Índice

1 Servidor web Apache

Un servidor web es un programa que se ejecuta de forma continua en un ordenador (también se utiliza el término para referirse al ordenador que lo ejecuta), se mantiene a la espera de peticiones por parte de un cliente (un navegador de Internet) y contesta a estas peticiones de forma adecuada, sirviendo una página web que será mostrada en el navegador o mostrando el mensaje correspondiente si se detectó algún error.

Uno de los servidores web más populares del mercado y el más utilizado actualmente es Apache, de código abierto y gratuito, disponible para Windows y GNU/Linux, entre otros.

En cuanto a su arquitectura podemos destacar lo siguiente:

- **Estructurado en módulos**. Cada módulo contiene un conjunto de funciones relativas a un aspecto concreto del servidor.
- El archivo binario **httpd** contiene un conjunto de módulos que han sido compilados.
- La **funcionalidad** de estos módulos puede ser **activada o desactivada al arrancar el servidor**.
- Los módulos de Apache se pueden clasificar en tres categorías:
    - **Módulos base**: Se encargan de las funciones básicas.
    - **Módulos multiproceso**: Encargados de la unión de los puertos de la máquina, aceptando las peticiones y atendiéndolas.
    - **Módulos adicionales**: se encargan de añadir funcionalidad al servidor.

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la **Apache Software Foundation**. La licencia de software, bajo la cual el software de la fundación Apache es distribuido, es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto.

La Licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

1.1 Instalación y configuración

Se va a realizar la instalación de un servidor Apache sobre una distribución de Linux. Para ello se utiliza el comando:

```
# sudo apt-get install apache2
```

Actividad: ¿Cuál es la dirección IP del servidor? ¿Hay más de una? ¿Hay alguna forma más de acceder al servidor?

Para verificar la correcta instalación del servidor, en un navegador web se puede acceder a la URL: http://localhost o http://127.0.0.1 también se podría acceder mediante http://IP_del_servidor

Por defecto Apache utiliza como **repositorio de páginas web** el directorio **/var/www**. En él se pueden crear carpetas para las nuevas aplicaciones web que se deseen tener en el servidor. Además, en las últimas versiones de Apache, el servidor funciona como un **host virtual** (más adelante se verá qué es) por lo que los archivos se tienen que ubicar dentro del directorio **/var/www/html**.

Actividad: Cambia el archivo por defecto del servidor por una página web en HTML 5 creada por ti.

Para configurar el servidor se utiliza el archivo **/etc/apache2/apache2.conf** en el cuál se encuentran todas las directivas de funcionamiento del servidor. Para facilitar la configuración del mismo, se han extraído partes de dicho archivo a otros y se utiliza la directiva **Include**, de esta manera es fácil localizar las diferentes secciones de la configuración. Hay que tener en cuenta que para que los cambios en la configuración sean efectivos habrá que **reiniciar el servicio apache2**.

```
# /etc/init.d/apache2 stop
# /etc/init.d/apache2 restart
# /etc/init.d/apache2 start
```

Actividad: Consultando el código del fichero /etc/apache2/conf-available/security.conf explica para qué sirven estas directivas: ServerTokens y ServerSignature.

Como se ha indicado anteriormente, en las últimas versiones de Apache, el servidor se configura mediante un servidor virtual. Mediante los servidores virtuales se pueden tener diferentes aplicaciones web (diferentes dominios) dentro del mismo servidor web. Así el host virtual principal se configura desde el archivo **/ect/apache2/sites-available/000-default.conf** y en él se pueden encontrar las siguientes directivas:

Actividad: Busca en la web oficial de Apache las directivas que se pueden usar dentro del archivo de configuración del host virtual.

Si se desea crear un nuevo host virtual lo habitual es hacer una copia del fichero **/ect/apache2/sites-available/000-default.conf**, renombrarlo con el nombre del dominio nuevo y cambiar los valores de las directivas para el nuevo host virtual. De esta forma con reiniciar el servicio ya funcionará.

Actividad: Indica en qué línea del archivo **/etc/apache2/apache2.conf** se incluyen todos los ficheros correspondientes a los hosts virtuales.

2 Funcionamiento de un servidor web

Detrás de cada página web debe existir un servidor web que ofrezca esa página, bien a los internautas, a los trabajadores de una empresa -por tratarse de una página web interna, de la empresa, no accesible a Internet, o a todo aquel que disponga de una conexión de red con la cual pueda acceder a la página.

