proteger deben contener un archivo **.htaccess.** Cuando el servidor tiene que acceder a una zona donde se encuentra un fichero .htaccess (definido por AccessFileName) comprobará las directivas definidas en él para otorgar o no el acceso a dicho directorio.

En el archivo **.htaccess** se incluirán las siguientes directivas:

- **AuthName**:

  Define un nombre para la zona protegida se visualizará cuando se intenta acceder al sitio protegido. Una vez que un usuario introduzca unas credenciales válidas para esta zona, cualquier otro recurso dentro de esta zona podrá ser accedido con las mismas credenciales.

- **AuthType**:

  Define el sistema de autentificación utilizado. Puede ser Basic o Digest. Este último método ofrece mayores características de seguridad al utilizar firmas MD5.

- **AuthUserFile**:

  Define la localización del fichero creado con el comando httpasswd. Se pueden

incluir más de una cláusula AuthUserFile para incluir más de un archivo de usuarios.

- **require**:

  Define qué nombres de usuario del fichero definido por la directiva anterior **(AuthUserFile)** tienen derechos de acceso. **valid-user** sería un caso especial refiriéndose a cualquier usuario del fichero: **require valid-user**¸Se puede especificar una lista de usuarios de los incluidos en el archivo de usuarios autorizados: **require usu1 usu2 usu5 …** (en este caso sólo dichos usuarios tendrán acceso al directorio web).

- **AuthGroupFile**:

  Define la localización del fichero de grupos que contendrá la lista de grupos de usuarios que tienen autorización de acceso al directorio: **AuthgroupFile /etc/httpd/.ht_grupos.** El formato de este archivo será, por cada grupo a incluir: **nombre-de-grupo: lista de usuarios** del grupo separados por un espacio en blanco, por ejemplo:

  informatica: alumno pepe juan maria

  secretaria: pepe maria ana

  Se pueden incluir más de una cláusula AuthGroupFile para incluir más de un archivo de grupos.

- **require group**:

  Define qué nombres de grupos de los incluidos en el fichero definido por la directiva anterior tienen derechos de acceso. Por ejemplo: r**equire group informatica**

Ejemplo: queremos proteger el directorio **/var/www/privado/**

1. Crearemos el archivo de usuarios con permiso de acceso: **httpasswd –c /etc/httpd/conf/.ht_usuarios usu1**

Repetimos el comando para añdir más usuarios (sin la opción **–c**):

httpasswd /etc/httpd/conf/.ht_usuarios usu2

2. Crearemos el archivo **.htaccess**, dentro de dicho directorio, tendrá el siguiente contenido:

AuthName “Sitio Protegido”

AuthType Basic

AuthUserFile /etc/httpd/conf/.ht_usuarios

require valid-user

También podríamos especificar, en lugar de **require valid-user**, una lista de usuarios: **require usu1 usu4 usu6** , en este caso, sólo ellos tendrán permiso de acceso al directorio.
3. Configuramos el directorio, en el archivo de configuración de apache:

Alias /privado /var/www/privado porque el directorio “privado” está fuera del raíz de documentos

<Directory "/var/www/privado">
Options Indexes, Includes
AllowOverride All (que se activen las opciones incluidas en el archivo .htaccess)
Order allow,deny
Allow from all
</Directory>

Control de acceso a grupos:
De la misma manera que con los usuarios, podemos especificar grupos de usuarios con la cláusula: **AuthGroupFile /ruta/al/archivodegrupos**

Por ejemplo: **AuthGroupFile /etc/httpd/.ht_grupos** crearemos el archivo **.ht_grupos** que tendrá el siguiente formato :por cada grupo su nombre y la lista de usuarios que pertenecen a dicho grupo, separados por un espacio en blanco: **nombre-grupo1: usuario1 usuario2**,….

Por ejemplo**:**
informatica: usu1 usu2….
direccion: usu1 usu5….
secretaria: usu3 usu4….

Si queremos que sólo unos cuantos grupos de los que están incluidos en dicho archivo, tengan permisos de acceso, al directorio, podemos especificarlos con: **require group nombre-grupo**, por ejemplo: **require group informatica** en este caso sólo el grupo "informática" tendrá permiso de acceso al directorio. El archivo **.htacces** quedaría así:
AuthName "Solo para usuarios de informática"

AuthType Basic

AuthUserFile /etc/httpd/conf/.ht_usuarios

AuthGroupFile /etc/httpd/.ht_grupos

require group informatica