```
PerlSetVar ProtectedSecure 1
PerlSetVar  ProtectedLoginScript  /protectedloginform
```

La siguiente lista le dice lo que está ocurriendo en la configuración anterior:

- La clave `ProtectedTicketExpires` determina el tiempo de expiración de la sesión (ticket) en minutos.
- La llave `ProtectedTicketLogoutURI` determina la URL que se muestra tras utilizar un registro de usuario.
- `ProtectedTicketLoginHandler` determina la ruta del manejador de registros, que debe corresponder a un contenedor `<Location>`, tal y como se discute más tarde.
- `ProtectedTicketIdleTimeout` determina el número de minutos en los que una sesión puede estar parada.
- `ProtectedPath` determina la ruta de la cookie. El valor por defecto `/` asegura que se devuelve la cookie con todas las solicitudes. Puede restringir la cookie al área protegida simplemente cambiando `/` a `/protected` (o a cualquier localización que esté protegiendo).
- `ProtectedDomain` determina el nombre de dominio de la cookie. El punto inicial asegura que la cookie es enviada a todos los host Web en el mismo dominio. Por ejemplo, si lo fijamos en `.mobidac.com` permitiría ver la cookie en `web1.Mobidac.com` o en `web2.Mobidac.com`. También puede restringir la cookie a un sólo host especificando aquí el *fully qualified host name*.
- Fijando `ProtectedSecure` en 1 aseguramos que la cookie es segura.
- `ProtectedLoginScript` determina la localización para el formulario de registro, que es generado por el módulo.

7. Ahora necesita crear un contenedor `<Location>` para el directorio `/protected` del siguiente modo:

```
<Location /protected>
     AuthType Apache::AuthTicket
     AuthName Protected
     PerlAuthenHandler Apache::AuthTicket->authenticate
     PerlAuthzHandler Apache::AuthTicket->authorize
     require valid-user
</Location>
```

Aquí se le dice a Apache que pida las credenciales válidas del usuario, que ha de ser autentificado por el módulo `Apache::AuthTicket`.

8. Ahora necesita establecer los manejadores para la pantalla de registro, script de registro y las funciones logout del módulo del siguiente modo:

```
<Location /protectedloginform>
   AuthType Apache::AuthTicket
   AuthName Protected
   SetHandler perl-script
   Perlhandler Apache::AuthTicket->login_screen
</Location>

<Location /protectedlogin>
   AuthType Apache::AuthTicket
   AuthName Protected
   SetHandler perl-script
   PerlHandler Apache::AuthTicket->login
</Location>

<Location /protected/logout>
   AuthType Apache::AuthTicket
   AuthName Protected
   SetHandler perl-script
   PerlHandler Apache::AuthTicket->logout
</Location> </Location>
```

9. Una vez que ha creado la configuración anterior, asegúrese de que ha añadido al menos un usuario a la tabla `wwwusers`. Ver la sección "Gestionar usuarios y grupos en una RDBM" de este capítulo para obtener los detalles de cómo gestionar usuarios en una base de datos.

10. Reinicie el servidor Web de Apache utilizando el comando `/usr/local/apache/bin/apachectl restart`.

11. Asegúrese de que ve la cookie, determine en su navegador Web que pida las cookies. Para Netscape Navigator, puede comprobar el Warn me antes de almacenar una opción cookie utilizando la opción Edit>Preference>Advanced>Cookies. Para Microsoft IE, debe utilizar las opciones Tools>Internet Options>Security>Custom Levels>Cookies>Prompt.

12. Ahora acceda al directorio `http://your_server_name/protected/` y verá un formulario Web solicitando su nombre de usuario y su contraseña. Introduzca un nombre de usuario válido y una contraseña no válida y el formulario Web debería simplemente volver a mostrarse. Ahora introduzca un par nombre de usuario/contraseña válido y su navegador Web le pedirá permiso para almacenar la cookie. A continuación tenemos una muestra de sesión (ticket) inválida.

```
Cookie Name: Apache::AuthTicket_Protected
Cookie Domain: nitec.com
Path: /
Expires: End of session
```

```
Secure: Yes
Data:
expires:988390493:version::user:kabir2:hash:
bf5ac94173071cde94489ef79f24b158:time:988389593
```

13. Capacite al navegador web para que almacene la cookie y tendrá acceso a la sección Web restringida.

14. A continuación, debería verificar que no hay un nuevo ticket en la tabla de tickets. Puede registrarse en su base de datos del servidor y ver el contenido de la tabla de tickets. Por ejemplo, en un sistema Linux ejecutando un servidor MySQL, puede ejecutar el comando `select * from tickets`, una vez que está registrado en MySQL mediante el comando `mysql -u httpd -p auth`. A continuación se muestra una salida:

```
mysql> select * from tickets;
+----------------------------------+-----------+
| ticket_hash                      | ts        |
+----------------------------------+-----------+
| 145e12ad47da87791ace99036e35357d | 988393278 |
| 6e115d1679b8a78f9b0a6f92898e1cd6 | 988393401 |
+----------------------------------+-----------+
2 rows in set (0.00 sec)
```

Aquí MySQL informa que hay dos sesiones conectadas actualmente a un servidor Web.

15. Puede forzar a los navegadores Web para que se registren de nuevo, eliminando los tickets almacenados en esta tabla. Por ejemplo, editando el comando `delete from tickets` en su servidor de bases de datos, elimina todos los registros en la tabla de tickets y fuerza a todo el mundo a que se registre de nuevo.

8 Monitorización del acceso a Apache

En este capítulo

1. Monitorizamos el estado de Apache.
2. Permitimos el registro.
3. Personalizamos el registro.
4. Archivamos sus registros.
5. Realizamos un seguimiento de usuarios.
6. Analizamos sus archivos de registro.
7. Mantenemos sus archivos de registro.

Seguramente se ha preguntado alguna vez quién accede a su sitio Web o cómo funciona el servidor Apache en su sistema. La monitorización, el registro y el análisis del servidor Apache, pueden proporcionar una gran cantidad de información vital para el administrador de sistemas del servidor Web, y además puede ser de gran ayuda en los aspectos de marketing de su sitio Web. En este capítulo, le mostraré la forma de controlar y registrar información en un servidor Apache, para satisfacer sus necesidades de conocimiento.

Entre otras cosas, en este capítulo le voy a mostrar cómo:

- Acceder rápidamente a las configuraciones de los servidores Apache
- Monitorizar el estado de un servidor de Apache que se está ejecutando
- Crear archivos de registro tanto en formato CLF como en formatos personalizados
- Analizar archivos de registro utilizando aplicaciones de terceras partes

Monitorizar Apache

Apache le permite monitorizar estos dos tipos de información valiosa vía Web:

- **Información de la configuración del servidor:** esta información es estática, pero acceder a ella rápidamente puede resultar muy útil cuando quiere determinar qué módulos están instalados en el servidor.
- **Información del estado del servidor:** esta información cambia constantemente. Puede monitorizar información del tipo tiempo de operación del servidor, número total de solicitudes servidas, transferencia total de datos, estado de los procesos hijo y manejo de los recursos del sistema.

Voy a analizar ambos tipos de información en las siguientes secciones.

Acceder a la información de configuración con mod_info

Se puede acceder a la información de la configuración del sistema mediante el módulo `mod_info`. Este módulo proporciona un resumen de la configuración del servidor, incluyendo todos los módulos instalados y todas las directivas de los archivos de configuración. Este módulo se encuentra en el archivo `mod_info.c`. No está compilado en el servidor por defecto. Tiene que compilarlo utilizando la opción `--enable-info` en el script de configuración. Por ejemplo:

```
./configure   --prefix=/usr/local/apache \
              --with-mpm=prefork \
              --enable-info
```

Este comando, configura Apache de modo que se queda instalado en el directorio `/usr/local/apache`, configura la fuente para que se ejecute como un servidor "preforking", y habilita el módulo `mod_info`. Ejecute `make` y `make install` para compilar e instalar el nuevo servidor construido de Apache.

Una vez que tiene instalado este módulo en el servidor, puede ver la información de la configuración de la Web, añadiendo la siguiente configuración al archivo `httpd.conf`:

```
<Location /server-info>
    SetHandler server-info
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1 .domain.com
</Location>
```

Esto le permite al host local (127.0.0.1) y a cada host de su dominio, acceder a la información del servidor. No olvide reemplazar `.domain.com` con el nombre de dominio de su máximo nivel. Por ejemplo, si su sitio es `www.nitec.com`, necesita añadir:

```
Allow from 127.0.0.1 .nitec.com
```

El punto delante del nombre de dominio permite que cualquier host del dominio acceda a la información del servidor. Sin embargo, si desea limitar todo esto a un solo dominio llamado `sysadmin.domain.com`, cambie la línea `Allow from` por:

```
Allow from 127.0.0.1 sysadmin.domain.com
```

Una vez que el servidor está configurado y reiniciado, la información del servidor se obtiene del host local (es decir, ejecutando el navegador Web con Lynx en el propio servidor) accediendo `http://localhost/server-info`.

Esto devuelve una página completa de configuración del servidor y de todos los módulos. Si quiere acceder desde localizaciones distintas, utilice el nombre completo del servidor, incluido el dominio, el *fully qualified name* en lugar del nombre del host local. Por ejemplo, si su servidor Web se llama `www.nitec.com`, accederá a la información del servidor utilizando: `http://www.nitec.com/server-info`.

El módulo `mod_info` también proporciona una directiva llamada `AddModuleInfo`, que le permite añadir texto descriptivo en la lista de módulos proporcionada por el módulo `mod_info`. El texto descriptivo podría ser cualquier cosa incluido texto HTML. `AddModuleInfo` tiene la siguiente sintaxis:

```
AddModuleInfo module_name descriptive_text
```

Por ejemplo:

```
AddModuleInfo mod_info.c 'See <a href="http://localhost/manual/
mod/mod_info.html">man mod_info</a>'
```

Esto muestra un enlace HTML junto con la lista de `mod_info.c`, proporcionando un modo sencillo de obtener información en el módulo desde el manual online de Apache, tal y como se muestra a continuación.

```
Module Name: mod_info.c
Content handlers: (code broken)
Configuration Phase Participation: Create Server Config, Merge
Server Configs
```

```
Module Directives:
AddModuleInfo - a module name and additional information on
that module
Current Configuration:
AddModuleInfo mod_info.c 'man mod_info'

Additional Information:
man mod_info
```

También puede limitar la información desplegada en la pantalla del siguiente modo:

- **Sólo configuración del servidor.** Utilice `http://server/server-info?server`, que muestra la siguiente información:

```
Server Version: Apache/2.0.14 (Unix)
Server Built: Mar 14 2001 12:12:28
API Version: 20010224:1
Hostname/port: rhat.nitec.com:80
Timeouts: connection: 300    keep-alive: 15
MPM Information: Max Daemons: 20 Threaded: no Forked: yes
Server Root: /usr/local/apache
Config File: conf/httpd.conf
```

- **Configuración de un sólo módulo.** Utilice `http://server/server-info?module_name.c`. Por ejemplo, para ver información sobre el módulo `mod_cgi`, ejecute `http://server/server-info?mod_cgi.c`, que mostrará la siguiente información:

```
Module Name: mod_cgi.c
Content handlers: (code broken)
Configuration Phase Participation: Create Server Config,
Merge Server Configs
Module Directives:
ScriptLog - the name of a log for script debugging info
ScriptLogLength - the maximum length (in bytes) of the
script debug log
ScriptLogBuffer - the maximum size (in bytes) to record of a
POST request
Current Configuration:
```

- **Una lista con los módulos compilados.** Utilice `http://server/server-info?list`, que muestra la siguiente información:

```
mod_cgi.c
mod_info.c
mod_asis.c
mod_autoindex.c
mod_status.c
prefork.c
mod_setenvif.c
mod_env.c
```

```
mod_alias.c
mod_userdir.c
mod_actions.c
mod_imap.c
mod_dir.c
mod_negotiation.c
mod_log_config.c
mod_mime.c
http_core.c
mod_include.c
mod_auth.c
mod_access.c
core.c
```

Por supuesto, su lista variará basándose en qué módulos tiene activados durante la configuración de la fuente. Ahora, vamos a ver cómo puede monitorizar el estado de un servidor Apache en ejecución.

Permitir páginas de estado con with mod_status

El módulo `mod_status` permite a los administradores de Apache monitorizar el servidor mediante la Web. Se crea una página HTML con las estadísticas del servidor. Se genera, además, otra página de programación muy intuitiva. La información que se muestra en estas páginas incluye:

- El momento actual del sistema servidor
- El momento en el que se reinició el servidor por última vez
- El tiempo transcurrido desde que empezó a funcionar
- El número total de accesos al servidor hasta ese momento
- El número total de bytes transferidos hasta ese momento
- El número de solicitudes hijo servidas
- El número total de hijos desocupados
- El estado de cada hijo, el número de solicitudes que el hijo procesa y el número de bytes servidos por hijo
- Medias del número de solicitudes por segundo, el número de bytes servidos por segundo y el número de bytes por solicitud
- El porcentaje actual de CPU utilizada por cada hijo y el total utilizado por Apache
- El host y las solicitudes que se están procesando

Al igual que el módulo `mod_info`, este módulo tampoco está compilado por defecto en la distribución estándar de Apache, por lo que tiene que utilizar la

opción `--enable-status` en el script de configuración y compilar e instalar Apache.

> **NOTA: Parte de la información que acabamos de nombrar sólo está disponible cuando es capaz de desplegar ese tipo de información utilizando la directiva `ExtendedStatus`, que se discutirá más tarde en esta sección.**

Ver páginas de estado

Una vez que tiene el módulo `mod_status` compilado y construido en su servidor Apache, necesita definir la localización URL que Apache debería utilizar para mostrar la información. En otras palabras, necesita decirle a Apache qué URL mostrará las estadísticas del servidor en su navegador Web.