La configuración del servidor web dependerá de las páginas web que ofrezca, así la configuración no será la misma si la página posee contenido estático o no, o si se necesita que modifique el contenido según interacción del usuario, o si se necesita de comunicación segura en la transacción de información, o si se debe tener en cuenta el control de acceso a determinados sitios de la página. Por lo tanto, según las páginas web que se ofrezcan el servidor web deberá estar configurado para tal fin: con soporte PHP, con soporte de cifrado, con soporte de control de acceso, etc.

Pero ¿un servidor web pueda alojar varias páginas web o solamente una? Es más, ¿puede alojar varios sitios (Conjunto de páginas web), dominios de Internet (Nombre por el cual se reconoce a un grupo de dispositivos o equipos conectados a la red. Éstos pueden ser nombres locales, no existentes en Internet, pero son mayoritariamente utilizados para su uso en Internet, por ejemplo: debian.org) o solamente uno, esto es, permite hosts virtuales (Dominios independientes que se pueden alojar en un mismo servidor web)?

Un servidor web puede alojar varias páginas, sitios, dominios de Internet, pero hay que tener en cuenta que la elección del servidor web será muy importante para la configuración y administración de uno o múltiples sitios,

2.1 Servicio de ficheros estáticos

¿Es necesario que todas las páginas web se modifiquen constantemente? ¿Un blog sería útil si el contenido no sufre cambios? ¿Y un manual? ¿Si actualizamos un manual la página deja de ser estática?

Todas aquellas páginas web que durante el tiempo no cambian su contenido no necesariamente son estáticas. Una página estática puede modificarse, actualizando su contenido y seguir siendo estática, ¿entonces? Entonces debemos diferenciar cuando accedemos a una página web entre código ejecutable en el lado del servidor y en el lado del cliente -equipo que solicita la página mediante el cliente web (navegador).

Ofrecer páginas estáticas es simple, puesto que solamente se necesita que el servidor web disponga de soporte html.

Para poder ofrecer páginas estáticas mediante el servidor Apache simplemente copias la página en la ruta correspondiente donde quieres que se visione la página.

Rutas de interés en la instalación de Apache (apache2) en un GNU/Linux Debian

```
/etc/apache2/
├── apache2.conf
├── conf.d
├── envvars
├── httpd.conf
├── magic
├── mods-available
├── mods-enabled
├── ports.conf
├── sites-available
└── sites-enabled
```

```
/etc/apache2/sites-available/
├── default
└── default-ssl

/var/www/
└── index.html

/etc/apache2/mods-available/mime.conf

/etc/apache2/apache2.conf
```

En la instalación de Apache se crea una página web en **/var/www/html/index.html** configurada a través del archivo **/etc/apache/sites-available/000-default.conf**, éste contiene la configuración por defecto, generada en la instalación de Apache, para esa página. Si solamente se quiere servir una página web la forma más fácil de hacerlo sería sustituyendo la página index.html por la página que quieres servir, por ejemplo empresa.html.

Actividad: En un entorno de desarrollo generalmente se tiene acceso físico al servidor, pero si la aplicación web ya está en producción ¿Cómo se pueden subir archivos al servidor?

2.2 Contenido dinámico

Imagínate que accedes a una **página web** (**recurso**) y dependiendo si posees una cuenta de usuario u otra el contenido es distinto, o que dependiendo del momento en el que entras a la página web el contenido es diferente, en ese caso te encuentras con una página dinámica.

Como bien puedes pensar, una página dinámica, necesita más recursos del servidor web que una página estática, ya que consume más tiempo de CPU y más memoria que una página estática. Además, la configuración y administración del servidor web será más compleja: cuántos más módulos tengamos que soportar, más tendremos que configurar y actualizar. Esto también tendrá una gran repercusión en la

seguridad del servidor web: cuántos más módulos más posibilidades de problemas de seguridad, así si la página web dinámica necesita, para ser ofrecida, de ejecución en el servidor debemos controlar que es lo que se ejecuta.

Algunos módulos con los que trabaja el servidor web Apache para poder soportar páginas dinámicas son: mod_actions, mod_cgi, mod_cgid, mod_ext_filter, mod_include, mod_ldap, mod_perl, mod_php5, mod_python.

