Servidor http Apache

0.- Terminología

Antes de explicar el tema de configuración y administración del servidor http se hará la descripción de algunos términos necesarios para el mejor entendimiento del resto del documento.

Servidor : Programa ejecutado en una máquina que responde a solicitudes de otros programas llamados clientes.

Cliente : Programa que solicita información al servidor para mostrársela al usuario.
Servidor web : Recibe solicitudes html de los clientes web. Ej. Apache

Cliente web : Solicita información al servidor web. Se les suele llamar navegadores. Ej. Mozilla.

Direccion IP : Numero de cuatro dígitos separados por puntos (x.x.x.x) para identificar a un equipo en una red TCP/IP lo cual permite su localización.

Nombres de dominio : Usados para facilitar al usuario recordar sitios web a los que pretende acceder.

Resolución de nombres : Traduce direcciones IP a un nombre de dominio o viceversa ya que para los usuarios es mas fácil de recordar nombres.

> www.ulpgc.es → nombre de dominio
> 192.168.0.1 → direccion ip

DNS (Domain Name Server): Máquina servidora de traducción de nombres. Es necesario dar de alta el nombre de dominio registrado en un servidor DNS.

Puerto : Para diferenciar las peticiones de los multiples clientes además de la dirección IP se usa un número de puerto entre 1 y 65535. Cada servicio que corre en una maquina tiene asignado uno o más de esos números. P ej. Web es el 80. Hay usos predeterminados para estos números. Si no se especifica se entiende el que se usa por defecto. El fichero /etc/services muestra los puertos predeterminados para cada servicio.

Socket : Elemento de programación que permite a dos máquinas comunicarse a través de una red, mediante el uso de la IP de origen, la IP de destino y el número de puerto. Se crea un socket cuando hay una conexión entre el cliente y el servidor.

Protocolo : lenguaje por el cual se comunican el cliente y servidor. http. (HiperText Markup Language)

http://www.ulpgc.es:80
http → protocolo se sobre entiende si no se pone en el cliente
www.ulpgc.es → nombre de una maquina que tiene una ip equivalente
80 → numero de puerto. Si no se usa se sobreentiende

fpt://ftp.ulpg.es
ftp → protocolo
ftp.ulpgc.es → nombre de maquina

Apache : servidor web más utilizado mundialmente. Por defecto lo traen instalado en todas las distribuciones Linux. También existe para otras plataformas incluso Windows. Su funcionamiento básico es ejecutando un proceso padre y tantos procesos hijos como peticiones reciba para atender a cada cliente.

El demonio (Proceso padre) de apache que estará escuchando por el puerto 80 que por defecto se llama httpd y para poder usarlo necesita estar en servicio. Esto se puede hacer bien con el comando service http Start/stop/restart o bien desde el directorio de inicio de demonios en /etc/init.d/httpd Start/stop/restart. (Realmente es un script que permite trabajar con el servidor).

También se puede decir al sistema que el demonio se arranque siempre que se inicie el servidor con el comando setup. En la opción servicios marcamos httpd.

1.- Instalación

Para empezar a utilizar, configurar y administrar Apache es necesario tenerlo instalado en nuestra máquina. Por defecto viene instalado en la mayoría de las distribuciones Linux, pero por si no fuera asi o estuvieramos utilizando cualquier otro sistema operativo para disponer de el en nuestra máquina debemos seguir los siguientes pasos.

- Descargar software de www.apache.org **(download)**
- Descomprimirlo con el comando **tar –zvxf apache_x.x.x.tar.gz** *(esto generará un directorio llamado apachex.x.x donde estarán todos los archivos de la distribución y x.x.x será la versión de apache que se recomienda tener actualizada para evitar vulnerabilidades. Si bajamos en formato rpm se realiza la descompresión e instalación a la vez con el comando rpm –ivh apache.org. Para saber si apache ya esta instalado podemos usar rpm –q apache.org)*
- **./configure --prefix = ruta de instalación** *(Mediante este script se puede modificar los distintos aspectos de la configuración a nuestro gusto, pero fundamentalmente lo que más se usa es darle la ruta donde se realizará la instalación. Con esto genera los ficheros make y makefile que configuran la instalación. Para una ayuda más detallada en la configuración podemos poner ./configure –help. Es posible modificar directamente un script de configuración configuration.tmpl, y se recomienda hacer una copia antes)*
- **make** *(Ayuda a la compilación recibiendo como entrada el fuente, las bibliotecas de librerias y reglas de ensamblado del fichero makefile)*
- **make install** (esto realiza la precompilación del codigo fuente con las opciones dadas en el script configure.

Este proceso de instalación de software suele ser estándar para la mayoría del software comprimido que queramos instalar en una máquina Linux. En el caso de Linux RedHat o Mandrake el proceso es incluso más fácil descargando archivos con extensión rpm.
Para la instalación de Apache en formato rpm lo único necesario es :

- **rpm –q apache** (para verificar si está instalado el paquete o no)
- **rpm –ivh apache_x.x.x.rpm** (para realizar el chequeo de dependencias por si necesitara algún otro paquete y realizar la instalación automáticamente)

A partir de este momento ya es posible lanzar/detener/reiniciar el servidor para comprobar su funcionamiento con el comando

- /ruta apache/sbin/apache Start (ejecuta el demonio httpd con las opciones de configuración preparadas por defecto), o bien en redhat con service httpd Start.