Vamos a suponer que su nombre de dominio es `domain.com`, y que quiere utilizar la siguiente URL:

```
http://www.domain.com/server-status
```

Utilizando el contenedor `<Location . . .>`, puede decirle al servidor que quiere que maneje esta URL utilizando el manejador de estado del servidor que se encuentra en el módulo `mod_status`. El siguiente contenedor es el encargado de llevar a cabo el trabajo:

```
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1 .domain.com
</Location>
```

A continuación, una vez que ha añadido la configuración en `httpd.conf`, reinicie el servidor y acceda a la URL desde un navegador. El contenedor `<Location . . .>` le permite acceder a la información de estado desde cualquier host de su dominio, o desde el servidor en sí. No olvide cambiar `.domain.com` a su verdadero nombre de dominio, ni olvide incluir el punto inicial.

> **TRUCO: Puede hacer que la página de estado se actualice a sí misma automáticamente utilizando la URL `http://server/server-status?refresh=N` para actualizar la página cada N segundos.**

Para ver la información de estado extendida, añada la directiva `ExtendedStatus On` en el contexto de configuración del servidor. Por ejemplo, la

configuración completa relacionada con el estado del servidor en `httpd.conf` podría ser la siguiente:

```
ExtendedStatus On
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1 .domain.com
</Location>
```

A continuación se muestra un ejemplo de la información de estado ampliada:

```
Apache Server Status for rhat.nitec.com
Server Version: Apache/2.0.14 (Unix)
Server Built: Mar 14 2001 12:12:28

------------------------------------------------------------
Current Time: Thursday, 15-Mar-2001 11:05:08 PST
Restart Time: Thursday, 15-Mar-2001 11:02:40 PST
Parent Server Generation: 0
Server uptime: 2 minutes 28 seconds
Total accesses: 17807 - Total Traffic: 529 kB
CPU Usage: u173.4 s.03 cu0 cs0 - 117% CPU load
120 requests/sec - 3660 B/second - 30 B/request
4 requests currently being processed, 8 idle servers
_WKKK.....................................................
_.....................................................
_.....................................................
_.....................................................
_.....................................................
_.....................................................
_.....................................................
_.....................................................

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"L" Logging, "G" Gracefully finishing, "." Open slot with no
current process

Srv PID Acc M CPU  SS Req Conn Child Slot Client VHost Request
0-0 0 0/87/87 _  0.07 1726072572 0 0.0 0.10 0.10
(unavailable)
0-0 0 105/105/105 W  0.00 1726072572 0 50.5 0.05 0.05
(unavailable)
0-0 0 166/166/166 K  0.02 1726072572 0 233.5 0.23 0.23
(unavailable)
0-0 0 49/49/49 K  0.01 1726072572 0 25.2 0.02 0.02
(unavailable)
0-0 0 77/77/77 K  0.08 1726072572 0 116.6 0.11 0.11
(unavailable)
```

```
4-0 0 0/0/17323 _   173.25 1726072572 0 0.0 0.00 0.00
(unavailable)

---------------------------------------------------------------
Srv Child Server number - generation
PID OS process ID
Acc Number of accesses this connection / this child / this slot
M Mode of operation
CPU CPU usage, number of seconds
SS Seconds since beginning of most recent request
Req Milliseconds required to process most recent request
Conn Kilobytes transferred this connection
Child Megabytes transferred this child
Slot Total megabytes transferred this slot

---------------------------------------------------------------

Apache/2.0.14 Server at rhat.nitec.com Port 80
```

Simplificar el despliegue de estado

La página de estado desplegada por el módulo `mod_status` proporciona información extra que la hace inapropiada para utilizarla como archivo de datos para cualquier programa analizador de datos.

Así, por ejemplo, si quiere crear un gráfico para el estado de los datos de su servidor utilizando un programa de hojas de cálculo, tendría que borrar los datos manualmente. Sin embargo, el módulo proporciona un modo de crear salidas legibles para la máquina, para esa misma URL, modificándolas utilizando el `?auto` en `http://server/server-status?auto`. Un ejemplo de salida de estado sería el siguiente:

```
Total Accesses: 17855
Total kBytes: 687
CPULoad: 14.1982
Uptime: 1221
ReqPerSec: 14.6233
BytesPerSec: 576.157
BytesPerReq: 39.4001
BusyServers: 8
IdleServers: 8
Scoreboard:
_KKWKKKKK.....................................................
..._..........................................................
....._........................................................
......_.......................................................
........_.....................................................
..........._..................................................
.............................................................
..............._..............................................
..............
```

Almacenar información del estado del servidor

Apache contiene un script Perl (lo puede encontrar en el directorio de soporte de la distribución fuente) llamado `log_server_status` que se puede utilizar para almacenar periódicamente la información del estado del servidor (utilizando la opción `auto`) en un archivo de texto.

Puede ejecutar este script como un trabajo del `cron` (demonio de Unix para la automatización de tareas) para aprovechar la información de estado, cada un cierto tiempo. Antes de que pueda utilizar el script, sin embargo, tiene que editar la fuente del script para modificar el valor de las variables `$wherelog`, `$port`, `$server` y `$request`. Los valores por defecto son:

```
$wherelog = "/var/log/graph/";
# Los registros serán del tipo "/var/log/graph/19960312"
$server = "localhost";
# Nombre del servidor, podría ser "www.foo.com"
$port = "80";
# Puerto del servidor
$request = "/status/?auto";
# Solicitud a enviar
```

En la mayor parte de los sitios funcionará lo siguiente:

```
$wherelog = "/var/log/apache";
$server   = "localhost";
$port     = "80";
$request  = "/server-status?auto"
```

Podría necesitar realizar los siguientes cambios:

- Cambiar el valor de `$wherelog` con la ruta en la que le gustaría almacenar el archivo creado por el script. Asegúrese de que la ruta existe ya o tendrá que crear utilizando el nombre de ruta `mkdir -p`. Por ejemplo, `mkdir -p /var/log/apache` asegurará que todos los directorios (`/var`, `/var/log`, `/var/log/apache`) se van creando a medida que se necesitan.
- El valor de la variable `$port` debe ser el número de puerto del servidor que quiere monitorizar. El valor por defecto 80 es perfecto si su servidor se está ejecutando en un puerto HTTP estándar.
- La variable `$server` debería asignarse al nombre del host de su servidor. El valor por defecto del host local es adecuado si el script y el servidor se ejecutan en el mismo sistema. Si el servidor se encuentra en otra máquina es necesario marcar como valor el nombre completo del host (por ejemplo, `www.mydomain.com`).
- La variable `$request` debería fijarse con lo que se utilice en la directiva `<Location . . .>` más el script de consulta `?auto`.

Si no le gusta el formato de registro que utiliza el script, puede modificar la siguiente línea para ajustar sus necesidades:

```
print OUT "$time:$requests:$idle:$number:$cpu\n";
```

El script utiliza una conexión socket con el servidor Apache para enviar las solicitudes URL; por lo tanto, necesita asegurarse que tiene soporte socket para Perl. Por ejemplo, en un sistema Linux, el código Perl del socket se encuentra en `socket.ph`. Puede utilizar este `socket.ph` para determinar si existe este archivo en su sistema.

Crear archivos de registro

Conocer la información del estado y de la configuración de su servidor es de gran ayuda a la hora de gestionar el servidor, pero conocer quién o qué está accediendo a su sitio o sitios Web es también muy importante, al tiempo que excitante. Puede obtener esta información utilizando las características de registro del servidor Apache. La siguiente sección discute cómo funciona el registro y cómo obtener el mejor beneficio de los módulos de registro de Apache.

Al tiempo que aparecieron los software de servidores Web en el mercado, fueron apareciendo muchos programas de análisis de registro en los servidores Web. Estos programas se convirtieron en parte del trabajo diario para muchos administradores Web. Todo esto conllevó a la era de las incompatibilidades de los archivos de registro, lo que hacía el análisis de registro difícil y engorroso; un solo programa de análisis no funcionaba en todos los archivos de registro. Entonces apareció la especificación *Common Log Format* (CLF). Esta especificación, permitía a todos los servidores Web, escribir registros de un modo razonablemente parecido, haciendo el análisis de registro entre servidores mucho más sencillo.

Por defecto, la distribución estándar de Apache incluye un módulo llamado `mod_log_config`, que es responsable del registro básico, y que escribe archivos de registro CLF por defecto. Puede cambiar su comportamiento utilizando la directiva `LogFormat`. Sin embargo, CLF cubre todas las necesidades en la mayoría de los entornos. A continuación se explica el contenido de cada línea de un archivo de registro CLF.

El archivo de registro CLF contiene una línea para cada solicitud. Una línea está compuesta por varias señales separadas por espacios:

```
host ident authuser date request status bytes
```

Si una señal no tiene un valor, entonces se representa por un guión (-). Las señales tienen los siguientes significados:

- `authuser`: si la URL solicitada requiere una autentificación HTTP Basic con éxito, entonces el nombre de usuario es el valor de esta señal.

- `bytes`: el número de bytes en el objeto devuelto al cliente, excluyendo todas las cabeceras HTTP.
- `date`: la fecha y hora de la última solicitud.
- `host`: el nombre completo de la máquina, dominio incluido, *fully qualified domain name*, del cliente, o su dirección IP.
- `ident`: si la directiva `IdentityCheck` está activada y la máquina del cliente ejecuta `identd`, entonces esta es la información que suministra el cliente.
- `request`: la línea de solicitud del cliente, encerrada entre dobles comillas (").
- `status`: el código de estado HTTP de tres dígitos que se devuelve al cliente.

Ver los apéndices para obtener una lista de todos los códigos de estado HTTP/1.1.

El campo de la fecha puede tener este formato:

```
date = [day/month/year:hour:minute:second zone]
```

El tamaño del campo de la fecha viene dado en la tabla 8.1.

Tabla 8.1. Tamaños de los campos

Campos	Valores
Día	2 dígitos.
Mes	3 letras.
Año	4 dígitos.
Hora	2 dígitos.
Minutos	2 dígitos.
Segundos	2 dígitos.
Zona	('+' | '-') 4*dígitos.

La siguiente sección ofrece un resumen de todas las directivas que puede utilizar con `mod_log_config`. Hay cuatro directivas disponibles en este módulo.

Directiva TransferLog

`TransferLog` asigna el nombre del archivo de registro o programa al que se envía la información de registro. Por defecto, la información de registro se en-

cuentra en el formato CFL. Este formato se puede personalizar utilizando la directiva `LogFormat`. Observe que cuando la directiva `TransferLog` se encuentra dentro de un contenedor de host virtual, la información de registro se formatea utilizando la última directiva `LogFormat` que se encuentre dentro del contexto. Si la directiva `LogFormat` no se encuentra dentro del mismo contexto, por el contrario, se utiliza el formato de registro del servidor.

Sintaxis: `TransferLog filename | "| path_to_external/program"`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

La directiva `TransferLog` toma como argumento o bien una ruta de un archivo de registro, o bien una tubería a un programa externo. Se considera que el nombre del archivo de registro es relativo al `ServerRoot` asignado si no se encuentra un carácter / inicial.

Por ejemplo, si `ServerRoot` está asignado a `/etc/httpd`, entonces la siguiente línea le dice a Apache que envíe la información de registro al archivo `/etc/httpd/logs/access.log`:

```
TransferLog logs/access.log
```

Cuando el argumento es una tubería a un programa externo. La información de registro se envía a la entrada estándar (`STDIN`) del programa externo.

NOTA: No se inicia un nuevo programa para un VirtualHost si este hereda el `TransferLog` del servidor principal. Si se utiliza un programa, entonces se ejecuta bajo el usuario que inició el `httpd`. Éste será la raíz si se inició el servidor como tal. Asegúrese de que el programa es seguro.

Directiva LogFormat

`LogFormat` determina el formato del archivo de registro nombrado en la directiva `TransferLog`. Si incluye un nickname para el formato en la línea de la directiva, puede utilizarlo en otras directivas `LogFormat` y `CustomLog` en vez de repetir la cadena completa del formato. Una directiva `LogFormat` que define un nickname no hace nada más; es decir, solo define el nickname, y no aplica realmente el formato.

Sintaxis: `LogFormat format [nickname]`

Predefinido: `LogFormat "%h %l %u %t \"%r\" %>s %b"`

Contexto: Configuración del servidor, host virtual

Ver la sección "Personalizar sus archivos de registro" en este mismo capítulo, para obtener los detalles de las opciones de formato disponibles.

Directiva CustomLog

Al igual que la directiva `TransferLog`, esta directiva le permite enviar información de registro a un archivo de registro o a un programa externo. A diferencia de `TransferLog`, sin embargo, le permite utilizar un formato de registro personalizado que se puede fijar como un argumento.

Sintaxis: `CustomLog file | pipe [format | nickname] [env=[!]environment_variable]`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

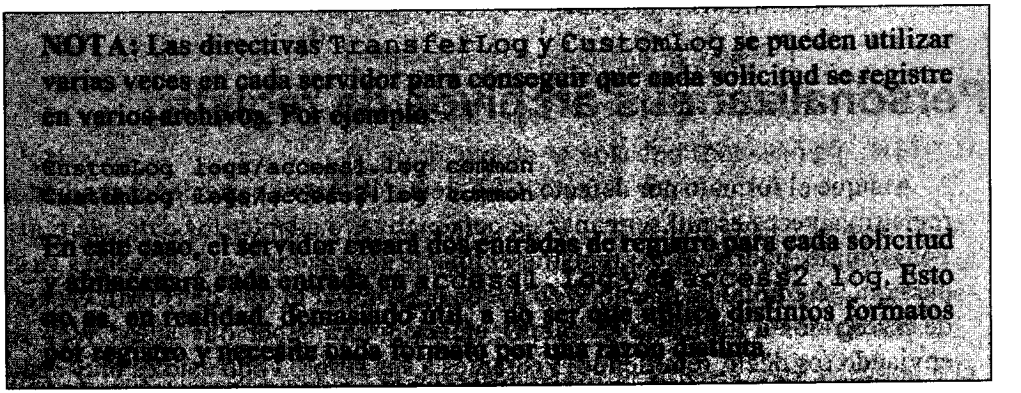
En el siguiente ejemplo, cada línea del archivo `access.log` se escribirá utilizando el formato los especificadores de formato dados. El formato determina un formato para cada línea del archivo de registro:

```
CustomLog logs/access.log "%h %l %u %t \"%r\" %>s %b"
```

Las opciones disponibles para el formato son exactamente las mismas que las que le sirven de argumento a la directiva `LogFormat`. Si el formato incluye cualquier espacio (lo cual ocurre en la mayoría de los casos), debe ir entre comillas dobles. En lugar de la cadena de formato real, puede utilizar un nickname definido con la directiva `LogFormat`. Por ejemplo:

```
LogFormat "%h %t \"%r\" %>s" myrecfmt
CustomLog logs/access.log myrecfmt
```

Aquí `access.log` tendrá líneas en el formato `myrecfmt`.



