Administración de Servidores Web

Apache

RAFAEL CABRERA PARGA

Terminología

- Procesos cliente/servidor.
- Servidor Web.
- Dirección IP.
- Servidor DNS.
- Puerto. Cada servicio tiene asignado un numero de puerto predeterminado (Web, puerto 80). En etc/services podemos ver el puerto asignado a cada servicio.
- Socket. Permite la comunicación entre dos maquinas.
- Protocolo. HTTP (HyperText Transfer Protocol).
- Servidor Apache.

Servidor Apache

- Servidor Web más utilizado. Por defecto en cualquier distribución de Linux.
- Funcionamiento básico: proceso padre que hace copias de si mismo para atender a todas las peticiones de los clientes.
- Demonio escuchando el puerto 80(httpd).
- Para ponerlo en servicio:
  - comando: service httpd Start/stop/restart
  - script: /etc/init.d/httpd Start/stop/restart
  - comando setup: opción servicios, marcamos httpd

Instalación

- Descargar: www.apache.org.
- Descomprimirlo: tar zvxf-apache_x.x.x.tar.gz Se generará el directorio apachex.x.x
- Configurarlo :
  - ./configure –preefix =ruta de instalacion. Indicamos el directorio de instalación y modificamos algunos aspectos de la configuración.
  - Modificando directamente el script de configuración configuration.tmpl(se recomienda hacer una copia)
- Make. Ayuda a la compilación a partir del fuente, las bibliotecas y los ficheros makefile.
- Make install. Precompila el código fuente con las opciones del script.

Instalación

- Existe otra forma de hacerlo, disponible para algunas distribuciones (RedHat o Mandrake) en la que en vez de descargarse el archivo binario .tar y descomprimirlo, descargamos el archivo .rpm y la descompresión e instalación se harán en un solo comando.
- En este otro caso los pasos seguir serán:
  - rpm –q apache
  - rpm ivh- apache_x.x.x.rpm
- Una vez hecho esto ya podemos ejecutar el proceso servidor. Esto lo hacemos mediante el comando:
  - /etc/init.d/httpd Start
  - service httpd Start

Para verificar el correcto funcionamiento, desde un cliente web, accedemos a la dirección IP de nuestro servidor Apache y este deberá responder con el test Apache.

Configuración

- La configuración del servidor se hace a través de directivas.
- El fichero de configuración del servidor Apache es: /etc/httpd/conf/httpd.conf.
- httpd.conf divide en 3 secciones.
  - Sección 1. Entorno global. Descripción del funcionamiento general del servidor y rutas de acceso a otros ficheros de configuración.
  - Sección 2. Entorno del servidor principal. Comportamiento predeterminado del servidor.
  - Sección 3. Servidores virtuales que se pueden definir en este fichero para emular diferentes servidores.

Directivas de entorno global

- ServerType *opción*. Tipo de respuesta del servidor.
  - *Inetd*. Cuando el servidor recibe una petición, el demonio inetd inicia el proceso httpd y luego lo mata.
  - *Standalone*. Un proceso httpd específico siempre esta en ejecución.
- ServerRoot. */ruta* Directorio raíz para el servidor. Por defecto es /etc/httpd
- Timeout *segs*. Indica el intervalo de espera entre peticiones web antes de desconectar. Para evitar bloqueos de conexiones.
- Maxclients *num*. Límite total de procesos del servidor que se pueden ejecutar a la vez. Evita que el servidor bloquee al sistema operativo. Por defecto es 150 y no se recomienda que sea mayor de 256.
- KeepAlive on/off. Determina si el servidor permite varias peticiones para un mismo cliente.

Directivas de entorno global

- MinSpareServers/MaxSpareServers *num*. Apache se adapta dinámicamente a la carga percibida y mantiene un número de servidores libres basado en el tráfico.
- StartServers *num*. Cuántos procesos se crean por defecto al arrancar el servidor.
- Listen *puerto/dir IP*. Identifica los puertos por los que el servidor aceptará peticiones entrantes. Por defecto son 80/443 para conexiones no seguras/seguras. Si el servidor está configurado para aceptar peticiones por puertos menores de 1024, necesitamos al servidor para arrancarlo.
- Pidfile. Indica el archivo en el que el servidor guarda su ID de proceso. Por defecto este archivo es: var/run/httpd.pid
- LoadModule nom_mod/ruta_mod. Carga el módulo que le indiquemos en esa ruta.