Una vez instalado apache podemos probar su funcionamiento con cualquier cliente web (navegador) en el caso de linux con mozilla y en windows con internet explorer por ejemplo. Colocamos en la barra de direcciones localhost o el nombre o ip de la máquina del servidor y debe aparecer el test de Apache lo que indica que la instalación es correcta.

2.- Configuración

El fichero de configuración es de Apache está situado en /etc/httpd/conf/httpd.conf Este fichero, aunque en inglés está muy bien documentado con comentarios de las diferentes opciones y significado de directivas. Cada vez que se haga en el fichero una modificación es necesario reiniciar el servicio para que estos cambios tengan efecto.

(Es posible permitir a cada usuario que configure su sitio web usando el fichero .htaccess, que tiene el mismo formato que httpd.conf y debe estar dentro del directorio donde se quieren realizar las modificaciones usando directivas contenedoras.)

El fichero de configuración httpd.conf se puede dividir en varias secciónes y todo lo que se encuentre detrás del símbolo # se considerará un comentario.

- *Sección 1* : Entorno global. Parte del fichero donde estan las rutas a otros ficheros de configuración y se describe el funcionamiento general del servidor.
- *Sección 2*: Entorno servidor principal. Aquí se describe la configuracion que no atiende a peticiones de los servidores virtuales. Comportamiento predeterminado del servidor.
- *Sección 3*: Servidores virtuales que se pueden configurar para trabajar bajo el mismo programa.

Apache usa lo que se conoce como directivas que son variables almacenadas en el archivo de texto de configuración para alterar y controlar el funcionamiento de Apache en tiempo de ejecución según sus valores y después de haber reiniciado el proceso servidor. Hay multitud de directivas y no es necesario conocerlas todas para un buen uso del servidor. Apache diferencia las tareas a realizar mediante el uso de modulos, que no son mas (DSO Objetos dinámicos compartidos) que se pueden añadir o quitar del servidor modificando ciertas directivas y darle la funcionalidad que nosotros deseemos. Por lo tanto cargamos los modulos que deseemos unicamente creando un servidor mas eficiente.

2.1.- Directivas de Entorno global

ServerType → Indica como será el tipo de respuesta del servidor. Sus posibles valores son:

- *Inetd.* Se ejecuta cuando hay una petición y es el demonio inetd el encargado de iniciar y matar el proceso httpd.
- *Standalone.* Esta siempre ejecutándose un proceso específico httpd y este genera hijos para las distintas conexiones de los diferentes clientes.

ServerRoot → Directorio en el que se monta la raiz del servidor, es decir de donde parten los ficheros de configuración del servidor

- Ej. ServerRoot /etc/httpd

TimeOut → Para evitar que se atasque la red Apache no mantendrá las conexiones del cliente activas indefinidamente. Número de segundos desde que se recibe la petición hasta que se envia la señal de timeout

- Ej. Timeout 300

MaxClients → Limita el número total de servidores que se ejecutan simultáneamente, o dicho de otra forma limita el maximo numero de clientes que se pueden conectar simultáneamente. (Si se supera se bloquean los slientes)

- MaxClients 150

Listen → Permite a Apache escuchar otra dirección y/o puertos añadidos además de la dirección o puertos por defecto. Puede haber varias directivas listen. Incluso si hubiera varias tarjetas se puede indicar que una ip determinada escuche por un puerto determinado.

- Ej. Listen 1.2.3.4:8080
- Listen 7000

BindAddress → permite el soporte de servidores virtuales. Se utiliza para indicar al servidor que direcciones IP se deben escuchar. Se puede incluir una IP o un nombre de dominio.

- BindAddress *

LoadModule → Carga un modulo para aportar mayor funcionalidad a apache. Para saber que modulos hay cargados ejecutar http -1. LoadModule nombre_modulo ruta.

- Ej. LoadModule Env_module libexec/mod_env.so

2.2.- Directivas de Configuración del servidor Principal

Port → Puerto por el que escuchará el servidor principal. Similar a listen pero solo puede haber una en todo el fichero de configuración. Usar puertos no específicos ya usados.

- Ej. Port 80

User, Group → Nombre del usuario o grupo que puede lanzar la ejecución de httpd. Como medida de seguridad no debe aparecer ningún grupo o usuario o el del propio Apache, asi solo se puede acceder via web a los documentos. Luego le damos permiso a los documentos al usuario apache y se los quitamos al resto.

- Ej User nobody
- Group nogroup

ServerAdmin → Dirección a la que enviar los problemas que puedan aparecer y deban ser enviadas por correo electrónico al responsable de la administración del servidor web. Esta dirección aparecerá en algunas páginas generadas por el servidor como pueden ser las páginas de error.

- ServerAdmin webmaster@dis.ulpgc.es

ServerName → Establece el nombre de servidor que se envia de vuelta a los clientes desde el propio servidor. No es seguro enviar el nombre real de la máquina.