Para terminar, si utiliza el `mod_setenvif` (instalado por defecto) o el módulo para rescribir URL (`mod_rewrite`, que no está instalado por defecto)

para determinar las variables de entorno basadas en una URL solicitada, puede crear registros condicionales utilizando la opción `env=[!]environment_variable` con la directiva `CustomLog`. Por ejemplo, imagine que permite a la gente bajarse un papel blanco PDF y quiere registrar todas las bajadas en un archivo de registro llamado `whitepaper.log` en su directorio de registro habitual. A continuación tenemos la configuración necesaria:

```
SetEnvIf Request_URI \.pdf$ whitepaper
CustomLog logs/whitepaper.log common env=whitepaper
CustomLog logs/access.log common env=!whitepaper
```

La primera línea determina la variable de entorno `whitepaper` cada vez que una URL solicitada tiene la extensión `.pdf`. Entonces, cuando se va a registrar la entrada, Apache utiliza los ajustes `env=whitepaper` en la primera directiva `CommonLog` para determinar si está asignada. Si lo está, se crea una entrada de registro en el archivo `logs/whitepaper.log` utilizando el formato habitual. Cuando no está asignada la variable de entorno `whitepaper`, la entrada de registro se crea utilizando el archivo `logs/access.log` de la forma habitual.

Directiva CookieLog

`CookieLog` le permite registrar información sobre cookies en un archivo relacionado con la ruta de la directiva `ServerRoot`. No se recomienda esta directiva, porque no parece que Apache vaya a traer soporte para ella durante mucho tiempo. Para registrar datos sobre cookies, utilice el módulo de seguimiento de usuarios (`mod_usertrack`). El módulo de seguimiento de usuarios se discute más tarde en este capítulo.

Sintaxis: `CookieLog filename`

Predefinido: configuración del servidor, host virtual

Personalizar sus archivos de registro

Aunque el formato por defecto, CLF, reúne la mayor parte de los requisitos de registro, a veces es útil ser capaz de personalizar o adaptar los datos de registro. Por ejemplo, podría desear registrar el tipo de navegadores que acceden a su sitio, para que su equipo de diseño Web pueda determinar el tipo de HTML específico de navegador a utilizar o evitar. O, quizá quiera saber qué sitios Web están enviando (es decir, remitiendo) visitantes a sus sitios. Todo esto es muy sencillo con Apache. El módulo de registro por defecto, `mod_log_config`, soporta registros personalizados. Los formatos personalizados están asignados con las directivas `LogFormat` y `CustomLog` del módulo. `LogFormat` y `CustomLog`

tienen una cadena como argumento. Esta cadena puede tener tanto caracteres literales como especificadores de formato especiales %. Cuando se utilizan valores literales en esta cadena, se copian en un archivo de registro para cada solicitud. Los especificadores %, sin embargo, son reemplazados con los valores correspondientes. Los especificadores % especiales se muestran en la tabla 8.2.

Tabla 8.2. Especificadores % especiales para entradas de registro

Especificador %	Descripción
%a	Dirección IP del cliente.
%A	Dirección IP del servidor.
%B	Bytes enviados, excluyendo las cabeceras HTTP; 0 para un byte no enviado.
%b	Bytes enviados, excluyendo las cabeceras HTTP; para un byte no enviado.
%c	Estado de conexión cuando la respuesta tiene lugar. El carácter "X" se escribe si la conexión fue abortada por el cliente antes de que se completase la respuesta. Si un cliente utiliza el protocolo keep-alive, se escribe un "+" para mostrar que la conexión se ha mantenido después de agotar el tiempo de operación. Se escribe un "–" para indicar que se cerró la conexión tras la respuesta.
%{mycookie}C	El contenido de una cookie llamada `mycookie`.
%D	La cantidad de tiempo (en microsegundos) que tarda en completar la respuesta.
%{myenv}e	El contenido de una variable de entorno llamada `myenv`.
%f	El nombre de archivo de la solicitud.
%h	El host remoto que realiza la solicitud.
%H	El protocolo de la solicitud (por ejemplo, HTTP 1/1).
%{ IncomingHeader }I	El contenido de `IncomingHeader`; es decir, la línea o líneas de cabecera en la solicitud enviada al servidor. El carácter i (incoming) al final, se refiere a que se trata de la cabecera de un cliente.
%l	Si la directiva `IdentityCheck` está activada y la máquina del cliente ejecuta `identd`, entonces esta es la información de identidad que suministra el cliente.

Especificador %	Descripción
%m	El método de la solicitud (GET, POST, PUT, etc.).
%{ ModuleNote }n	El contenido de `ModuleNote` desde otro módulo.
%{ OutgoingHeader }o	El contenido de `OutgoingHeader`; es decir, la línea o líneas de cabecera en la respuesta. El carácter o (outgoing) al final quiere decir que se trata de una cabecera de un servidor.
%p	El puerto al que fue servida la solicitud.
%P	El ID del proceso del hijo que sirvió la solicitud.
%q	La cadena de consulta.
%r	La primera línea de la solicitud.
%s	El estado devuelto por el servidor en respuesta a la solicitud. Observe que cuando se redirige una solicitud, el valor de este especificador de formato continúa siendo el estado original de la solicitud. Si quiere almacenar el estado de la solicitud redirigida, utilice %>s.
%t	Tiempo de la solicitud. El formato de tiempo es el mismo que en el formato CLF.
%{format}t	El tiempo, en la forma dada por el formato. (Puede ver además la página man de `strftime` en los sistemas Unix .)
%T	El tiempo que tarda el servidor en contestar, en segundos.
%u	Si la URL solicitada, necesita una autentificación HTTP `Basic` con éxito, entonces el nombre de usuario es el valor de este especificador de formato. El valor debería ser falso si el servidor devuelve un estado 401 (Necesita autentificación) después del intento de autentificación.
%U	La ruta de la URL solicitada.
%v	El nombre del servidor o del host virtual de la solicitud.
%V	El nombre del servidor para cada directiva `UseCanonicalName`.

Es posible incluir información condicional en cada uno de los especificadores que hemos visto. Las condiciones pueden ser la presencia (o ausencia) de ciertos

códigos de estados HTTP. Por ejemplo, imagine que quiere registrar todas las URL que dirigen a un usuario a una página que no existe. En ese caso, el servidor produce una cabecera de estado 404 (No encuentra). Por lo tanto, para registrar estas URL puede utilizar el especificador de formato:

```
'%404{Referer}i'
```

De igual modo, para registrar las URL resultantes de un estado inusual, puede utilizar:

```
'%!200,304,302{Referer}i'
```

Observe la utilización del carácter ! para indicar la ausencia de la lista de los estados del servidor. De igual modo, para incluir información adicional al final del especificador de formato CLF, puede extender el formato CLF, que es definido por la cadena de formato:

```
"%h %l %u %t \"%r\" %s %b"
```

Por ejemplo:

```
"%h %l %u %t \"%r\" %s %b \"%{Referer}i\" \"%{User-agent}i\"".
```

Estos registros de especificación CLF, formatean datos y añaden la información Referer y User-agent encontrada en las cabeceras proporcionadas por el cliente en cada entrada de registro. Hemos visto cómo añadir campos personalizados para el archivo de registro pero no sabemos qué es lo que ocurre si necesita almacenar estos datos en más de un archivo de registro. La siguiente sección discute cómo utilizar varios archivos de registro.

Crear varios archivos de registro

En ocasiones, es necesario crear varios archivos de registro. Por ejemplo, si está utilizando un programa de análisis de registro que no puede manejar datos que no sean CLF, tendrá que escribir los datos que no son CLF en un archivo distinto. Puede crear varios archivos de registro fácilmente utilizando la directiva `TransferLog` y/ o la directiva `CustomLog` del módulo `mod_log_config`. Simplemente repita estas directivas para crear más de un archivo de registro.

Si, por ejemplo, quiere crear un archivo de acceso CLF estándar y un archivo personalizado de todas las URL, puede utilizar algo parecido a esto:

```
TransferLog  logs/access_log
CustomLog    logs/referrer_log      "%{Referer}i"
```

Cuando tiene definidas `TransferLog` o `CustomLog` en la configuración del servidor principal, y tiene un host virtual definido, el registro relacionado con el host también se lleva a cabo en estos registros. Por ejemplo:

```
TransferLog  logs/access_log
CustomLog    logs/agents_log      "%{User-agent}i"

<Virtual Host 206.171.50.51>

    ServerName reboot.nitec.com
    DocumentRoot   "/www/reboot/public/htdocs"
    ScriptAlias /cgi-bin/ "/www/reboot/public/cgi-bin/"

</VirtualHost>
```

El host virtual `reboot.nitec.com` no tiene definida una directiva `TransferLog` o una directiva `CustomLog` dentro de las etiquetas del host virtual. Toda la información de registro se almacenará en `logs/access_log` y en `logs/agents_log`. Si se añade la línea siguiente dentro del contenedor del host virtual:

```
TransferLog  vhost_logs/reboot_access_log
```

Todos los registros del host virtual `reboot.nitec.com` se realizan en el archivo `vhost_logs/reboot_access_log`. No se utilizarán los archivos `logs/access_log` y `logs/agents_log` con el host virtual `reboot.nitec.com`.

Registrar cookies

Hasta ahora, las opciones de registro discutidas, no le permiten identificar visitantes. La identificación de visitantes es muy importante, porque si sabe quién está realizando qué solicitudes, se puede hacer una idea de cómo se utiliza su contenido. Por ejemplo, imagine que tiene una página realmente buena es algún sitio de su página Web, y tiene algún modo de identificar a los visitantes en su registro. Si observa el registro y puede ver que muchos visitantes van de una página a otra hasta encontrar dicha página, podría reconsiderar su diseño Web y dejar la página más accesible. Apache tiene un módulo llamado `mod_usertrack` que le permite hacer un seguimiento de los visitantes a su sitio Web registrando las cookies HTTP.

COOKIES HTTP: Una cookie HTTP es simplemente una pieza de información que el servidor le pasa al navegador Web. Esta información se almacena normalmente en un par clave = valor y puede ir asociada con todo el sitio Web o con una URL particular en el sitio Web. Una vez que el servidor envía una cookie y el navegador la acepta, la cookie residirá en el sistema del navegador Web. Cada vez que el navegador Web solicite la misma URL, o cualquier URL que se encuentre bajo el alcance de la URL

de la cookie, la información de la cookie se devuelve al servidor. Cuando asignamos la cookie, el servidor puede decirle al navegador Web que expire la cookie tras un tiempo determinado. El tiempo se puede determinar de modo que la cookie nunca se utilice en una sesión posterior, o se puede utilizar durante un largo período de tiempo.
Ha habido una gran controversia con la utilización de las cookies. Mucha gente considera las cookies como una intrusión de la privacidad. Utilizar cookies para hacer el seguimiento del comportamiento de usuarios es muy popular. De hecho, muchas compañías de anuncios de Internet hacen un gran uso de cookies para llevar a cabo el seguimiento de usuarios.

Los datos de las cookies se escriben normalmente en un archivo de texto en un directorio de su software del navegador. Por ejemplo, utilizando la directiva `CustomLog` en el módulo de registro estándar, puede almacenar las cookies en archivos separados:

```
CustomLog logs/clickstream "%{cookie}C %r %t"
```

A continuación, vamos a ver el nuevo módulo `mod_usertrack`.

Recuerde que `mod_usertrack` no guarda un registro de cookies; simplemente genera cookies únicas para cada visitante. Puede utilizar `CustomLog` (como se vio antes) para almacenar estas cookies en un archivo de registro para su análisis.

La directiva `mod_usertrack` no está compilada en la versión estándar de la distribución de Apache, por lo que necesita compilarla utilizando la opción `--enable-usertrack` antes de utilizarla. El módulo proporciona las directivas de las secciones siguientes.

Directiva CookieExpires

Esta directiva se utiliza para determinar el período de expiración de las cokies que son generadas por este módulo. El período de expiración se puede definir en términos de número de segundos, o en un formato del tipo "1 month 2 days 3 hours."

Sintaxis: `CookieExpires expiry-period`

Contexto: configuración del servidor, host virtual

En el ejemplo siguiente, la primera directiva define el tiempo de expiración en segundos, y la segunda directiva define el tiempo de expiración utilizando el formato especial.

Observe que cuando el tiempo de expiración no está definido en un formato numérico, se asume el formato especial. Sin embargo, el formato especial requiere que coloque dobles comillas alrededor de la cadena de formato. Si no se utiliza esta directiva, las cookies sólo aguantan esa sesión del navegador.

```
CookieExpires 3600
CookieExpires "2 days 3 hours"
```

Directiva CookieTracking

Esta directiva activa o desactiva la generación automática de cookies. Cuando tiene asignado el valor `on`, Apache comienza a enviar cookies de seguimiento de usuarios para cada nueva solicitud. Esta directiva se puede utilizar para activar o desactivar este comportamiento en servidores o en directorios. Por defecto, compilar `mod_usertrack` no activa las cookies.