2.3 Tipos MIME

Las aplicaciones web además de servir páginas web creadas con código HTML son capaces de contener otros tipos de archivos, como pueden ser imágenes, vídeos, javascript, css… El estándar MIME (Multipurpose Internet Mail Extensions), especifica **cómo un programa debe transferir archivos** de texto, imagen, audio, vídeo o cualquier archivo que no esté codificado en US-ASCII. MIME está especificado en seis RFC (Request for Comments, serie de documentos en los que se detalla prácticamente todo lo relacionado con la tecnología de la que se sirve Internet:protocolos, recomendaciones, comunicaciones...):

http://tools.ietf.org/html/rfc2045 http://tools.ietf.org/html/rfc2046 http://tools.ietf.org/html/rfc2047

http://tools.ietf.org/html/rfc4288 http://tools.ietf.org/html/rfc4289 http://tools.ietf.org/html/rfc2077

¿Cómo funciona?

Cuando el navegador solicita un recurso, en la respuesta del servidor se incluyen unas cabeceras con información sobre la respuesta, entre ellas existe una cabecera llamada **Content-Type** la cual especifica el tipo MIME del recurso que se está sirviendo. El valor que contiene esta cabecera consta de dos partes la primera indica el tipo de archivo y la segunda la codificación de dicho archivo. “text/html”

Los tipos MIME pueden indicarse en tres lugares distintos: el servidor web, la propia página web y el navegador.

- El servidor debe estar capacitado y habilitado para manejar diversos tipos MIME.
- En el código de la página web se referencia tipos MIME constantemente en etiquetas link, script, object, form, meta, por ejemplo:
  - El enlace a un archivo hoja de estilo CSS:
    <link href="./miarchivo.css" rel="stylesheet" **type="text/css"**>
  - El enlace a un archivo código javascript:
    <script language="JavaScript" **type="text/javascript"** src="scripts/mijavascript.js">
  - Con las etiquetas meta podemos hacer que la página participe en el diálogo servidor-cliente, especificando datos MIME:
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
- En el navegador cuando se realiza una petición también se informa de los tipos MIME admitidos mediante una cabecera:
  http_accept cuyo valor podría ser: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

2.3.1 Configurar el servidor para enviar tipos de MIME

En un servidor web podemos especificar el tipo MIME por defecto para aquellos archivos que el servidor no pueda identificar automáticamente como pertenecientes a un tipo concreto, esto es, para aquellos los cuales no se resuelven según su extensión.

Para el servidor web Apache se utilizan dos directivas:

- **DefaultType:** asigna la cabecera Content-Type a cualquier archivo cuya MIME no pueda determinarse desde la extensión del archivo.

- **ForceType:** hace que todos los ficheros cuyos nombres tengan una equivalencia con lo que se especifique sean servidos como contenido del tipo MIME que se establezca.

Ejemplos:

- DefaultType text/plain: Esto significa que cuando el navegador web solicita y recibe ese archivo como respuesta, desplegará el contenido como un archivo de texto.
- DefaultType text/html: Desplegará el contenido como un archivo HTML.
- ForceType image/gif: Desplegará el contenido como un archivo de imagen gif.
- ForceType video/mp4: Desplegará el contenido como un archivo de vídeo mp4.

En el siguiente enlace puedes encontrar más información sobre la directiva DefaultType.

http://httpd.apache.org/docs/2.0/mod/core.html#defaulttype

Puedes consultar más información en la documentación de Apache sobre directivas.

http://httpd.apache.org/docs/2.0/mod/directives.html

http://httpd.apache.org/docs/2.0/mod/quickreference.html

En el servidor web Apache existe el archivo **/etc/apache2/mods-available/mime.conf** donde encontrarás una referencia al archivo **/etc/mime.types**, el cual contiene la lista de tipos MIME reconocidos por el servidor.

En el siguiente enlace encontrarás la lista oficial de los tipos MIME.

http://www.iana.org/assignments/media-types/

Actividad: Busca en el fichero mime.conf ¿ En qué fichero se relaciona la aplicación con una extensión de fichero? Cita 5 extensiones que reconozcas con su correspondiente aplicación.

2.4 Hosts Virtuales

Éstos básicamente lo que hacen es permitir que un mismo servidor web pueda alojar múltiples dominios, así configurando hosts virtuales podemos alojar: empresa1.com, empresa2.com, ..., empresaN.com en el mismo servidor web. Cada empresa tendrá su **virtualhost** único e independiente de las demás.