- ServerName ulpgc

ServerSignature on/off/email : Se usa para que cuando se acceda a una pagina inexistente el servidor muestre una pagina de error con un mensaje donde aparece el nombre de la maquina y la versión de apache usada.

DocumentRoot → Directorio en el que se colocan los documentos web que el servidor pondrá disponibles a los clientes.

- DocumentRoot /var/www/html *(esta es la ruta por defecto. Directorio raiz de documentos del servidor web)*

Directivas contenedoras

A cada directorio que Apache tiene acceso, se debe crear una estructura que lo habilite. Son lo que se conocen como directivas de contenedor.
Se pueden usar tres formas para limitar el ambito con directivas de contenedor.

1) Directory, DirectoryMatch y el archivo .htaccess

<Directory /rutaapche/directorio>
Todo lo que este aquí se aplica solo al directorio. Es recursivo si hay directorios
dentro del directorio
Options opciones
AllowOverride opciones
Order opciones
Allow opciones
</Directory>

<DirectoryMatch "/rutaapche/directorio[1-3]">
Todo lo que este aquí se aplica al/los directorio/s implicados. El argumento en
este caso es una expresión regular. Es recursivo si hay directorios dentro del
directorio
Options opciones
AllowOverride opciones
Order opciones
Allow opciones
</DirectoryMatch>

Es posible permitir a cada usuario que configure su sitio web usando el fichero .htaccess, que tiene el mismo formato que httpd.conf y debe estar dentro del directorio donde se quieren realizar las modificaciones usando directivas contenedoras. Para que apache sepa de la existencia de estos archivos y los busque se le indicará mediante la opción Allowoverride cuyos argumentos se explicarán posteriormente. También será posible cambiar el nombre del fichero .htaccess por otro mediante la directiva AccessFileName.

2) Limitar el ambito de un directorio URL mediante <Location> y <LocationMatch> *(similares a Directory y DirectoryMatch respectivamente.)*

<Location directorio>
Igual que el contenedor directory pero aquí se usa direccion relativa de
directorio.
</Location>

<LocationMatch "directorio[1-3]">
Igual que el contenedor directoryMatch pero aquí se usa direccion relativa de
directorio.
</LocationMatch>

3) Limitar el ambito a ficheros mediante <Files> y <FilesMatch>

<Files archivo>
#Todo lo que metamos aquí hace referencia al fichero o ficheros. Es posible usar
comodines. Suele ir dentro del contenedor directory.
</Files>

Argumentos para Options. Va dentro de una directiva contenedora. Permite definir las caracteristicas disponibles para un directorio determinado.
None : Ninguna
All : permite Todas las opciones excepto Multiviews
Indexes : Permite que se visualicen indices. Permite ver el contenido del directorio si no hay pagina de inicio. Lo cual se considera vulnerable.
Includes : Permite incluir determinadas rutas o ficheros
FollowSymLinks : Permite el salto a través de enlaces. Permite seguir los enlaces simbolicos entre este directorio y otro donde este el enlace simbolico. (accesos directos a www).
SymLinksIfOwnerMatch : Sigue enlaces simbólicos en caso de que el propietario del archivo o directorio de destino sea el mismo que el propietario del enlace.
ExecCGI : Permite la ejecución de CGIs.
IncludesNOEXEC : permite includes pero descativa comando exec.
Multiviews : Permite la vista del directorio desde varios clientes al mismo tiempo.

Argumentos para AllowOverride son:
None : No permite el uso de directivas y desactiva la búsqueda del fichero .htaccess
All : Permite Todas las directivas y activa la busqueda
Options : Permite Options, XbitHack
FileInfo : Información de ficheros. Permite el uso de AddEncoding, AddType, AddLanguage…
AuthConfig : Permite el uso de AuthUseFile, Require, AuthType, AuthName…
Limit : Limite de ficheros a mostrar permitiendo allow, deny, order.
Indexes : Permite el uso de IndexOptions, DirectoryIndex, AddIconType…

Order, Allow, deny : Se usan conjuntamente para proporcionar un mecanismo de control de acceso.

Opciones para Order.
Allow : A los que se le Permite acceder
Deny : A los que se les deniega el acceso.
Ej. Order Allow, deny → permite todo lo que no este en denegaciones.
Order deny, allow → deniega todo lo que no esta permitido.

Argumentos para Allow o deny son:
From : De donde se permiten las peticiones. Su puede usar allow from all. O direcciones IP's y nombres de dominio.
Ej allow from all
Deny from apache.org 192.168.0.

Las opciones pueden ser aditivas, basta separarlas con un espacio.
Ej . Options Indexes FollowSymLinks

La ultima opcion eliminara a la anterior. Si lo que se quiere es anadir o eliminar opciones se puede usar – o +. Ej. Options + Indexes

AccessFileName → Nombre del fichero que se usara para la configuración personal de los sitios en lugar de .htaccess. El fichero debe ser oculto. Para permitir que se pueda leer el fichero en httpd.conf es necesario colocar la opcion allowoverride con valor all que permite el uso de las directivas y activa el uso de fichero de configuración independientes. La opcion none de Allowoverride no permite las directivas y desactiva la búsqueda de ficheros de configuracion independientes.