Sintaxis: `CookieTracking On | Off`

Contexto: configuración del servidor, host virtual, directorio, archivo control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `FileInfo`

Utilizar registros de error

Este capítulo ha discutido varias formas de registrar datos interesantes y fases de respuesta en cada transacción Web. Cuantos más datos obtenga sobre sus visitantes, más contento estará su departamento de marketing. Como administrador del sistema, sin embargo, estará feliz si todo marcha sin problemas. Apache le permite saber qué es lo que no funciona escribiendo registros de error. Sin registros de error, no será capaz de determinar qué es lo que va mal y dónde está teniendo lugar el error.

No es extraño que ese registro de error esté soportado por Apache en lugar de en un módulo del tipo `mod_log_config`.

La directiva `ErrorLog` le permite registrar todos los errores que encuentra Apache. Esta sección explora el modo en el que puede incorporar su registro de errores Apache en la facilidad `syslog` que se encuentra en la mayor parte de las plataformas Unix.

`Syslog` es el modo tradicional de registrar mensajes enviados por los procesos demonio (servidor). Podría preguntarse que si Apache es un demonio, por qué no puede escribir en `syslog`. De hecho, puede hacerlo. Todo lo que necesita hacer es reemplazar su directiva `ErrorLog` en el archivo de configuración por:

```
ErrorLog syslog
```

y reiniciar Apache. Utilizando un navegador Web, acceda a una página que no exista y compruebe si el archivo de registro `syslog` muestra una entrada `httpd`. Debería mirar su archivo `/etc/syslog.conf` para encontrar pistas sobre dónde aparecerán los mensajes `httpd`.

Por ejemplo, el listado 8.1 muestra `/etc/syslog.conf` para un sistema Linux.

Listado 8.1. /etc/syslog.conf

```
# Registra todos los mensajes del kernel para la consola.
# Registra toda la pantalla.
#kern.*                         /dev/console

# Registra todo (excepto correo) del nivel de información o
superior.
# No registra mensajes privados de autentificación
*.info;mail.none;authpriv.none      /var/log/messages

# El archivo authpriv tiene acceso restringido.
authpriv.*              /var/log/secure

# Registra todos los mensajes de correo en un solo sitio.
mail.*                  /var/log/maillog

# Todo el mundo obtiene mensajes de emergencia,
# los registra en otra máquina.
*.emerg                 *

# Mantiene los errores en correo y en noticias en un nivel err
# y superior, en un archivo especial.
uucp,news.crit          /var/log/spooler

# Mantiene mensajes boot en boot.log
local7.*                /var/log/boot.log
```

Hay dos líneas importantes (relacionadas con Apache) en el listado, que he marcado en negrita.

La primera línea (que empieza con `*.info;mail.none;`) le dice a `syslog` que escriba todos los mensajes del tipo informativo (excepto para autentificación de correo y privada) en el archivo `/var/log/messages`, y la segunda línea (que empieza con `*.emerg`) determina que todos los mensajes de emergencia deben escribirse en todos los archivos de registro. Utilizando la directiva `LogLevel`, puede especificar qué tipo de mensajes Apache deberían enviarse a `syslog`. Por ejemplo:

```
ErrorLog syslog
LogLevel debug
```

Le está diciendo a Apache que envíe mensajes de depuración de errores a `syslog`. Si quiere almacenar mensajes de depuración de errores en un archivo distinto mediante `syslog`, entonces tiene que modificar `/etc/syslog.conf`. Por ejemplo:

```
*.debug             /var/log/debug
```

Apache le permitirá guardar todos los mensajes de depuración en el archivo `/var/log/debug` si añade esta línea en `/etc/syslog.conf` y reinicia

`syslogd` (`kill -HUP syslogd_PID`). Hay varios niveles de registro asignados:

- **Alert:** mensajes de alerta
- **Crit:** mensajes críticos
- **Debug:** mensajes registrados al nivel de depuración de errores que incluirán el archivo fuente y el número de línea en el que se genera el mensaje, para ayudar a la depuración de errores y al desarrollo del código
- **Emerg:** mensajes de emergencia
- **Error:** mensajes de errores
- **Info:** mensajes informativos
- **Notice:** mensajes de notificación
- **Warn:** advertencias

TRUCO: Si quiere ver las actualizaciones en `syslog` o en cualquier otro archivo de registro mientras que están ocurriendo, puede utilizar la utilidad `tail` que se encuentra en la mayoría de los sistemas Unix. Por ejemplo, si quiere ver las actualizaciones de un registro llamado `/var/log/messages` mientras que está ocurriendo, utilice:

```
tail -f /var/log/messages
```

Analizar sus archivos de registro

Hasta ahora, ha aprendido a crear registros estándar basados en CLF y registros personalizados. Ahora, necesita un modo de analizar estos registros para utilizar los datos registrados. Sus necesidades de análisis pueden variar. En ocasiones necesitará producir largos informes, o quizá simplemente quiera verificar los registros. Para tareas sencillas, es mejor utilizar cualquiera que tenga a mano. La mayor parte de los sistema Unix tienen utilidades suficientes y herramientas de script disponibles para llevar a cabo este trabajo.

Utilizando las utilidades Unix, puede recoger rápidamente la información necesaria; sin embargo, este método requiere algunos conocimientos de Unix, y no siempre es conveniente porque su jefe podría querer un informe "bonito" en lugar de una simple lista de texto. En tal caso, puede desarrollar su propio programa de análisis o utilizar una herramienta de análisis de terceras partes.

Vamos a emplear una utilidad Unix para obtener una lista de todos los host. Si usa la utilidad de registro por defecto o un registro personalizado con soporte CLF, puede encontrar fácilmente una lista de todos los host. Por ejemplo:

```
cat /path/to/httpd/access_log | awk '{print $1}'
```

imprime todas las direcciones IP (si tiene activada una búsqueda DNS [domain name server], cuando se muestran los alias de los host). La utilidad `cat` realiza una lista del archivo `access_log`, y el resultado se dirige al interpretador `awk`, que imprime únicamente el primer campo en cada línea utilizando la sentencia `print`. Esta sentencia imprime todos los host; pero ¿qué es lo que ocurre si quiere excluir los host de su red? En ese caso, debería utilizar:

```
cat /path/to/httpd/access_log | awk '{print $1}'  | egrep -v
'(^206.171.50)'
```

donde `206.171.50` debería reemplazarse con su dirección de red. He supuesto que tiene una red de tipo C. Si tiene una red de tipo B, sólo tiene que utilizar los primeros dos octetos de su dirección IP. Esta versión le permite excluir sus propios host utilizando la utilidad `egrep`, a la que se le indica que muestre (vía `-v`) sólo los host que no comienzan con la dirección de red `206.171.50`. Sin embargo, esto no acaba de resultar totalmente satisfactorio, porque hay muchas posibilidades de repetición. Por lo tanto, la versión final es:

```
cat /path/to/httpd/access_log | awk '{print $1}'  | uniq |
egrep -v '(^206.171.50)'
```

La utilidad `uniq` filtra las repeticiones y muestra solo una lista por cada host. Por supuesto, si quiere ver el número total de host que tienen acceso a su sitio Web, puede dirigir el resultado final a la utilidad `wc` con la opción `-l`:

```
cat /path/to/httpd/access_log | awk '{print $1}' | \
uniq | egrep -v '(^206.171.50)' | wc -l
```

Esto le da el número total de conteo de líneas (es decir, el número de accesos del host).

Hay disponibles muchas herramientas de análisis de registro de servidores Web de terceras partes. La mayoría de estas herramientas esperan que los archivos de registro se encuentren en el formato CLF, por lo que es necesario que se asegure de tener un formato CLF en sus registros. La tabla 8.3 contiene una lista de estas herramientas y dónde encontrarlas.

Tabla 8.3. Herramientas de análisis de terceras partes

Nombre del producto	URL del producto
WebTrends	www.webtrends.com/
Wusage	www.boutell.com/wusage/
Wwwstat	www.ics.uci.edu/pub/websoft/wwwstat/
Analog	www.statslab.cam.ac.uk/~sret1/analog/

Nombre del producto	URL del producto
http-analyze	www.netstore.de/Supply/http-analyze/
Pwebstats	www.unimelb.edu.au/pwebstats.html
WebStat Explorer	www.webstat.com/
AccessWatch	http://netpressence.com/accesswatch/

El mejor modo de aprender qué herramienta le va a funcionar es probarlas todas, o al menos visitar sus sitios Web para comparar sus características. Dos utilidades que encuentro muy prácticas son Wusage y wwwstat. Wusage es mi aplicación de análisis de registro favorita. Es muy configurable y produce informes gráficos de alta calidad utilizando la librería de gráficos GD de la compañía. Wusage se distribuye en formato binario. Las copias de evaluación de wusage se distribuyen gratuitamente para varias plataformas Unix y Windows.

wwwstat es uno de mis programas de análisis gratuitos preferidos. Está escrito en Perl, por lo tanto, tiene que tener Perl instalado en el sistema en el que quiera ejecutar esta aplicación. Puede leer los resúmenes de salidas wwwstat con `gwstat` para producir gráficos de alta calidad en las estadísticas resumidas.

Crear registros en Apache es sencillo y útil. Crear registros le permite aprender más sobre lo que está ocurriendo en su servidor Apache. Los registros le ayudan a detectar e identificar los problemas de su sitio Web, a determinar las mejores características de su sitio Web, y mucho más. Pero debe haber alguna trampa. Lo que ocurre es que los archivos de registro ocupan mucho espacio disponible en el disco, por lo que es necesario un mantenimiento muy regular.

Mantenimiento de registros

Permitiendo el registro, será capaz de ahorrar mucho trabajo, pero los registros en sí, le van a añadir trabajo extra: ha de mantenerlos. En los sitios Apache con muchos dominios virtuales, los archivos de registro se pueden convertir en enormes en poco tiempo, lo que puede causar fácilmente una crisis en el disco. Cuando los archivos de registro se vuelven muy grandes, debe rotarlos.

Tiene dos opciones para rotar sus registros: puede utilizar la utilidad de Apache llamada `rotatelog`, o puede utilizar `logrotate`, una facilidad disponible en la mayoría de los sistemas Linux.

Utilizar rotatelog

Apache tiene soporte para una utilidad llamada `rotatelog`. Puede utilizar este programa del siguiente modo:

```
TransferLog "| /path/to/rotatelogs logfile
rotation_time_in_seconds>"
```

Por ejemplo, si quiere rotar el registro de acceso cada 86.400 segundos (es decir, 24 horas), utilice la línea siguiente:

```
TransferLog "| /path/to/rotatelogs /var/logs/httpd 86400"
```

Cada acceso diario a la información de registro se almacenará en un archivo llamado `/var/logs/httpd.nnnn`, donde `nnnn` representa un número grande.

Utilizar logrotate

La utilidad `logrotate` rota, comprime y envía archivos de registro. Está diseñada para facilitar la administración del sistema de archivos de registro. Permite la rotación, compresión, eliminación y envío automáticos de los archivos de registro a diario, mensualmente, o basándose en el tamaño. Normalmente, `logrotate` se ejecuta como un trabajo diario del cron (demonio de Unix para la automatización de tareas). Lea las páginas de información de `logrotate` para aprender más sobre él.

Si su sistema soporta la utilidad `logrotate`, debería crear un script llamado `/etc/logrotate.d/apache` tal y como se muestra en el listado 8.2.

Listado 8.2. /etc/logrotate.d/apache

```
# Observe que este script supone que:
#
# a. Tiene instalado Apache en /usr/local/apache
# b. Su ruta de registro es /usr/local/apache/logs
# c. Su registro de acceso se llama access_log (por defecto en
Apache)
# d. Su registro de error se llama error_log (por defecto)
# e. el archivo PID, httpd.pid, para Apache, se almacena en el
#    directorio log (por defecto en Apache)
#
# Si alguna de estas suposiciones son falsas, por favor cambie
# la ruta o el nombre de archivo.
#
/usr/local/apache/logs/access_log {
    missingok

    compress
    rotate 5
    mail webmaster@yourdomain.com
    errors webmaster@yourdomain.com
    size=10240K

    postrotate
        /bin/kill -HUP `cat /usr/local/apache/logs/httpd.pid
2>/dev/null` 2> /dev/null || true
```

```
    endscript
}

/usr/local/apache/logs/error_log {
    missingok

    compress
    rotate 5
    mail webmaster@yourdomain.com
    errors webmaster@yourdomain.com
    size=10240K


    postrotate
        /bin/kill -HUP `cat /usr/local/apache/logs/httpd.pid
2>/dev/null` 2> /dev/null || true
    endscript
}
```

Esta configuración determina que tanto los accesos como los archivos de registro de error de Apache sean rotados cada vez que crece por encima de 10MB (10,240K), y que todos los archivos antiguos de registro sean comprimidos y enviados a `webmaster@yourdomain.com` una vez que han pasado por cinco rotaciones, en lugar de ser eliminados. Cualquier error que tenga lugar durante el procesamiento de los archivos de registros se envía a `root@yourdomain.com`.