Directivas de entorno global

Directivas de contenedor

Se crean para definir y agrupar las directivas que afectan a los directorios a los que tiene acceso nuestro servidor.

- <Directory */ruta*>
  Options *opciones*
  AllowOverride *opciones*
  Order *opciones*
  Allow *opciones*
  Deny opciones
  </Directory>

  Crea un contenedor. Este grupo de directivas, se aplican al directorio indicado. Por defecto existe un contendor para DocumentRoot.
- <DirectoryMatch */ruta*> </DirectoryMatch> Permite expresiones regulares en la ruta.
- <File *fichero*> </File> Solo aplica las directivas al fichero indicado

Directivas de entorno global

Directivas de contenedor

- Opciones para options:
  - None
  - All (exceto Multiviews).
  - Indexes. Permite ver el contenido del directorio aunque no haya página de inicio.
  - Includes. Permite incluir determinadas rutas o ficheros.
  - FollowSymLinks. Sigue los enlaces simbólicos entre este directorio y otro.
  - SymLinksIfOwnerMatch. Solo si coincide el propietario del enlace y del destino.
  - ExcecCGI. Permite la ejecución de scripts CGI.
  - Multiviews. Permite la vista del directorio desde varios clientes simultaneamente.

Directivas de entorno global

Directivas de contenedor

- Fichero htacces. Cada usuario puede configurar su sitio web mediante un htacces. Éste tiene el mismo el mismo formato que httpd.conf y debe estar en el directorio en el que se quieren realizar.
- Opciones de allowOverride: le indica al servidor que las opciones del contenedor las tome del fichero htacces. Éstas últimas anulan las opciones de options.

  Algunos posibles valores son:
  - None. No permite directivas y no busca el fichero htacces
  - All. Permite todas las directivas y busca el fichero.
  - Options. Permite el uso de options.
  - Limit. Número de ficheros a mostrar

Directivas de entorno global

Directivas de contenedor

- Order.
  - Allow,deny. Permite a todos los que no estén denegados.
  - Deny,allow. Deniega todo lo que no está permitido.
- Allow y Deny definen quién puede acceder al directorio. Los posibles valores son:
  - All. Permite o deniega a todos.
  - Dirección IP.
  - Nombre de Dominio

Directivas del servidor principal

- Port *num*. Puerto por el que escucha el servidor principal. Solo puede haber uno.
- User/Group *nombre.* Establece el nombre de usuario para el proceso del servidor. Debe ser *Apache*
- ServerAdmin *e-mail*. Esta dirección de correo aparecerá en los mensajes de error generados por el servidor.
- ServerName *DNS:puerto.* Define el nombre de servidor y puerto que se enviará a los clientes. No debe coincidir con el nombre real de la maquina, pero debe ser un nombre de dominio que pueda ser resuelto por un servidor DNS.

Directivas del servidor principal

- ServerSignature *on/off/email.* En un mensaje de error se muestra el nombre de la máquina y la versión de apache usada.
- DocumentRoot */ruta.* Directorio que contiene los archivos HTML para enviar a los clientes en respuesta a sus peticiones. Por defecto es /var/www/html.

  Ejemplo:

  El cliente solicita: http://ejemplo.com/foo.html

  El servidor buscará el archivo: /var/www/html/foo.html

Directivas del servidor principal

- DirectoryIndex *doc*. Cuando se produce una petición que termina en /, es decir, un directorio,el servidor mostrará la página que indiquemos aquí. Por defecto es index.html.

  Ejemplo:

  DirectoryIndex index.html

  petición: http://example.com/directorio/

  se sirve: http://example.com/directorio/index.html

  Si el servidor no encuentra este documento, comprobará si está la opción indexes, y si es así, el servidor genera una lista en formato HTML con los subdirectorios que contiene ese directorio.