-Ej. AccessFileName .pepeconf

<IfModule> → Ejecuta de forma condicional las directivas si el servidor puede acceder al módulo especificado.
Ej. <IfModule mod_userdir.c>
Userdir pepe
</IfModule>

UserDir enable/disable→ Nombre de directorio que se añade como directorio de usuario cuando se recibe una petición del tipo ~usuario. Permite que los usuarios de la maquina tengan sus propias paginas personales. http://servidor/~pepe → pagina personal de pepe. Pueden haber varias directivas userdir.
Ej. UserDir public_html → las paginas del usuario pepe estaran en el directorio /home/pepe/public_html. El propietario del directorio public_html debera ser apache asi nadie vera los documentos salvo via web.
Ej. UserDir disable juan antonio → Deshabilita las paginas personales para estos dos usuarios.

DirectoryIndex → Indica el nombre del fichero que se toma como indice del directorio por defecto cuando el usuario no indica el nombre en el cliente web. (pueden haber varios y entonces los buscara por orden de entrada)
Ej. DirectoryIndex index.html home.html default.php

AddModule → Permite el uso de módulos compilados pero no activos.
Ej. AddModule mod_auth_dbm.c

ErrorLog → Contiene la ruta (absoluta o relativa al path de apache) del fiechero de log de errores. Registra lo que sucede con apache, errores y accesos de clientes. Si no se especifica en un virtual host los mensajes de error de todos los servidores virtuales se almacenarán en el mismo fichero. Si no cada servidor virtual tendra el suyo propio.
Ej. ErrorLog /var/log/error_log

LogLevel → Los que registra apache esta dividido en niveles (hasta 8), registrando mas o menos errores dependiendo de la importancia. controla el numero de mensajes que se guardarán en fichero de log de errores. Los valores posibles son:

- Debug → controla todo
- info
- notice
- warn
- error
- crit
- alert
- emerg → controla solo errores graves
- Ej LogLevel warn

CustomLog → Se usa para hacer y personalizar estadisticas de acceso al servidor.

- Ej. CustomLog logs/acces_log

LogFormat → Formato que tendran las lineas del fichero de log.
%h ip del cliente
%t fecha y hora
%b bytes enviados
%p puerto usado
%T segundos usados para efectuar la transmisión

%u nombre de usuario si hay que poner contraseña para entrar a la pagina
%U URL del cliente
%h servidor remoto que efectua la operación

Ej. Logformat “Ip del cliente: %h; hora %t” miformato
Customlog logs/access_log miformato

Aliases (Redirección interna)→ Se pueden añadir tantos alias como sean necesarios, sin limites, con el formato siguiente. Redirige de un lugar a otro dentro del propio servidor . Se suelen usar para paginas personales de los usuarios. Alias nombre_falso nombre_real

- Ej. Alias /icons/ “/var/lib/apache/icons/”
- Ej. Alias pepe /home/pepe/web → http://servidor/pepe

Redirect → Redirige a un directorio en otro servidor.

- E. Redirect /directorio1 http://www.ulpgc.es

ScriptAlias → Controla que directorios contienen archivos de comandos del servidor. Los scriptsalias son esencialmente lo mismo que los alias excepto que los documentos en el directorio real se tratan como aplicaciones y son ejecutadas por el servidor cuando se solicita en lugar de ser envitados al cliente como documentos.

- Ej. ScriptAlias /cgi-bin/ “/var/lib/apache/cgi-bin/”

Redirect → Permite que los clientes sepan que documentos residen en el servidor y cuales no. Esto permite comunicar a los clientes donde buscar el documento reubicado. El formato es : Redirect URL-vieja URL-nueva

Los módulos de Apache

La idea de modularizar apache viene de la necesidad de incorporar funciones al servidor de manera sencilla yeficaz. Los modulos permiten integración de bases de datos, búsquedas de datos, autentificación.

Servidores Virtuales

Ventajas

- Versatilidad :
- Precio: Solo un sistema para varios servidores
- Configuración : Solo una configuración del sistema para todos los servidores
- Actualizaciones : Solo una unica vez

Desventajas

- Fragilidad : Un fallo en el sistema hará que caigan todos los servidores
- Configuración : Una configuración erronea puede hacer que no funciones ninguno de los restantes.
- Actualizaciones : Si es necesario detener el sistema para actualizarlo se pararan todos los servidores.
- Seguridad : Todos los servidores se ven afectados ante problemas de seguridad.

Normalmente el factor economico hace decantar por un sistema de servidores virtuales. Aun que siempre quedará a criterio del administrador la ultima elección en función de las ventajas e inconvenientes comentados.

Servidores virtuales por dirección IP

Apache permite albergar diferentes sitios con lo que se llaman servidores virtuales. Por ejemplo que dos sitios diferentes sean respondidos por el mismo servidor.

Es posible configurar los servidores virtuales asignando a cada uno de ellos una dirección IP, que a su vez podemos tener configuradas una a cada adaptador (una máquina con varias tarjetas de red) o bien asignar a un mismo adaptador varias direcciones ip. En primer lugar es fundamental tener dadas de alta las direcciones ip de los servidores instalados en el dns correspondiente.