Utilizar logresolve

Por razones de rendimiento, debería deshabilitar la búsqueda de nombre de usuarios utilizando la directiva `HostNameLookups` con el valor `off`. Esto significa que sus entradas de registro mostrarán direcciones IP en lugar de nombres de host para los clientes remotos. Cuando se analizan los registros, es de gran ayuda tener los nombres de host de modo que pueda determinar quién a accede a qué lugar fácilmente. Por ejemplo, a continuación tenemos unos cuantos ejemplos de entradas de registro en el archivo `/usr/local/apache/logs/access_log`.

```
207.183.233.19 - - [15/Mar/2001:13:05:01 -0800] "GET /book/
images/back.gif HTTP/1.1" 304 0
207.183.233.20 - - [15/Mar/2001:14:45:02 -0800] "GET /book/
images/forward.gif HTTP/1.1" 304 0
207.183.233.21 - - [15/Mar/2001:15:30:03 -0800] "GET /book/
images/top.gif HTTP/1.1" 304 0
```

Si tiene activada `HostNameLookups`, Apache traducirá las direcciones IP del cliente `207.183.233.19`, `207.183.233.20` y `207.183.233.21` a los nombres de host apropiados; y si deja `LogFormat`:

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

y utiliza el formato común en registro utilizando `CustomLog logs/access_log`, las muestras serán las siguientes:

```
nano.nitec.com - - [15/Mar/2001:13:05:01 -0800] "GET /book/
images/back.gif HTTP/1.1" 304 0
rhat.nitec.com - - [15/Mar/2001:14:45:02 -0800] "GET /book/
images/forward.gif HTTP/1.1" 304 0
r2d2.nitec.com - - [15/Mar/2001:15:30:03 -0800] "GET /book/
images/top.gif HTTP/1.1" 304 0
```

Como activar la búsqueda DNS da lugar a que el servidor Apache tarde más tiempo en completar la respuesta, se recomienda que la búsqueda de nombres de host se realice de forma separada utilizando la utilidad `logresolve`, que se puede encontrar en el directorio bin (`/usr/local/apache/bin`) de Apache. El script `log_resolver.sh` que se muestra en el listado 8.3 puede ejecutar esta utilidad.

Listado 8.3. log_resolver.sh

```
#!/bin/sh

#
# Asegúrese de que cambia los nombres de las rutas según
# su instalación Apache
#

# Fully qualified path name (FQPN) de la
# utilidad log-resolver
LOGRESOLVER=/usr/local/apache/bin/logresolve

# Archivo estático generado por la utilidad
STATFILE=/tmp/log_stats.txt

# Archivo de registro de Apache
LOGFILE=/usr/local/apache/logs/access_log

# Nuevo archivo de registro que tiene traducida la dirección IP
OUTFILE=/usr/local/apache/logs/access_log.resolved

# Ejecuta el comando
$LOGRESOLVER -s $STATFILE < $LOGFILE > $OUTFILE

exit 0;
```

Cuando se ejecuta este script desde la línea de comando o desde un trabajo `cron`, crea un archivo llamado `/usr/local/apache/logs/access_log.resolved`, que tiene todas las direcciones IP traducidas a sus nombres de host respectivos. Además, el script genera un archivo estático llamado `/tmp/log_stats.txt` que muestra la información sobre el uso del caché,

las direcciones IP traducidas, y otra información de la utilidad `resolver`. A continuación se muestra un ejemplo de este tipo de archivos estáticos:

```
logresolve Statistics:
Entries: 3
    With name    : 0
    Resolves     : 3
Cache hits       : 0
Cache size       : 3
Cache buckets    :      IP number * hostname
  130    207.183.233.19 - nano.nitec.com
  131    207.183.233.20 - rhat.nitec.com
  132    207.183.233.21 - r2d2.nitec.com
```

Observe que la utilidad no puede usar el caché porque las tres direcciones IP que están traducidas (para las entradas de registro mostradas) son únicas. Sin embargo, si su archivo de registro tiene direcciones IP del mismo host, se utilizará el caché para traducirlas en lugar de realizar solicitudes DNS a ciegas.

Si piensa que puede utilizar este script, le recomiendo que lo ejecute como un trabajo del `cron`. Por ejemplo, en mi servidor Web Apache ejecutando Linux, simplemente añado el script a `/etc/cron.daily` para crear una versión traducida del registro diario.

9 Reescribir las URL

En este capítulo

1. Trabajamos con el motor de reescritura de URL de Apache.
2. Analizamos la distribución URL.
3. Manejamos contenido.
4. Restringimos el acceso.

Las URL llevan a los visitantes a su sitio Web. Como administrador Apache, necesita asegurar que todas las URL posibles de su sitio Web son funcionales. ¿Cómo se hace? Ha de mantener la monitorización de los registros de error para todas las solicitudes URL fallidas. Si observa que las solicitudes se están devolviendo con un código de estado 404 Not Found, es el momento de investigar estas URL. A menudo, cuando el autor de un documento HTML actualiza un sitio Web, olvida que volver a nombrar un directorio puede dar lugar a la pérdida de muchos visitantes debido a la pérdida de coherencia en la carpeta de favoritos de estos.

Como administrador, ¿cómo podría resolver este problema? Las buenas noticias son que hay un módulo llamado `mod_rewrite` que le permite resolver

estos problemas y crear, además, soluciones muy interesantes utilizando reglas de reescritura de URL. Este capítulo discute este módulo y proporciona ejemplos prácticos de reescritura de URL.

El motor de reescritura de URL de Apache

Cuando Apache recibe una solicitud URL, la procesa sirviendo el archivo al cliente (el navegador Web). ¿Qué ocurre si quiere intervenir en este proceso de integración de una URL a un archivo distinto o incluso a una URL distinta? Aquí es donde el módulo `mod_rewrite` muestra su valor. Le proporciona un mecanismo flexible para reescribir la URL solicitada en una nueva utilizando las reglas personalizadas de reescritura de URL. Una regla de reescritura de URL tiene la forma siguiente:

```
regex_pattern_to_be_matched
regex_substitution_pattern
```

Sin embargo, también es posible añadir condiciones a una regla, de modo que la substitución sólo se aplique si se encuentra la condición. Apache puede manejar estas URL sustitutas como si fueran subsolicitudes internas, o puede devolverlas al navegador Web como si fueran redirecciones externas. La figura 9.1 muestra un ejemplo en el que aparece una solicitud y el resultado de una regla `mod_rewrite`.

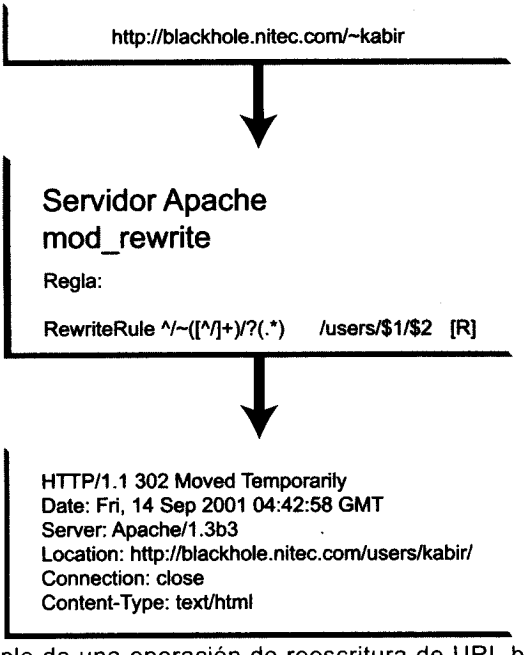


Figura 9.1. Ejemplo de una operación de reescritura de URL basada en reglas

La figura muestra una solicitud de `http://blackhole.nitec.com/~kabir` realizada desde el servidor Apache. El servidor recibe la solicitud y la pasa al módulo `mod_rewrite`, a la fase de traducción de la URL dentro del procesamiento de la solicitud. El módulo `mod_rewrite` aplica la regla de reescritura definida por una directiva llamada `RewriteRule`. En este ejemplo particular, la regla determina que si se encuentra un patrón del tipo `/~([^/]+)/?(.*)`, debe ser reemplazado por `/users/$1/$2`. Como hay una indicación de redireccionamiento `[R]` en la regla, se enviará también al navegador Web una respuesta de redirección URL externa. La salida muestra como redirección a `http://blackhole.nitec.com/users/kabir/`.

Como puede ver, este tipo de redireccionamiento puede venir muy bien en muchas situaciones. Vamos a ver las directivas que le dan la opción de rescribir las URL. Debería familiarizarse también con las variables de servidor que se muestran en la tabla 9.1, que se pueden utilizar en muchas reglas y condiciones de reescritura.

Tabla 9.1. Variables del servidor disponibles para las reglas de reescritura de URL

Variable	Explicación
SERVER_NAME	Nombre del servidor en el host.
SERVER_ADMIN	Administrador de las direcciones de correo en el servidor Web.
SERVER_PORT	Dirección del puerto del servidor Web.
SERVER_PROTOCOL	Versión del protocolo HTTP que utiliza el servidor Web.
SERVER_SOFTWARE	Nombre del fabricante del servidor Web.
SERVER_VERSION	Versión del software del servidor Web.
DOCUMENT_ROOT	Directorio de documentos de máximo nivel en el sitio Web.
HTTP_ACCEPT MIME	Tipos aceptables para el cliente Web.
HTTP_COOKIE	Cookie recibida desde el cliente Web.
HTTP_FORWARDED	URL enviada.
HTTP_HOST	Nombre del host del servidor Web.
HTTP_PROXY_CONNECTION	La información de la conexión proxy HTTP.
HTTP_REFERER	La URL que se refiere a la URL actual.
HTTP_USER_AGENT	Información sobre el cliente Web.
REMOTE_ADDR	Dirección IP del cliente Web.

Variable	Explicación
REMOTE_HOST	Nombre del host del cliente Web.
REMOTE_USER	Nombre de usuario del usuario autentificado.
REMOTE_IDENT	Información sobre la identificación del usuario remoto.
REQUEST_METHOD	Método de solicitud HTTP utilizado para solicitar la URL actual.
SCRIPT_FILENAME	Ruta física del archivo de script solicitado.
PATH_INFO	Ruta de la URL solicitada.
QUERY_STRING	Datos de consulta enviados junto con la URL solicitada.
AUTH_TYPE	Tipo de autentificación utilizada.
REQUEST_URI	URI solicitado.
REQUEST_FILENAME	Igual que `SCRIPT_FILENAME`.
THE_REQUEST	URL solicitada.
TIME_YEAR	Año actual.
TIME_MON	Mes actual.
TIME_DAY	Día actual.
TIME_HOUR	Hora actual.
TIME_MIN	Minuto actual.
TIME_SEC	Segundo actual.
TIME_WDAY	Día de la semana actual.
TIME	Momento actual.
API_VERSION	Versión del API utilizado.
IS_SUBREQ	Determina si la solicitud es una subsolicitud.

RewriteEngine

Esta directiva le proporciona un mecanismo de activación/desactivación para el motor de reescritura de URL en el módulo `mod_rewrite`. Por defecto, la reescritura está desactivada. Para utilizar el motor de reescritura, debe activar el motor asignándole a esta directiva el valor `on`.

Sintaxis: `RewriteEngine On | Off`

Predefinido: `RewriteEngine Off`

Contexto: configuración del servidor, host virtual, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Cuando se permite la reescritura de URL en la configuración del archivo de directorios (`.htaccess`), debe activar (es decir, darle el valor `On`) a esta directiva dentro de cada archivo de configuración de directorios y asegurarse de que está activando la siguiente directiva en el contexto adecuado para el directorio:

```
Options FollowSymLinks
```

En otras palabras, si el directorio pertenece a un sitio de un host virtual, asegúrese de que esta opción está activada dentro del contenedor de host adecuado. Del mismo modo, si el directorio en cuestión forma parte del espacio de documentos del servidor Web principal, asegúrese de que esta opción está activada en la configuración del servidor principal.

NOTA: Activar las reglas de reescritura a nivel de las configuraciones de directorio podría disminuir el rendimiento de su servidor Apache. Esto se debe a que `mod_rewrite` emplea un truco para soportar reglas de escritura a nivel de directorio, y este truco implica el aumento de la carga de procesos en el servidor. Por tanto, debe evitar utilizar reglas de reescritura en los archivos de configuración a nivel de directorio siempre que sea posible.

RewriteOptions

Esta directiva le permite determinar opciones para cambiar el comportamiento del motor de reescritura. Actualmente, la única opción disponible es `inherit`. Dándole el valor `inherit` a esta directiva, puede forzar que una configuración de bajo nivel herede la configuración de mayor nivel.

Sintaxis: `RewriteOptions option1 option2 [...]`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Por ejemplo, si asigna esta directiva en el área de configuración de su servidor principal, un host virtual definido en el archivo de configuración heredará toda la configuración rescrita, así como las reglas de reescritura, las condiciones, las integraciones, y similares. De igual modo, cuando esta directiva se asigna en un archivo de configuración en el ámbito y las integraciones, por defecto, el motor de reescritura no permite heredar una configuración rescrita, pero esta directiva le permite alterar el comportamiento por defecto.

RewriteRule

Esta directiva le permite definir una regla de reescritura. La regla debe tener dos argumentos. El primer argumento es el patrón buscado que debe cumplir para aplicar la cadena de sustitución. El patrón buscado se escribe utilizando una expresión regular (ver apéndice B para obtener los conceptos básicos sobre las expresiones regulares). La cadena de sustitución puede construirse con todo texto, con referencias a las subcadenas en el patrón de búsqueda, con valores para las variables del servidor, e incluso con funciones map. La lista de indicadores puede contener una o más cadenas de indicadores, separados por comas, para informar al motor de reescritura sobre que es lo que hay que hacer después con la sustitución.

> **Sintaxis:** `RewriteRule search_pattern substitution_string [flag_list]`
>
> **Predefinido:** ninguno
>
> **Contexto:** configuración del servidor, host virtual, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Vamos a ver un ejemplo:

```
RewriteRule /~([^/]+)/?(.*)   /users/$1/$2 [R]
```

Aquí, el patrón de búsqueda es `/~([^/]+)/?(.*)` y la cadena de sustitución es `/users/$1/$2`. Observe la utilización de referencias en la cadena de sustitución. La primera cadena de referencia `$1` corresponde a la cadena encontrada en el primer conjunto de paréntesis (desde la izquierda). Por lo tanto `$1` está asignado a cualquiera que coincida con `([^/]+)` y `$2` con la siguiente cadena encontrada en `(.*)`. Cuando una solicitud URL es como la siguiente:

```
http://blackhole.evoknow.com/~kabir/welcome.html
```

El valor de `$1` es `kabir`, y `$2` es `welcome.html`; por lo que la cadena de sustitución será la siguiente:

```
/users/kabir/welcome.html
```

Cuando tiene especificada más de una `RewriteRule`, la primera `RewriteRule` opera en la URL original y si tiene lugar una coincidencia, la segunda regla no vuelve a operar en la URL original. En lugar de eso, toma la URL sustituida por la primera regla como la URL en la que ha de aplicar las reglas.