Aunque como se ha comentado anteriormente cada virtualhost es único e independiente de los demás, todo aquello que no esté incluido en la definición de cada virtualhost se heredará de la configuración principal: apache2.conf, así, si quieres definir una directiva común en todos los virtualhost no debes modificar cada uno de los virtualhost introduciendo esa directiva sino que debes definir esa directiva en la configuración principal del servidor web Apache, de tal forma que todos los virtualhost heredarán esa directiva, por ejemplo en apache2.conf puedes encontrar la directiva **Timeout 300**, indica el número de segundos antes de que se cancele un conexión por falta de respuesta.

Existen **tres tipos** de **virtualhost:**

- ✓ Basados en nombre
- ✓ Basados en IP
- ✓ Basados en varios servidores principales.

En un servidor web en producción se configuran los servidores DNS para permitir el acceso a diferentes dominios ubicados en la misma dirección IP.

Sin embargo, en un entorno de desarrollo hay que configurar el propio servidor para que acepte esos nombres de dominios, esta acción se realiza en el archivo de configuración de **resolución de nombres** propio: **hosts**. En linux: **/etc/hosts** en windows: **c:\windows\system32\drivers\etc\hosts**.

La configuración del archivo **hosts** solo tiene efecto en el equipo en el que se realice dicha configuración.

```
192.168.200.250 empresa1.com www.empresa1.com
192.168.200.250 empresa2.com www.empresa2.com
```

2.4.1 Virtual hosts basados en nombre

La IP necesaria será la ip del servidor web.

¿Cómo se hace?

1. En la configuración de Apache2 existe un directorio **/etc/apache2/sites-available** donde se definen los virtualhosts, cada virtualhost en un fichero de texto de configuración distinto, así crea los dos ficheros siguientes en la ruta /etc/apache2/sites-available.
2. Fichero configuración virtualhost: empresa1

```
<VirtualHost IP_Servidor_Web:80>
        DocumentRoot /var/www/empresa1/
        ServerName www.empresa1.com
        ServerAlias empresa1.com empresa1.es www.empresa1.es
</VirtualHost>
```

3. Fichero configuración virtualhost: empresa2

```
<VirtualHost IP_Servidor_Web:80>
        DocumentRoot /var/www/empresa2/
        ServerName www.empresa2.com
        ServerAlias empresa2.com empresa2.es www.empresa2.es
</VirtualHost>
```

Explicación de las directivas:

<VirtualHost IP_Servidor_Web:80>: Inicio etiqueta virtualhost, define la IP del servidor web donde se aloja la página de la empresa, en este caso empresa1. El puerto TCP para el protocolo HTTP por defecto es el 80, definido en la configuración principal del servidor, mediante la directiva Listen, por lo cual no es necesario ponerlo. Se pueden usar varias directivas Listen para especificar varias direcciones y puertos de escucha. El servidor responderá a peticiones de cualquiera de esas direcciones y puertos. Por ejemplo, para hacer que el servidor acepte conexiones en los puertos 80 y 8080, usa: Listen 80 Listen 8080

DocumentRoot /var/www/empresa1/: Definición de la ruta donde está alojada la página web en el servidor, en este caso: /var/www/empresa1/ mediante la directiva DocumentRoot.

ServerName www.empresa1.com: Definición del nombre DNS que buscará la página alojada en la ruta anterior del servidor mediante la directiva ServerName. Es el nombre que escribes en el navegador para visitar la página.

ServerAlias empresa1.com: La directiva ServerAlias permite definir otros nombres DNS para la misma página.

</VirtualHost>: Fin de la etiqueta VirtualHost: fin de la definición de este virtualhost para la empresa1.

Actividad: Crea los dos hosts virtuales del ejemplo.

2.4.2 Virtual hosts basados en IP

Es posible que el servidor disponga de diferentes interfaces de red ya sean físicas o virtuales así dicho servidor dispondrá de diferentes IP mediante las cuales acceder a él. Se pueden crear hosts virtuales atendiendo a esas diferentes IP.

La creación y configuración es similar a los basados en nombre, únicamente habrá que poner la IP de cada servidor en cada directiva: **<VirtualHost IP_Servidor_Web:80>**.

Este tipo de servidores y configuración no suele ser muy habitual. Y su mantenimiento es más complicado.