Directivas del servidor principal

- <IfModule *módulo*> </IfModule> Crea un contenedor que se activa si se carga el módulo especificado.
- UserDir *enable/disable/ruta.* Indica cual es el directorio del usuario que realiza la petición. Cuando se produce una petición del tipo ~user, el servidor buscará la pagina pedida en el directorio del usuario especificado en esta directiva.

  Ejemplo:

  UserdDir /public_html

  petición: http://exaple.com/~username/foo.html

  servidor accede a: home/username/public_html/foo.html
- AddModule *modulo*. Permite el uso de módulos compilados que no estén activos.
- AddType *tipoMIME extensión.* Crea una asociación entre el tipo MIME (Multimedia Internet Mail Extensión) y una extensión.

  Ejemplo:

  AddType text/html .shtml

Directivas del servidor principal

- DefaultType *tipo*. Cuando se le pase un documento cuyo tipo MIME desconozca, le asignará el que indiquemos aquí. Por defecto es plain/text.
- ErrorLog */ruta*. Determina el archivo donde se guarda los errores que se producen en el servidor.
- TransferLog */ruta*. Guarda información acerca de los datos que salen y entran al servidor. Esta opción carga mucho la directiva.
- CustombLog */ruta frmt*. Crea un fichero de registro con el formato especificado en *frmt*. A *frmt* le damos formato en la directiva LogFormat.

Directivas del servidor principal

- LogFormat */ruta frmt* . Crea un formato determinado. Las diferentes opciones son:
  - %A Dirección Ip local
  - %a Dirección Ip remota
  - %f Ruta del documento solicitado
  - %p Puerto TCP por el que se recibió la petición
  - %t Fecha y hora
  - %b Bytes enviados
  - %T segundos que se tardó en procesar la petición
  - %u Usuario remoto en conexiones autentificadas
  - %v Nombre del servidor

  Ejemplo:

  LogFormat “Dir. Local = %A” formato

Directivas del servidor principal

- LogLevel *nivel*. Existen 8 niveles de error, dependiendo del que especifiquemos aquí, se guardará mas o menos información en el fichero de ErrorLog.

  Los distintos niveles son:
  - Debug. Guarda todo.
  - Info. Solo guarda los mensajes de información.
  - Notice.
  - Warn. Advertencias.
  - Error.
  - Crit.
  - Alert.
  - Emerg. Solo guarda los mensajes mas graves.

Directivas del servidor principal

- Alias *alias ruta*. Permite que haya directorios fuera del DocumentRoot a los que puede acceder el servidor. Cualquier URL que acabe en el alias, será traducida a la ruta.

  Ejemplo:

  Alias pepe /home/pepe/web

  URL: http://server/pepe

  acceso: http://server/home/pepe/web
- ScriptAlias *alias ruta*. Define un directorio fuera del DocumentRoot que contiene scripts y procesos del servidor.
- Redirect */ruta nuevaURL*. Cualquier petición del documento solicitado en la ruta, será automáticamente redirigido a la nueva ubicación.

  Ejemplo:

  Redirect /pepe.html http://www.pepeweb.com/pepe.html

Directivas de servidor virtual

- Servidor Virtual. Permite ejecutar varios servidores dentro de la misma máquina y así alojar varios sitios en un mismo servidor.
- Ventajas
  - Configuración. Con una sola configuración de entorno global ya tendré configurados todos.
  - Altamente personalizable. Al definir un Host Virtual puedo añadir cualquier configuración aplicable a un servidor principal.
  - Actualizaciones. Solo será necesario hacerlas una vez.
  - Precio.
- Desventajas
  - Fragilidad. Un fallo en el sistema provocará que se caigan todos los servidores.
  - Actualizaciones. Si hay que reiniciar Apache, se pararán todos los servidores
  - Seguridad. Cualquier problema de seguridad afecta a todos los servidores.