En un escenario en el que hay una coincidencia a cada paso, una asignación de tres reglas de reescritura funcionará del siguiente modo:

```
RewriteRule search-pattern-for-original-URL substitution1
[flags]
```

```
RewriteRule  search-pattern-for-substitution1 substitution2
[flags]
RewriteRule  search-pattern-for-substitution2 substitution3
[flags]
```

Es posible aplicar más de una regla a la URL original utilizando el indicador C para decirle al motor de reescritura cómo encadenar varias reglas. En este caso, no debería realizar una sustitución hasta que se hayan aplicado todas las reglas, de modo que pueda utilizar una cadena especial de sustitución para permitir una sustitución en una regla. La tabla 9.2 tiene una lista de los detalles de los indicadores posibles.

Tabla 9.2. Indicadores RewriteRule

Indicador	Significado
C | chain	Este indicador determina que la regla actual se encadene con la siguiente regla. Cuando una regla se encadena con un indicador C, sólo se utiliza si ha habido coincidencia con la regla anterior. Cada regla en la cadena da lugar a una coincidencia. Cada regla de la cadena debe contener el indicador, y si la primera regla no encuentra coincidencia, se ignora la cadena completa de reglas.
E=var:value | env=var:value	Puede determinar una variable de entorno utilizando esta directiva. La variable es accesible para las condiciones de reescritura, los Server Side Includes, los scripts CGI, y similares.
F | forbidden	Cuando una regla, que está utilizando este indicador, encuentra coincidencia, envía al navegador una cabecera de respuesta HTTP llamada FORBIDDEN (código de estado 403). Esto deshabilita con eficacia la URL solicitada.
G | gone	Cuando una regla, que está utilizando este indicador, encuentra coincidencia, envía al navegador una cabecera de respuesta HTTP llamada GONE (código de estado 410). Esto le dice al navegador que la URL solicitada no está disponible en este servidor.
L | last	Le dice al motor de reescritura que termine el procesamiento de reglas inmediatamente para que no se apliquen más reglas a la nueva URL tras la sustitución.

Indicador	Significado
N | next	Le dice al motor de búsqueda que vuelva a empezar desde la primera regla. Sin embargo, la primera regla no vuelve a intentar una coincidencia con la URL original, porque ahora opera sobre la URL nueva producto de la sustitución. Esto crea un loop. Debe tener condiciones de término en el loop para evitar un loop infinito.
NC | nocase	Le dice al motor de reescritura que no haga distinción entre mayúsculas y minúsculas en la búsqueda de coincidencias.
NS | nosubreq	Utilice este indicador para evitar aplicar una regla en una solicitud URL generada internamente.
P | proxy	El uso de este indicador convertirá una solicitud URL en una solicitud proxy interna. Esto sólo funcionará si tiene compilado Apache con el módulo `mod_proxy` y configurado para utilizar el módulo del proxy.
QSA | qsappend	Este indicador le permite adjuntar datos (del tipo pares clave = valor) a la parte de la cadena de la consulta de la URL producida tras la sustitución.
R [= HTTP code] | redirect	Fuerza redirecciones externas de los clientes mientras que prefija la sustitución con `http://server[:port]/`. Si no viene dado el código de respuesta HTTP, se utiliza el código de respuesta por defecto 302 (MOVED TEMPORARILY). Esta regla se debería utilizar con L o con el indicador `last`.
S=n | skip=n	Se salta las n reglas siguientes.
T=MIME-type | type=MIME-type	Fuerza a que el MIME-type especificado, sea el MIME-type del archivo objetivo de la solicitud.

NOTA: Puede añadir condiciones a sus reglas precediéndolas con una o varias directivas `RewriteCond`, que se discuten en la sección siguiente.

RewriteCond

La directiva `RewriteCond` es útil cuando queremos añadir una condición extra para una regla de reescritura especificada por la directiva `RewriteRule`. Puede tener varias directivas `RewriteCond` para cada `RewriteRule`. Deben definirse todas las condiciones de reescritura antes de la regla en sí.

> **Sintaxis:** `RewriteCond test_string condition_pattern [flag_list]`
>
> **Predefinido:** ninguno
>
> **Contexto:** configuración del servidor, host virtual, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

La cadena de prueba debe estar construida con texto simple, variables del servidor o referencias, tanto de la regla de reescritura actual como de la última condición de reescritura. Para acceder a la referencia `nth` de la última directiva `RewriteRule`, utilice `$n`; para acceder a la referencia `nth` de la última directiva `RewriteCond`, utilice `%n`.

Para acceder a la variable del servidor, utilice el formato `%{variable name}`. Por ejemplo, para acceder a la variable `REMOTE_USER`, especifique `%{REMOTE_USER}` en la cadena de prueba.

La tabla 9.3 contiene una lista de varios formatos de acceso a datos.

Tabla 9.3. Formatos de acceso a datos para la directiva RewriteCond

Especificador de formato	Significado
%{ENV:variable}	Utilícelo para acceder a cualquier variable de entorno disponible en el proceso Apache.
%{HTTP:header}	Utilícelo para acceder a la cabecera HTTP utilizada en la solicitud.
%{LA-U:variable}	Utilícelo para acceder al valor de la variable que no está disponible en la fase actual del proceso. Por ejemplo, si necesita utilizar la variable del servidor `REMOTE_USER` en una condición de reescritura almacenada en el archivo de configuración del servidor (`httpd.conf`), no puede utilizar `%{REMOTE_USER}` porque esta variable sólo se define una vez que el servidor ha llevado a cabo la fase de autentificación, que tiene lugar después de la fase de procesamiento de URL del módulo `mod_rewrite`. Para saber qué usuario es el que da lugar a una autentificación exitosa del nombre de usuario,

Especificador de formato	Significado
	puede utilizar `%{LA-U:REMOTE_USER}`. Sin embargo, si está accediendo a datos `REMOTE_USER` de un `RewriteCond` en un archivo de configuración de directorios, puede utilizar `%{REMOTE_USER}` porque la fase de autorización ha terminado ya y la variable del servidor está disponible como siempre. La búsqueda tiene lugar generando una subsolicitud basada en URL.
%{LA-F:variable}	Igual que `%{LA-U:variable}` en la mayoría de los casos, pero la búsqueda se realiza utilizando una subsolicitud interna basada en el nombre de archivo.

El patrón de condición puede utilizar también alguna notación determinada además de ser una expresión regular. Por ejemplo, puede realizar comparaciones léxicas entre la cadena de prueba y el patrón prefijando el patrón con un <, un >, o un =. En este caso, el patrón se compara con la cadena de prueba como una cadena de solo texto.

Es posible que, en algunas ocasiones, quiera comprobar si la cadena de prueba es un archivo, un directorio o un enlace simbólico. En este caso, puede reemplazar el patrón con cadenas especiales que puede encontrar en la tabla 9.4.

Tabla 9.4. Opciones condicionales para la cadena de prueba en la directiva RewriteCond

Opciones condicionales	Significado
-d	Comprueba si existe el directorio especificado por la cadena de prueba.
-f	Comprueba si existe el archivo especificado por la cadena de prueba.
-s	Comprueba si existe el archivo de tamaño distinto de cero especificado por la cadena de prueba.
-l	Comprueba si existe el enlace simbólico especificado por la cadena de prueba.
-F	Comprueba si el archivo especificado por la cadena de prueba existe y es accesible.
-U	Comprueba si la URL especificada por la cadena de prueba es válida y accesible.

Puede utilizar ! delante de estas condiciones para negar sus significados. La lista de indicadores opcionales puede consistir en una o más cadenas separadas por comas tal y como se muestra en la tabla 9.5.

Tabla 9.5. Opciones de indicadores para la directiva RewriteCond

Indicador	Significado
NC | nocase	Realiza una prueba de condiciones sin hacer distinción entre mayúsculas y minúsculas.
OR | ornext	Normalmente, cuando tiene más de una `RewriteCond` para una directiva `RewriteRule`, estas condiciones se añaden juntas para que tenga lugar la sustitución final. Sin embargo, si necesita crear una relación OR entre dos condiciones, utilice este indicador.

RewriteMap

La directiva `RewriteMap` facilita una búsqueda clave valor a través de la utilización de un mapa. Piense en este mapa como una tabla de datos en la que cada fila tiene una clave y un valor. Normalmente, un mapa está almacenado en un archivo. Puede ser un archivo de texto, un archivo DBM, una función interna de Apache, o un programa externo. El tipo de mapa se corresponde con la fuente del mapa. La tabla 9.6 contiene una lista de tipos aplicables de mapas.

Sintaxis: `RewriteMap name_of_map type_of_map:source_of_map`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

Tabla 9.6. Opciones de indicadores para la directiva RewriteMap

Tipo de mapa	Descripción
txt	Archivo de texto que tiene líneas de valores de claves de modo que cada par clave valor se encuentra en una sola línea separados por, al menos, un carácter en blanco. El archivo puede contener líneas de comentarios que empiezan por # o puede tener líneas en blanco. Tanto los comentarios como las líneas en blanco son ignorados. Por ejemplo: Key1 value1

Tipo de mapa	Descripción
	Key2 value2 Define dos pares clave valor. Observe que los mapas basados en archivos de texto se leen durante el arranque de Apache y sólo se vuelven a leer si los archivos se han actualizado una vez que el servidor está preparado y ejecutándose. Los archivos también se vuelven a leer durante el reinicio del servidor.
rnd	Un tipo especial de archivo de texto, que tiene todas las restricciones del tipo `txt` pero que permite flexibilidad en la definición del valor. El valor de cada clave se puede definir como un conjunto de valores ORed utilizando el carácter | (barra vertical). Por ejemplo: Key1 first_value_for_key1 | second_value_for_key1 Key2 first_value_for_key2 | second_value_for_key2 Define dos pares de clave valor en las que cada clave tiene varios valores. El valor seleccionado se decide de forma aleatoria.
Int	Las funciones internas de Apache toupper (key) y tolower (key) se pueden utilizar como fuentes de mapas. La primera función convierte todas las letras de la llave en letras mayúsculas y la segunda convierte todas las letras de la clave en letras minúsculas.
dbm	Se puede utilizar un archivo DBM como fuente de mapas. Esto puede ser muy útil y rápido (comparado con los archivos de texto) cuando tiene un gran número de pares clave valor. Tenga en cuenta que estos mapas basados en archivos DBM sólo se leen durante el arranque de Apache y únicamente se vuelven a leer si el archivo se actualiza una vez que el servidor está preparado y ejecutándose. Los archivos también se vuelven a leer durante el reinicio del servidor.
prg	Un programa externo puede generar el valor. Cuando se utiliza un programa externo, se inicia en el arranque de Apache y los datos (clave, valor) se transfieren entre Apache y el programa mediante una entrada estándar (stdin) y una salida estándar (stdout). Asegúrese de que utiliza la directiva `RewriteLock` para definir un archivo de bloqueo cuando utiliza un programa externo, así como de que lee la entrada desde stdin y la escribe en stdout en un modo I/O que no se encuentre en un buffer.

RewriteBase

Esta directiva sólo es útil si está utilizando reglas de escritura en archivos de configuración a nivel de directorio. Sólo es necesario para las rutas de URL que no integran en el directorio físico del archivo objetivo. Asígnele a esta directiva el alias que utilice para el directorio. Esto asegurará que `mod_rewrite` utilizará el alias en lugar de la ruta física en la URL final (sustituida).

Sintaxis: `RewriteBase base_URL`

Predefinido: ruta actual de directorio de la configuración para directorios `(.htaccess)`

Contexto: archivo de control de acceso a directorios `(.htaccess)`

Por ejemplo, cuando se asigna un alias del siguiente modo:

```
Alias /icons/   "/www/nitec/htdocs/icons/"
```

y las reglas de reescritura están disponibles en el archivo `/www/nitec/htdocs/icons/.htaccess`, la directiva `RewriteBase` debe determinarse del siguiente modo:

```
RewriteBase /icons/
```

RewriteLog

Si quiere registrar las aplicaciones de sus reglas de reescritura, utilice esta directiva para asignar un nombre de archivo de registro. Al igual que el resto de las directivas, supone que una ruta sin barra inclinada inicial (/) significa que quiere escribir un archivo de registro en el directorio raíz del servidor.

Sintaxis: `RewriteLog path_to_logfile`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva escribe un archivo de registro en el subdirectorio de registros bajo el directorio raíz de su servidor:

```
RewriteLog logs/rewrite.log
```

Tal y como se mencionó antes, únicamente el usuario del servidor puede escribir en un registro escrito por un servidor.

RewriteLogLevel

Esta directiva le permite especificar qué se ha registrado en el archivo de registros. El valor por defecto 0 significa que no se va a registrar nada. De hecho,

un nivel de registro de 0 significa que no va a producirse ningún proceso relacionado con registros dentro de este módulo. Por lo tanto, si quiere deshabilitar el registro, mantenga este valor en 0.

Sintaxis: `RewriteLogLevel level`

Predefinido: `RewriteLogLevel 0`

Contexto: configuración del servidor, host virtual

NOTA: Si fija la directiva `RewriteLog` en `/dev/null` y `RewriteLogLevel` en un valor que no sea 0, se llevará a cabo el proceso interno relacionado con el registro, pero no se producirá ningún registro. Este proceso gasta gran cantidad de recursos de su sistema, por lo que si no quiere registro, mantenga el valor por defecto en esta directiva. Tiene 10 niveles de registro, que van de 0 a 9. Cuanto más alto sea el nivel, más datos de registro se escriben.