Ejemplo

Dominio www.web1.com → 1.2.3.4
www.web2.com → 1.2.3.4

Se modificaría la base de datos del DNS. Hay que decirle al servidor DNS que todos los nombres hacen referencia a la misma IP.

www.web1.com IN A 1.2.3.4
www.web2.com IN A 1.2.3.4

Y modificar la configuracion de apache en el fichero httpd.conf de la siguiente manera.

Namevirtualhost :Indica la maquina que va a disponer de los dominios virtuales.
EJ.
Namevirtualhost 1.2.3.4
<virtualhost 1.2.3.4>
#Una seccion virtualhost para cada uno de los directorios virtuales
Servername www.ferreteriamanolo.com
ServerAlias www.ferreteriamanolo.org
ServerAlias www.ferreteriamanolo.es

DocumentRoot /var/www/ferreteriamanolo
...
...
</virtualhost>

<virtualhost 1.2.3.4>
Servername www.viveresmartin.com
DocumentRoot /var/www/viveresmartin
...
...
</virtualhost>

<virtualhost _default_>
Si intenta entrar a nuestro servidor por un nombre que no existe
</Virtualhost>

Servidores virtuales basados en nombre
Igualmente, a como ocurre con los servidores virtuales, es posible disponer de una unica direccion ip y asignar a esta varios dominios, haciendo que a cada dominio se le asigne un servidor virtual.

Modificar la base de datos DNS

www.web1.com IN CNAME web.subdominio.com
www.web2.com IN CNAME web.subdominio.com

Es possible configurar los servidores virtuales para que estos sean ejecutados por un unico demonio que los atiende a todos, o bien por un conjunto de demonios que se asigne a cada uno de los servidores virtuales.

Contenido Dinámico
SSI → Server side includes. Extensión .shtml
CGI → Common Gateway Interface. Programa escrito en cualquier lenguaje soportado por el sistema operativo que genera una salida. Apache ejecuta el programa y el resultado es lo que devuelve a los clientes. Si configuramos apache para leer cgi mostrara el resultado sino muestra el contenido del fichero. Hay que darle al fichero permiso de ejecución.

ScriptAlias /cgi-bin /var/www./gci-bin → Alias para directorios con permiso de ejecución de scripts.

Ej. Programa cgi.

```
#!/bin/bash
echo
echo hola
echo
echo
cat /etc/passwd
```

Guardar como hola.cgi

Codigos de estado http
El mecanismo de los codigos de estado http funciona simplemente devolviendo un codigo de 3 cifras que el cliente interpreta y responde en consecuencia. Ademas, el servidor proporciona junto al codigo de estado un breve mensaje con una descripción. Ej. http/1.1 404 Not Found → Error fichero no encontrado

Según la especificación http 1.1 existen cinco tipos de codigos de error

Informativos	100-109
Petición Correcta	200-299
Redirección Petición	300-399
Petición Incorrecta	400-499
Error en el servidor	500-599

Informativos

Son los que el cliente web recibe como mera información de manera que no tiene que interpretar nada ni responder a nada, simplemente darse por enterado.

Codigo	Mensaje	Descripción
100	Continue	Indica que puede recibir la siguiente petición
101	Switching Protocols	Indica que existe otro codigo mejor que el que utiliza actualmente. Esta funcion se usa para hacer upgrades de protocolo.

Petición correcta

Codigos que envia el servidor indicando al cliente la recepcion y aceptación de la peticion

Codigo	Mensaje	Descripción
200	OK	Acepta la petición y devuelve el documento solicitado.
201	Created	El servidor crea el URI solicitado en la cabecera de la petición del cliente.
202	Accepted	La petición del cliente se ha aceptado, pero no ha sido procesada aun o no ha finalizado su proceso.
203	Non-Authoritative Information	La información de la cabecera de la petición no la ha generado el servidor, ha sido copiada de otro servidor.
204	No Content	Petición completada. No será necesario enviar más información para que el cliente tenga el documento completo.
205	Reset Content	Hay que reiniciar el documento actual, es util cuando hay que borrar un formulario.
206	Partial Content	El documento ha sido enviado parcialmente al cliente. Junto con el codigo se indica el segmento de datos enviado.

Redirección de petición
Codigos que se envian al cliente para indicarle que la accion solicitada requiere mas acciones para completarse.

Codigo	Mensaje	Descripción
300	Multiple Choises	Se hace una petición que en realidad son varios documentos. El servidor puede enviar información sobre cada uno de los documentos de manera que el cliente pueda seleccionar cual de ellos es el que esta buscando.
301	Moved Permanently	El cliente solicita un documento que ha cambiado temporalmente de direccion. A partir de la recepcion de este codigo el cliente hara las redirecciones automáticamente en futuras peticiones.
303	See Other	El cliente pide un documento que ya no esta en la URL solicitada. El cliente debera utilizar el metodo GET para recuperarla.
304	Not Modified	Se indica que el documento no ha sido modificado desde la ultima visita del cliente. Se trata de utilizar la copia que reside en la cache del cliente en lugar de devolver el documento.
305	Use Proxy	El cliente debe utilizar el servidor Proxy que aparece en la cabecera de la petición, el documento se devuelve a través del servidor Proxy.