2.4.3 Virtual hosts basados en varios servidores principales

Este método es el más complejo de todos, sólo tiene sentido cuando quieras tener varios archivos de configuración apache2.conf independientes organizando cada uno sus propios hosts virtuales, en otro caso, mejor emplear alguno de los dos métodos anteriores.

2.5 Módulos

La importancia de un servidor web radica en su: **estabilidad**, **disponibilidad** y **escalabilidad**. Es muy importante poder dotar al servidor web de nuevas funcionalidades de forma sencilla, así como del mismo modo quitárselas. Es por esto que la posibilidad que nos otorga el servidor web Apache mediante sus módulos sea uno de los servidores web más manejables y potentes que existen.

que necesito soporte SSL pues módulo SSL

que necesito soporte PHP pues módulo PHP

que necesito soporte LDAP pues módulo LDAP

que necesito…

En linux existen dos comandos fundamentales para el funcionamiento de los módulos en el servidor web Apache:

- **a2enmod**: Utilizado para habilitar un módulo de apache. Sin ningún parámetro preguntará que módulo se desea habilitar. Los ficheros de configuración de los módulos disponibles están en **/etc/apache2/mods-available/** y al habilitarlos se crea un enlace simbólico desde **/etc/apache2/mods-enabled/**.
- **a2dismod**: Utilizado para deshabilitar un módulo de Apache. Sin ningún parámetro preguntará que módulo se desea deshabilitar. Los ficheros de configuración de los módulos disponibles están en **/etc/apache2/mods-available/** y al deshabilitarlos se elimina el enlace simbólico desde **/etc/apache2/mods-enabled/**.

Si no se dispone de esos comandos para poder habilitar y deshabilitar módulos Apache simplemente se pueden crear a mano los enlaces simbólicos correspondientes desde **/etc/apache2/mods-enabled/** hasta **/etc/apache2/mods-available/**.

Mediante el comando **a2ensite** se pueden habilitar aplicaciones web/hosts virtuales. Los ficheros de configuración de los ''sitios web'' disponibles (normalmente son configuraciones de hosts virtuales) están en **/etc/apache2/sites-available/** y al habilitarlos se crea un enlace simbólico desde **/etc/apache2/sites-enabled/**

2.5.1 operaciones sobre módulos

Los módulos de Apache puedes instalarlos, desinstalarlos, habilitarlos o deshabilitarlos, así, puedes tener un módulo instalado, pero no habilitado. Esto quiere decir que, aunque instales módulos hasta que los habilites no funcionarán.

En la tabla siguiente encontrarás un resumen de operaciones, ejemplos y comandos necesarios que se le pueden realizar a los módulos:

Operaciones sobre los módulos Apache en Linux	
Instalar un módulo	apt-get install nombre-modulo Ej: apt-get install libapache2-mod-gnutls
Desinstalar un módulo	apt-get remove nombre-modulo Ej: apt-get remove libapache2-mod-gnutls
Habilitar un módulo	a2enmod nombre-modulo-apache Ej: a2enmod ssl
Deshabilitar un módulo	a2dismod nombre-modulo-apache Ej: a2dismod ssl

Instalar Módulos de Apache

1. Listar los módulos de Apache disponibles: **apt**-cache search libapache2*
2. **Instalar** cualquier módulo deseado: **apt-get install** [**nombre-módulo**]
3. Todos los módulos se encuentran en el directorio the /etc/**apache2**/mods-available. Edita el archivo .conf de cualquier módulo instalado si fuera necesario y luego habilita el **módulo**

2.5.2 Configuración en módulos

Parte de la configuración por defecto de Apache en las últimas versiones se ha convertido en módulos. Como ejemplo se puede ver el **módulo de directorio**, mediante este módulo se puede indicar qué recurso se tiene que devolver cuando en la petición al servidor web no se especifica ningún recurso concreto como cuando se realiza la petición indicando únicamente el nombre de dominio: **www.dominio.com** o cuando se indica un directorio: www.dominio.com/directorio.

En el caso anteriormente indicado, la directiva que se utiliza se ha extraído a un archivo llamado **dir.conf** que se encuentra dentro del directorio **mods-available**. En la directiva se indican varios archivos a buscar en caso de no pedirse ningún recurso y en orden de preferencia, por defecto el recurso que se sirve cuando no se pide ningún recurso en concreto es **index.html**.