RewriteLock

La directiva `RewriteLock` le permite determinar un programa de mapeado externo para crear mapas de reescritura. Cuando utilice la directiva `RewriteLock` tiene que especificar un nombre de registro. Este archivo se utiliza como archivo de bloqueo para comunicación sincronizada con los programas externos de mapeo.

Sintaxis: `RewriteLock filename`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

Distribución de las URL

Esta sección proporciona ejemplos de URL reescritas que trabajan con la distribución de las URL. A menudo, necesitará redireccionar o amplían una solicitud URL a otra URL. Los siguientes ejemplos le muestran cómo `mod_rewrite` puede ayudar en estos casos.

Ampliar una URL a la forma canónica de las URL

Los sitios Web que ofrecen páginas home para los usuarios, habitualmente soportan un esquema URL del siguiente tipo:

```
http://hostname/~username
```

Esta es una URL de atajo y necesita estar integrada a una URL canónica. Podría tener también otros atajos u otras URL internas que necesitasen estar integradas en sus URL canónicas. Este ejemplo le muestra cómo se traduce `~username` a `/u/username`. La figura 9.2 ilustra lo que tiene que ocurrir.

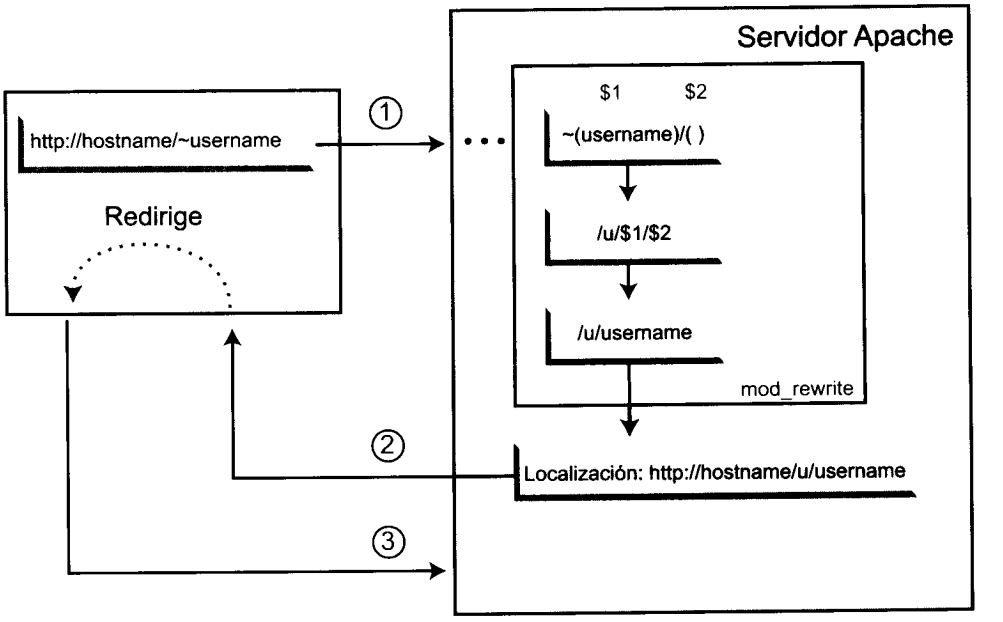


Figura 9.2. Ampliar una URL solicitada a una URL canónica

Cuando se recibe una solicitud para `http://hostname/~username` en (1), la regla de reescritura la traducirá en `/u/username` y redirigirá al navegador a la nueva URL (2). El navegador volverá a solicitar la URL `http://hostname/u/username` en (3) y el procesamiento habitual de solicitudes en Apache completará el proceso de la solicitud.

Es necesario redirigir el HTTP externo porque cualquier solicitud posterior debe utilizar también la URL canónica traducida en lugar de `~username`. La regla tiene que hacer lo siguiente:

```
RewriteRule    ^/~([^/]+)/?(.*) /u/$1/$2   [R,L]
```

NOTA: Observe que el indicador R se utiliza para redirigir, y el indicador L se utiliza para indicar que no se puede aplicar otra regla de reescritura a la URL sustituida.

Muchos sitios ISP con miles de usuarios utilizan una distribución estructurada de directorios home; es decir, cada directorio home es un subdirectorio que empieza, por ejemplo, con el primer carácter del nombre de usuario. Por lo tanto, /

`~foo/anypath` es `/home/f/foo/www/anypath`, mientras que `/~bar/anypath` es `/home/b/bar/www/anypath`. Para implementar un esquema de traducción de las URL a la forma canónica de las URL, en este caso, se puede utilizar la siguiente regla:

```
RewriteRule    ^/~(([a-z])[a-z0?9]+)(.*) /home/$2/$1/www$3 [R,L]
```

Redirigir un directorio home de usuario a un nuevo servidor Web

Si tiene muchas páginas home de usuarios en un servidor Web y necesita moverlas a una máquina nueva por alguna razón, necesita tener una regla de redirección parecida a la que se muestra en la figura 9.3.

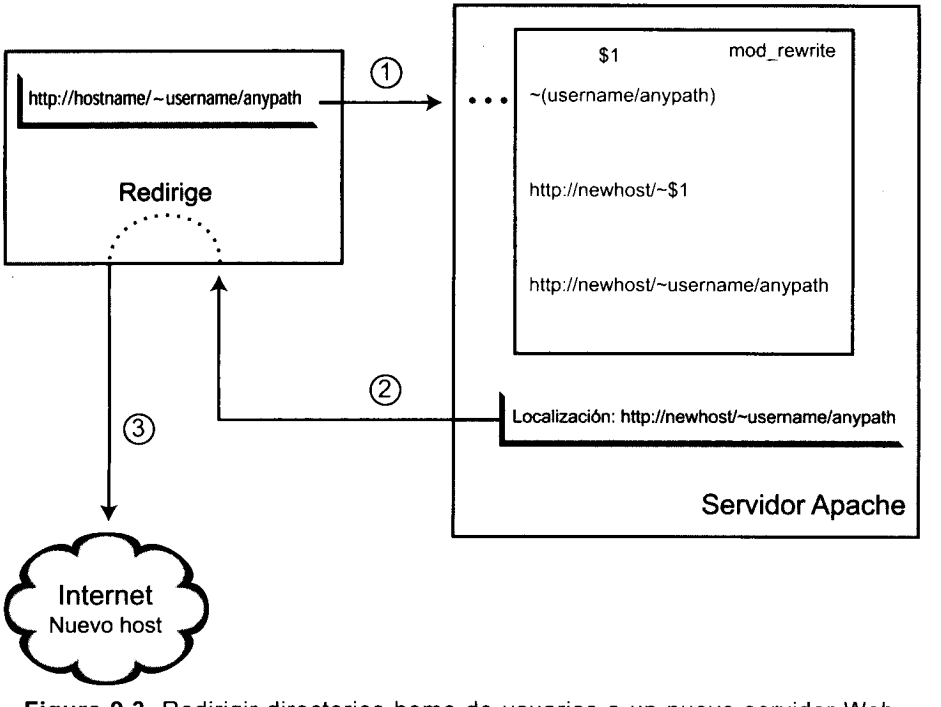


Figura 9.3. Redirigir directorios home de usuarios a un nuevo servidor Web

La solución es muy fácil con `mod_rewrite`. Cuando el navegador solicita `http://hostname/~username/anypath` como se muestra en (1) en la figura, el servidor Web la traduce a `http://newhost/~username/anypath` como se muestra en (2) y redirige el navegador a esta nueva localización. En el servidor Web antiguo (el que redirige la URL) simplemente redirige todas las URL `/~user/anypath` a `http://new~host/~user/anypath`, del siguiente modo:

```
RewriteRule    ^/~(.+)  http://newhost/~$1  [R,L]
```

Buscar una página en varios directorios

En ocasiones es necesario dejar que el servidor Web busque páginas en más de un directorio. Ni `MultiViews` ni otras técnicas de este tipo nos pueden ayudar en esta ocasión. Por ejemplo, imagine que quiere manejar una solicitud para `http://hostname/filename.html` de modo que si `filename.html` no se encuentra en el directorio `dir1` de su servidor Web, el servidor lo intenta en el subdirectorio `dir2`. La figura 9.4 ilustra lo que tiene que ocurrir.

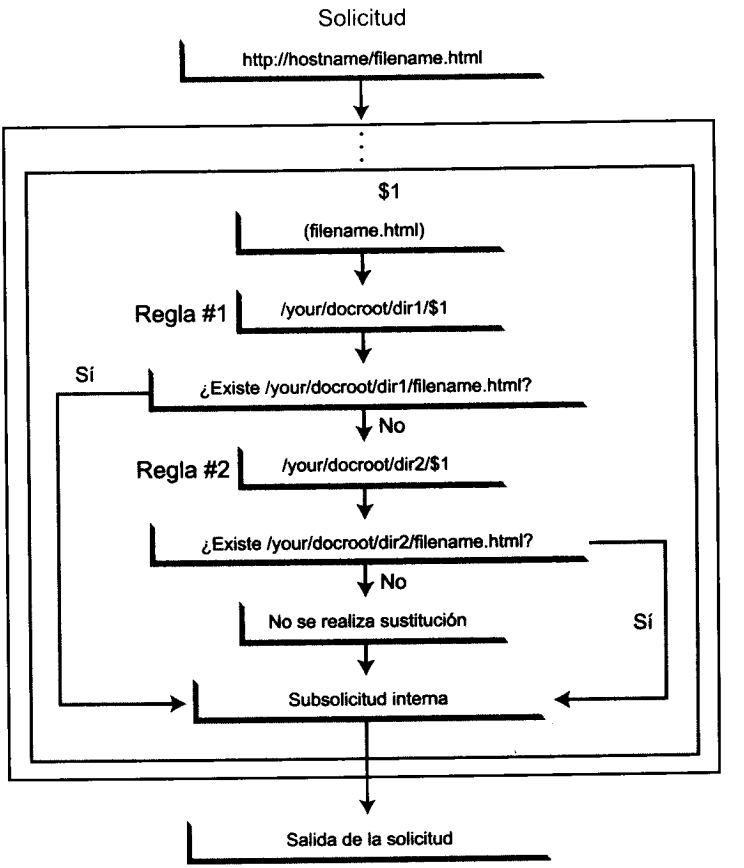


Figura 9.4. Buscar una página en varios directorios

Las reglas que hay que implementar son:

```
RewriteCond          /your/docroot/dir1%{REQUEST_FILENAME}   -f
RewriteRule   ^(.+)  /your/docroot/dir1$1   [L]

RewriteCond          /your/docroot/dir2%{REQUEST_FILENAME}   -f
RewriteRule   ^(.+)  /your/docroot/dir2$1   [L]

RewriteRule   ^(.+)  -   [PT]
```

La primera regla sustituye la URL solicitada por `/your/docroot/dir1/$1` (donde `$1` es el archivo objetivo en la solicitud), sólo si el archivo solicitado existe en el subdirectorio `your/docroot/dir1/`. Si se satisface la condición, esta es la última regla aplicada a esta URL. Sin embargo, si no se encuentra una coincidencia, se aplica la siguiente regla. Esta regla hace lo mismo que la primera pero utiliza el subdirectorio `dir2` para la ruta. Esta regla se convierte en la última en el caso de encontrar una coincidencia. En el caso de que no se satisfaga ninguna de las reglas, la solicitud no se sustituye y pasa al procesamiento normal.

Para comprobar que la regla utiliza el registro, primero, desactive el registro de reescritura añadiendo las siguientes directivas antes de las reglas anteriores.

```
RewriteLog logs/rewrite.log
RewriteLogLevel 5
```

El registro de reescritura `rewrite.log` estará escrito en el subdirectorio que se encuentra bajo el directorio `ServerRoot`. El nivel de registro está fijado en 5 para incluir una cantidad razonable de información. Suponiendo que su directiva `DocumentRoot` se encuentra asignada en `/usr/local/apache/htdocs` y `ServerRoot` en `/usr/local/apache`, necesitará utilizar las siguientes directivas específicas para reglas de escritura en su `httpd.conf`.

```
RewriteLog logs/rewrite.log
RewriteLogLevel 5

RewriteCond          /usr/local/apache/htdocs/
dir1%{REQUEST_FILENAME}   -f
RewriteRule   ^(.+)   /usr/local/apache/htdocs/dir1$1   [L]

RewriteCond          /usr/local/apache/htdocs/
dir2%{REQUEST_FILENAME}   -f
RewriteRule   ^(.+)   /usr/local/apache/htdocs/dir2$1   [L]

RewriteRule   ^(.+)   -   [PT]
```

Una vez que ha reiniciado Apache, haga lo siguiente:

1. Ejecute, como raíz, `tail -f /usr/local/apache/logs/rewrite.log`. Este comando le permite ver entradas de registro a medida que se añaden al archivo `rewrite.log`.
2. Ejecute `mkdir -p /usr/local/apache/htdocs/dir1` y `chmod -R httpd:httpd /usr/local/apache/htdocs/dir1` para crear el directorio `dir1` bajo su documento y para cambiar su dueño a usuario y grupo `httpd`. Estoy suponiendo que ha ejecutado Apache como el usuario `httpd`. Haga lo mismo para `dir2`.
3. Ejecute el comando `lynx -dump -head http://localhost/kabir.html`. Esto lanzará el navegador Web Lynx y le dirá que muestre

únicamente las cabeceras de respuesta devueltas por el servidor Web. Ahora, suponiendo que no tiene un archivo llamado `Kabir.html` en el directorio `/usr/local/apache/dir1`, ni en el `/usr/local/apache/dir2`, ni en el `/usr/local/apache`, debería ver una respuesta parecida a la siguiente:

```
HTTP/1.1 404 Not Found
Date: Fri, 16 Mar 2001 06:06:51 GMT
Server: Apache/2.0.14 (Unix)
Connection: close
Content-Type: text/html; charset=iso-8859-1
```

4. A continuación podemos observar las siguientes entradas `rewrite.log`. He borrado las direcciones IP, el temporizador y otros campos, por cuestiones de brevedad.

```
(2) init rewrite engine with requested uri /kabir.html
(3) applying pattern '^(.+)' to uri '/kabir.html'
(4) RewriteCond: input='/usr/local/apache/htdocs/dir1/
kabir.html' pattern='-f' => not-matched
(3) applying pattern '^(.+)' to uri '/kabir.html'
(4) RewriteCond: input='/usr/local/apache/htdocs/dir2/
kabir.html' pattern='-f' => not-matched
(3) applying pattern '^(.+)' to uri '/kabir.html'
(2) forcing '/kabir.html' to get passed through to next API
URI-to-filename handler
```

NOTA: Observe cómo `mod_rewrite` intentó localizar el archivo `kabir.html` en `dir1` y en `dir2` y cómo después se rindió, lo que dio lugar a una búsqueda en Apache del mismo archivo en la raíz de documentos porque la solicitud fue `http://localhost/kabir.html`.