Petición incompleta

Son codigos que indicaran al cliente que debe enviar mas información para completar la petición.

Codigo	Mensaje	Descripción
400	Bad Request	Es un error de sintaxis en la cabecera de la petición por parte del cliente
401	Unauthorized	Es necesaria la autentificación del cliente. El servidor devuelve una cabecera para indicar la autentificación y el alcance de esta.
402	Payment Required	Uso. Futuro. Se usará para indicar que se debe hacer un pago anterior a la obtención del documento.
403	Forbidden	El cliente no tiene acceso a la fuente solicitada.
404	Not Found	Se solicita un documento que no es posible encontrar en el servidor.
405	Method Not Allowed	El metodo usado para la petición no es valido
406	Not Aceptable	El documento solicitado no esta en un formato que el cliente pueda reconocer.
407	Proxy Authentification Required	El cliente no se ha autentificado en el servidor proxy.
408	Request Time-Out	La petición no ha podido ser atentida en su tiempo establecido, por lo que deberá repetir la operación.
409	Conflict	La petición de documento solicitada por el cliente entra en conflicto con otra de otro cliente.
410	Gone	El documento solicitado ha sido eliminado del servidor.
411	Length Required	El cliente debe suministrar una cabecera Content-type para completar la petición
412	Precondition Failed	Es posible hacer peticiones de documentos que dependan de una o mas condiciones. El servidor usara esta misma cabecera para devolver la condicion generada indicando cual de ellas es falsa.
413	Request Entity too Large	La petición tiene un cuerpo con una longitud excesiva. El servidor cierra la conexión para impedir la entrada de la petición.
414	Request-URI too long	El servidor no procesará la petición porque la URI es demasiado larga.
415	Unsupoorted Media Type	El servidor no procesará la petición porque la URI es demasiado larga.

Errores en el servidor
Son codigos que el servidor estando activo (de otro modo no seria capaz de devolver ni siquiera los codigos de error) devuelve para indicar la causa por la que no puede devolver un documento o no puede finalizar la petición.

Codigo	Mensaje	Descripción
500	Internal Server Error	Hay un error en la configuración del servidor o de un programa ligado a este.
501	Not Implemented	El servidor no dispone de recursos suficientes como para atender a la petición o para completarla.
502	Bad Gateway	Se ha producido un error en el servidor Proxy o se ha obtenido una respuesta no valida del servidor.
503	Service Unavailable	El servicio no esta disponible.
504	Gateway time-out	No se ha podido establecer una conexión con la puerta de enlace.
505	http versión not supported	La versión del protocolo http que utiliza el cliente no se admite.

Instalacion y configuracion de apache con mysql y php.

Una de las caracteristicas mas atrayentes de un servidor web reside en la posibilidad de tener contenidos dinamicos (posibilidad que nos brinda PERL o PHP) pero si ademas se puede acceder a estas paginas con contenido dinamico a traves de un servicio de acceso a bases de datos a traves de ordenes SQL mucho mejor.

El metodo de trabajo es sencillo de explicar y comprender y se resume en concepto de funcionamiento con el siguiente esquema.

El funcionamiento se ve en los siguientes pasos:

1. El cliente web hace una petición al servidor Apache (protocolo http y puerto 80).
2. El servidor Apache reconoce la petición y comprueba que hay un archivo de comandos PHP.
3. Pasa la ejecución a mod_php del script.
4. El archivo de comandos php incluye una serie de ordenes de acceso a la base de datos por lo que pasa la consulta al motor de la base de datos.
5. La base de datos ejecuta la consulta, y los resultados de esta se devuelven al arvhivo de comandos php que espera los datos devueltos de la consulta.
6. El archivo de comandos php con los datos de la consulta montara los datos correspondientes sobre el documento html solicitado por el cliente.
7. Finalmente el servidor apache con el documentos html envia el documento al cliente.

Instalación de mysql

Descargar la aplicación de www.mysql.com
Descomprimirlo con tar –xzvf mysql-xx.xx.xx.tar.gz
Una vez descomprimido accedemos al directorio creado mysql-xx.xx.xx

Definimos antes de configurar la ruta propicia para que la compilación se haga sobre un directorio que sea logico y funcional encontrar el motor de la base de datos.
Configure –prefix=/usr/local/mysql

Ejecutamos el clasico make que configurara los archivos de comandos de instalación
Make
Make install

En este punto, tenemos un directorio /usr/local/mysql-xx.xx.xx sobre el que crearemos una ruta mas accesible como /usr/local/mysql

Ln –s /usr/local/mysql-xx.xx.xx /usr/local/mysql

Una vez finalizada la instalación será necesario que creemos las tablas de sistema mysql y definir los permisos de supervisor con la siguiente secuencia de ordenes.

Scripts/mysql_install_db
Cd/usr/mysql/bin
./safe_mysqld &
./mysqladmin –u root password 'nueva-password'

Para asegurarnos que mysql se ha instalado correctamente podemos ejecutar la siguiente orden
…/mysqlshow –p

que nos devuelve un listado de las bases de datos de sistema que hay disponibles en el motor de mysql, que por defecto solo contiene la tabla de sistema: mysql, y otra llamada test.