Directivas de servidor virtual

- Servidor Virtual basado en nombre. A una misma dirección IP se le asignan varios dominios. A cada uno de estos dominios se les asignará un servidor virtual. (solo compatible con http v1.1).
- Para definirlo usaremos varias directivas:
  - NameVirtualHost *dir.IP:puerto*. La dirección IP será la de la máquina. Aquí se hace referencia a la maquina virtual predeterminada.
  - <VirtualHost *dir.IP*></VirtualHost> Se define un contendor para las directivas de nuestro servidor virtual. La IP será también del servidor.
  - <VirtualHost_default_></VirtualHost> Si una petición no coincide con ninguna configuración de ninguna máquina virtual, éste servidor será el que la atienda.
- ServerAlias. A cada servidor virtual le daremos un alias que lo identifique

Directivas de servidor virtual

- Servidor virtual basado en dirección IP. Se asocia cada servidor virtual a una dir.IP diferente. Nuestra máquina deberá tener varias direcciones IP asignadas.
- Para definirlos crearemos un contendor VirtualHost.
  - <VirtualHost *dir.IP:puerto*></VirtualHost>
  - ServerName. Dentro del contenedor definido en VirtualHost tendremos que definir un nombre para el servidor.

httpd.conf

```
## httpd.conf - configuration for the Apache web server
#
# Generated automatically... if you edit manually, the changes will be lost
# the next time you run "apacheconfig".
#
# What we listen to
#

ServerType StandAlone

ServerRoot /etc/httpd/



# We don't handle this yet...
```

httpd.conf

```
...
# Dynamic Shared Object (DSO) Support
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding `LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Please read the file README.DSO in the Apache 1.3 distribution for more
# details about the DSO mechanism and run `httpd -l' for the list of already
# built-in (statically linked and thus always available) modules in your httpd
# binary.
#
# Note: The order is which modules are loaded is important.  Don't change
# the order below without expert advice.
#
# Example:
# LoadModule foo_module modules/mod_foo.so

#LoadModule mmap_static_module modules/mod_mmap_static.so
LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule env_module        modules/mod_env.so
LoadModule config_log_module  modules/mod_log_config.so
LoadModule agent_log_module   modules/mod_log_agent.so
...
```

httpd.conf

```
...
<IfDefine HAVE_PERL>
LoadModule perl_module       modules/libperl.so
</IfDefine>
<IfDefine HAVE_PHP>
LoadModule php_module        modules/mod_php.so
</IfDefine>
<IfDefine HAVE_PHP3>
LoadModule php3_module       modules/libphp3.so
</IfDefine>
<IfDefine HAVE  PHP4>
LoadModule php4_module       modules/libphp4.so
</IfDefine>
...
#  Reconstruction of the complete module list from all available modules
#  (static and shared ones) to achieve correct module execution order.
#  [WHENEVER YOU CHANGE THE LOADMODULE SECTION ABOVE UPDATE THIS,
TOO]
ClearModuleList
#AddModule mod_mmap_static.c
AddModule mod_vhost_alias.c
AddModule mod_env.c
AddModule mod_log_config.c
...
```

httpd.conf

```
...
# Documents
DocumentRoot /var/www/html
UserDir public_html
IndexOptions FancyIndexing
# Who runs the server?
User apache
Group apache
# Performance parameters
MaxClients 150
TimeOut 300
ErrorLog /var/log/httpd/error_log
LogLevel warn
# Need to fix  this
LogFormat "%h %l %u %t %b" common
CustomLog /var/log/httpd/access_log common
...
# The following is for PHP3:
<IfModule mod_php3.c>
  AddType application/x-httpd-php3 .php3
  AddType application/x-httpd-php3-source .phps
</IfModule>
...
```

httpd.conf

```
...
# Virtual host www.pepe-informatica.com
<VirtualHost 192.168.0.2>
        UserDir /home/*/www
        DocumentRoot /home/httpd/pepe-informatica.com/html
        ServerAdmin webmaster@pepe-informatica.com
        ServerName www.pepe-informatica.com
        DirectoryIndex index.php default.html default.php index.html
        <Directory "/home/httpd/pepe-informatica.com/html/">
                AllowOverride none
        </Directory>
        scriptalias /cgi-bin/ /home/httpd/pepe-informatica.com/cgi-bin/
        <Directory "/home/httpd/pepe-informatica/cgi-bin/">
                Options ExecCGI
                AllowOverride none
                order allow,deny
                allow from all
        </Directory>
        LogLevel debug

</VirtualHost>
...
```