Si entramos en mysql deberemos hacerlo dando un nombre de usuario (que debe ser el administrador o root, al menos hasta que este configurada adecuadamente)

Mysql –u root –p

Una vez dentro del sistema de mysql mostraremos las bases de datos (recordemos que solamente habra dos, la de sistema mysql y una de prueba test)
Ademas crearemos otra base de datos llamada pruebaweb y a continuación seleccionaremos esa base de datos para crear una tabla.

Show databases;
Create database pruebaweb;
Use pruebaweb;
Create table web1 (numero int(3) not null auto_increment,
Titulo char(30) not null, unique (numero), primary key (numero));

Comprobamos que se ha creado la tabla

Show tables; ó mysqlshow ejmplobd tabla

Introducimos algun registro

Insert into web1 (titulo) values('registro de prueba N-1);

Insert into web1 (titulo) values(‘registro de prueba numero dos);
Insert into web1 (titulo) values(‘registro de prueba N-3);

Comprobamos que los datos esten en la tabla

Select * from web1;

Borramos el registro 2

Delete from web1 where numero=2;

Es mejor realizar todas las operaciones en un unico fichero que podamos modificar con mayor facilidad y luego pasarlo como parametro en una unica linea de la siguiente forma:
Mysql –u root ejemplodb –p < backup.sql

Instalación de php

Descargar php de la pagina www.php.org

Lo primero que debemos tener en cuenta es que php debe estar preconfigurado para funcionar con apache, lo cual implica que apache ya este instalado. (simplemente instalado no es necesario compilarlo), ya que php buscara el fichero de configuracion de apache sin modificaciones, y puesto que la ejecución de make para preparar la compilación de apache lo modifica, tendremos que asegurarnos de esta manera de que los ficheros no han sido modificados.

Descomprimimos php : tar –zxvf php-xx.xx.xx.tar.gz

Entramos en el directorio creado de php

Podemos compilar php de forma estatica o dinamica

Compilación estatica.
Preparamos el archivo de comandos de configuración “configure” correspondiente con los siguientes parametros

./configure –with-mysql=/dirmysql –with-apache=/dirapache –enable-track-vars \
--prefix=/dirphp

compilamos php
make
make install

En este momento php ya esta instalado pero apache debe saber que php esta disponible por lo que debemos modificar su configuración y volver a compilarlo.

Cd /dirapache
./configure –prefix=/dirapache –activate-module=src/modules/php4/libphp4.a \
--enable-module=php4

make
make install

finalmente copiaremos el fichero de inicialización al lugar donde los buscara el programa posteriormente

cp php.ini-dist /usr/local/lib/php.ini

Compilación dinamica

Igualmente preparamos el archivo para la instalacion y configuracion

Cd /dirphp
./configure –with-mysql=/dirmysql –with-apache=/dirapache –enable-track-vars \
with-apxs=/usr/local/apache/bin/apxs --prefix=/dirphp

make
make install

En este caso ya esta todo finalizado, no sera necesario volver a compiler apache como en la compilación estatica porque la nueva librería se lanza con la ejecución de los demonios (httpd) de apache. Este metodo de compilación nos aporta entre otras ventajas la de no tener que recomponer el servidor completamente cada vez que deseemos actualizar nuestro php.

Descomentar las lineas de httpd.conf

LoadModule php4_module modules/libphp4.so
AddModule mod_php4.c
…
AddType application/x-httpd-php4 .php4 .pthml .php .php3
AddType aplicacion/-x-httpd-php4-source .phps

Descomentar la linea del fichero php.ini
Extensión=mysql.so

phpmyadmin

Descargar php de la pagina www.php.org

Lo primero que debemos tener en cuenta es que php debe estar preconfigurado para funcionar con apache, lo cual implica que apache ya este instalado. (simplemente instalado no es necesario compilarlo), ya que php buscara el fichero de configuracion de apache sin modificaciones, y puesto que la ejecución de make para preparar la compilación de apache lo modifica, tendremos que asegurarnos de esta manera de que los ficheros no han sido modificados.

Descomprimimos php : tar –zxvf phpmyadmin_2.1.0.tar.gz
Para facilitar el acceso al directorio le cambiamos de nombre de la siguiente forma
Mv phpmyadmin_2.1.0.tar.gz myadmin

make
make install

Estas son las lineas que hay que descomentar para configurar phpmyadmin

```
$cfgServers[1]['host'] = 'localhost'; // colocar aquí la ip
$cfgServers[1]['port'] = '80'; // colocar aquí el puerto que usa la web
$cfgServers[1]['adv_auth'] = false;
$cfgServers[1]['user'] = 'root'; // colocar aquí el usuario
require("spanish.inc.php3"); // librería en castellano
```

Interfaz de configuración para apche : Comanche
Siempre es pesado tener que hacer la configuracion de cualquier programa o servicio editando un fichero de texto una y otra vez, ello nos lleva en ocasiones a usar herramientas intermedias como es el caso de comanche que no es otra cosa que un interfaz grafico bajo entorno X para facilitar la tarea de configuración del servidor web. Para su intalacion simplemente se deben proporcionar las rutas correctas para evitar que configuremos algun otro software.

Seguridad

Un ejemplo sobre los permisos que deberian tener las carpetas seria el siguiente.
/home/pablo/web/*
El directorio pablo permisos pablo:pablo 711
El directorio web permisos pablo:apache 750
El resto de los directorios pablo:apache 644

User, Group → Nombre del usuario o grupo que puede lanzar la ejecución de httpd. Como medida de seguridad no debe aparecer ningún grupo o usuario o el del propio Apache, asi solo se puede acceder via web a los documentos. Luego le damos permiso a los documentos al usuario apache y se los quitamos al resto.

- Ej User nobody
- Group nogroup

DocumentRoot → Directorio en el que se colocan los documentos web que el servidor pondra disponibles a los clientes.

- DocumentRoot /var/www/html → esta es la ruta por defecto. Directorio raiz de documentos del servidor web)

A cada directorio que Apache tiene acceso, se debe crear una estructura que lo habilite. Son lo que se conocen como directivas de contenedor.

<Directory /rutaapche/directorio>
Todo lo que este aquí se aplica solo al directorio. Es recursivo si hay directorios
#dentro del directorio
Options opciones
AllowOverride opciones
Order opciones
Allow opciones
</Directory>

<Files archivo>
#Todo lo que metamos aquí hace referencia al fichero o ficheros. Es posible usar comodines. Suele ir dentro del contenedor directory.
</Files>

<Location directorio>

Igual que el contenedor directory pero aquí se usa direccion relativa de directorio.
</Location>

Opciones para options. Va dentro de una directiva contenedora. Permite definir las caracteristicas disponibles para un directorio determinado.
None : Ninguna
All : Todas
Indexes : Permite que se visualicen indices. Permite ver el contenido del directorio si no hay pagina de inicio.
Includes : Permite incluir determinadas rutas o ficheros
FollowSymLinks : Permite el salto a través de enlaces. Permite seguir los enlaces simbolicos entre este directorio y otro donde este el enlace simbolico. (accesos directos a www).
ExecCGI : Permite la ejecución de CGIs.
La ultima opcion eliminara a la anterior. Si lo que se quiere es anadir o eliminar opciones se puede usar – o +. Ej. Options + Indexes

Multiviews: Permite la vista del directorio desde varios clientes al mismo tiempo.

Las opciones para AllowOverride son:
None : Ninguna
All : Todas
Options :
FileInfo : Información de ficheros
AutoConfig : Información por defecto
Limit : Limite de ficheros a mostrar

Opciones para Order
Allow : Permitir
Deny : Denegar

Las opciones para Allow son:
From : De donde se permiten las peticiones. Su puede usar allow from all

Las opciones pueden ser aditivas, basta separarlas con un espacio.
Ej . Options Indexes FollowSymLinks

Ej.

```
<Directory directorio_a_restringir>
#Autentificación basica. Restriccion para permitir a pocos
      order deny, allow
      deny from all
      allow form 192.168.0.   apache.org
      options indexes
      DirectoryIndex index.html
</Directory>

<Directory directorio_a_restringir2>
<Files restringido.html>
#Autentificación basica. Restriccion para permitir a muchos
       order allow, deny
       allow from all
       deny from 192.168.
       options indexes
       DirectoryIndex index.html
</Files>
```

</Directory>

ErrorLog → Contiene la ruta (absoluta o relativa al path de apache) del fiechero de log de errores. Registra lo que sucede con apache, errores y accesos de clientes. Si no se especifica en un virtual host los mensajes de error de todos los servidores virtuales se almacenarán en el mismo fichero. Si no cada servidor virtual tendra el suyo propio.
Ej. ErrorLog /var/log/error_log

LogLevel → Los que registra apache esta dividido en niveles (hasta 8), registrando mas o menos errores dependiendo de la importancia. controla el numero de mensajes que se guardarán en fichero de log de errores. Los valores posibles son:

- Debug → controla todo
- info
- notice
- warn
- error
- crit
- alert
- emerg → controla solo errores graves
- Ej LogLevel warn

CustomLog → Se usa para hacer y personalizar estadisticas de acceso al servidor.

- Ej. CustomLog logs/acces_log

LogFormat → Formato que tendran las lineas del fichero de log.
%h ip del cliente
%t fecha y hora
%b bytes enviados
%p puerto usado
%T segundos usados para efectuar la transmisión
%u nombre de usuario si hay que poner contraseña para entrar a la pagina
%U URL del cliente
%h servidor remoto que efectua la operación

Ej. Logformat “Ip del cliente: %h; hora %t” miformato
Customlog logs/access_log miformato

Aliases (Redirección interna)→ Se pueden añadir tantos alias como sean necesarios, sin limites, con el formato siguiente. Redirige de un lugar a otro dentro del propio servidor . Se suelen usar para paginas personales de los usuarios. Alias nombre_falso nombre_real

- Ej. Alias /icons/ “/var/lib/apache/icons/”
- Ej. Alias pepe /home/pepe/web → http://servidor/pepe

Un ejemplo sobre los permisos que deberian tener las carpetas seria el siguiente.
/home/pablo/web/*
El directorio pablo permisos pablo:pablo 711
El directorio web permisos pablo:apache 750
El resto de los directorios pablo:apache 644

Autentificación

SSL