5. A continuación ha de crear el archivo de prueba, `kabir.html`, en el subdirectorio `dir1` o en el subdirectorio `dir2`, y cambiar el dueño del archivo de modo que Apache (usuario httpd) pueda leerlo. Entonces ejecute el mismo comando `lynx -dump -head http://localhost/kabir.html` de nuevo, y mire el contenido de `rewrite.log`. Verá que una de las reglas ha tenido éxito basándose en el lugar (`dir1` o `dir2`) en el que colocó el archivo.

6. Si tiene el archivo de prueba en los tres directorios, `/usr/local/apache/htdocs/dir1`, `/usr/local/apache/htdocs/dir2` y `/usr/local/apache/htdocs`, la regla de reescritura elige el archivo del subdirectorio `dir1` porque gana la primera condición de coincidencia como consecuencia del indicador `[L]` (last).

Asignar una variable de entorno basándose en una URL

Es posible que quiera mantener información de estado entre solicitudes y utilizar la URL para codificarla. Pero puede evitar utilizar un script CGI para todas las páginas con el fin de sacar esta información. Puede utilizar una regla de reescritura para obtener la información de estado y almacenarla mediante una variable de entorno que más tarde se puede desreferenciar desde XSSI o desde CGI. De este modo, se traduce una URL `/foo/S=java/bar/` a `/foo/bar/` y a la variable de entorno llamada `STATUS` se le asigna el valor `java`. La figura 9.5 ilustra lo que está ocurriendo.

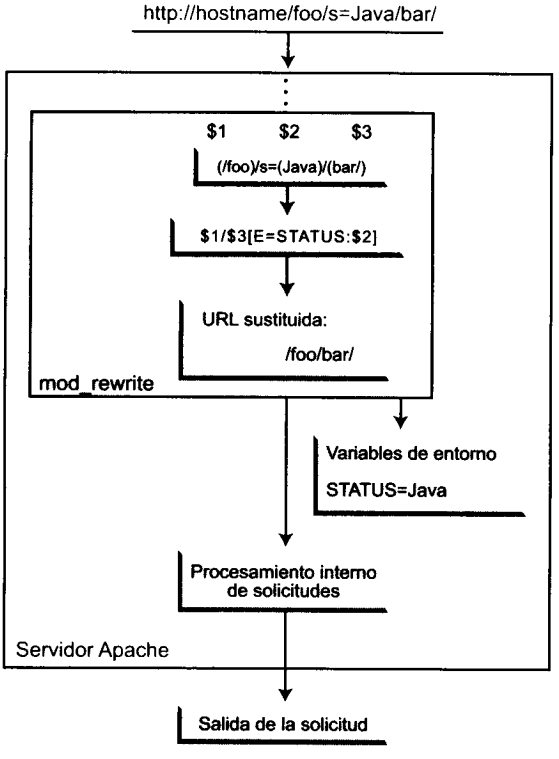


Figura 9.5. Asignar una variable de estado desde una URL

Cuando se detecta una solicitud para `http://hostname/foo/s=java/bar/` el servidor Apache asigna `$1` a `foo`, `$2` a `java`, y `$3` a `bar`, tal y como se muestra en la figura 9.5. El valor para `$2` se utiliza para asignar la variable de entorno `STATUS` y el servidor Web lleva a cabo un redireccionamiento interno a la localización `/foo/bar`. Los pasos de reescritura de URL mostrados en la figura 9.5 se pueden implementar utilizando esta regla de reescritura:

```
RewriteRule    ^(.*)/S=([^/]+)/(.*) $1/$3 [E=STATUS:$2]
```

El valor de $2 se almacena en la variable de entorno llamada STATUS utilizando el indicador E. Cuando esta regla está en su sitio y se realiza una solicitud del tipo `lynx -dump -head http://localhost/dir1/S=value/kabir.html`, el archivo de reescritura (si está disponible) mostrará:

```
(2) init rewrite engine with requested uri /dir1/S=value/
kabir.html
(3) applying pattern '^(.*)/S=([^/]+)/(.*)' to uri '/dir1/
S=value/kabir.html'
(2) rewrite /dir1/S=value/kabir.html -> /dir1/kabir.html
(5) setting env variable 'STATUS' to 'value'
(2) local path result: /dir1/kabir.html
(2) prefixed with document_root to /usr/local/apache/htdocs/
dir1/kabir.html
(1) go-ahead with /usr/local/apache/htdocs/dir1/kabir.html [OK]
```

He acortado la salida de registro de reescritura por motivos de brevedad. Observe que mod_rewrite ha fijado la variable STATUS en 'value', por lo que ahora si un script CGI basado en Perl quiere acceder al valor de la variable de entorno STATUS, puede utilizar $ENV{STATUS}. De igual modo, una directiva Server-Side Include (SSI) también puede acceder a esta variable de entorno.

Crear sitios www.username.domain.com

Vamos a imaginar que tiene unos cuantos amigos que quieren sitios Web en su servidor. En lugar de darles el sitio del tipo http://www.domain.com/~username (http://wwww.dominio.com/nombredeusuario), puede crear sitios del tipo http://www.username.domain.com (http://www.nombredeusuario.com) para cada uno de ellos. Naturalmente, tiene que añadir cada nombre de host basado en el nombre de usuario (por ejemplo, www.kabir.domain.com) en su DNS, utilizando un registro CNAME que está dirigido a su servidor Web. Por ejemplo, en su DNS podría tener:

```
www.domain.com.              IN      A       192.168.1.100.
```

Para crear un sitio Web para dos amigos llamados Joe y Jennifer, necesita los siguientes registros DNS para domain.com:

```
www.domain.com.              IN      A       192.168.1.100.
www.joe.domain.com.          IN      CNAME   www.domain.com.
www.jennifer.domain.com.     IN      CNAME   www.domain.com.
```

Una vez que el DNS está preparado y probado, tendrá que configurar Apache para servir estos sitios utilizando directorios /home/username/www, en los que cada username es el nombre de la cuenta de usuario de su amigo.

La siguiente asignación de la regla de reescritura se puede utilizar para reescribir internamente http://www.username.domain.com/anypath a /home/username/www/anypath:

```
RewriteCond    %{HTTP_HOST}     ^www\.[^.]+\.domain\.com$
RewriteRule    ^(.+)          %{HTTP_HOST}$1                [C]
RewriteRule    ^www\.([^.]+)\.domain\.com(.*)  /home/$1/www/$2
```

La figura 9.6 ilustra este trabajo.

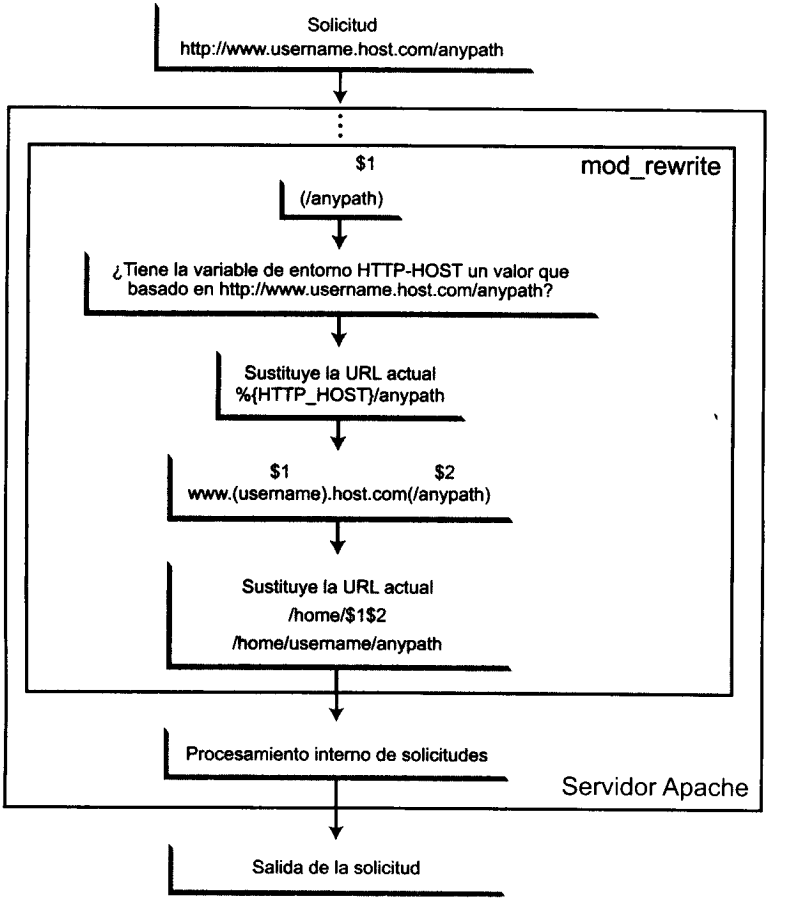


Figura 9.6. Host virtual para cada username

Este es un ejemplo de un conjunto de reglas encadenadas. La primera regla tiene una condición que comprueba si la variable de entorno `HTTP_HOST` tiene un patrón del tipo `www.username.domain.com`. En caso de tenerlo, se aplica la regla. En otras palabras, `www.username.domain.com/anypath` se sustituye por una solicitud del tipo `http://www.username.domain.com/anypath`. Esto puede resultar algo confuso porque la sustitución no es demasiado obvia. Esta sustitución es necesaria por lo que se puede extraer el nombre de usuario (username) utilizando la segunda regla. Ésta extrae la parte del nombre de usuario de la solicitud sustituida y crea una nueva URL `/home/username/www/anypath` para una subsolicitud interna.

Redireccionar una URL fallida a otro servidor Web

Si tiene una red de varios servidores Web y mueve contenido de uno a otro con cierta frecuencia, se podría enfrentar a la necesidad de redirigir solicitudes URL fallidas de un servidor Web A a un servidor Web B. Existen varias formas de hacerlo: puede utilizar la directiva `ErrorDocument`, escribir un script CGI o utilizar `mod_rewrite` para rescribir las URL fallidas en el otro servidor. Utilizar la solución basada en `mod_rewrite` es menos adecuado que utilizar una directiva `ErrorDocument` o un script CGI. La solución `mod_rewrite` tiene el mejor rendimiento, pero es menos flexible y segura en cuanto a errores:

```
RewriteCond  /your/docroot/%{REQUEST_FILENAME}  !-f
RewriteRule  ^(.+)  http://Web serverB.dom/$1
```

El problema es que esta solución únicamente funcionará para páginas en el interior de la directiva `DocumentRoot`. Aunque pueda añadir más condiciones (para manejar directorios home, por poner un ejemplo), hay una variante mejor:

```
RewriteCond    %{REQUEST_URI}  !-U
RewriteRule    ^(.+)           http://Web serverB.dom/$1
```

Esta variante utiliza la característica URL de proyecto de `mod_rewrite`, y funcionará en todos los tipos de URL. Esto va a tener un impacto de rendimiento en el servidor Web porque para cada solicitud realizada, hay una subsolicitud interna. Utilice esta opción si su servidor Web se ejecuta en una CPU potente; si se trata de una máquina lenta, utilice la primera aproximación, o, aún mejor, una directiva `ErrorDocument` o un script CGI.

Crear un acceso multiplexor

Este ejemplo le muestra cómo crear un conjunto de reglas para redirigir solicitudes basadas en un tipo de dominio, como `.com`, `.net`, `.edu`, `.org`, `.uk`, `.de`, y similares. La idea es redirigir al visitante al sitio Web geográficamente más cercano. Las grandes corporaciones emplean esta técnica para redireccionar a los clientes internacionales al sitio Web o al servidor FTP adecuado.

El primer paso para crear una solución de este tipo es crear un archivo map. El siguiente ejemplo muestra un archivo map basado en texto llamado `site-redirect.map`:

```
com        http://www.mydomain.com/download/
net        http://www.mydomain.com/download/
edu        http://www.mydomain.com/download/
org        http://www.mydomain.com/download/
uk         http://www.mydomain.uk/download/
```

```
de          http://www.mydomain.de/download/
ch          http://www.mydomain.ch/download/
```

Cuando se recibe una solicitud para `http://www.mydomain.com/downlod/anypath` desde un host llamado `dialup001.demon.uk`, la solicitud necesita ser redirigida al sitio Web `www.mydomain.uk/download/`; del mismo modo, cualquier solicitud desde un host que pertenezca a los dominios de nivel máximo `.com`, `.net`, `.edu` y `.org` son enrutados al sitio `www.mycompany.com/download/`.

Aquí tenemos las reglas que se necesitan para el sistema anterior:

```
RewriteMap    sitemap            txt:/path/to/site-redirect.map
RewriteRule   ^/download/(.*)    %{REMOTE_HOST}::$1   [C]
RewriteRule   ^.+\.([a-zA-Z]+)::(.*)$
%{sitemap:$1|www.mydomain.com/download/}$2  [R,L]
```

La figura 9.7 ilustra la utilización de esta regla.

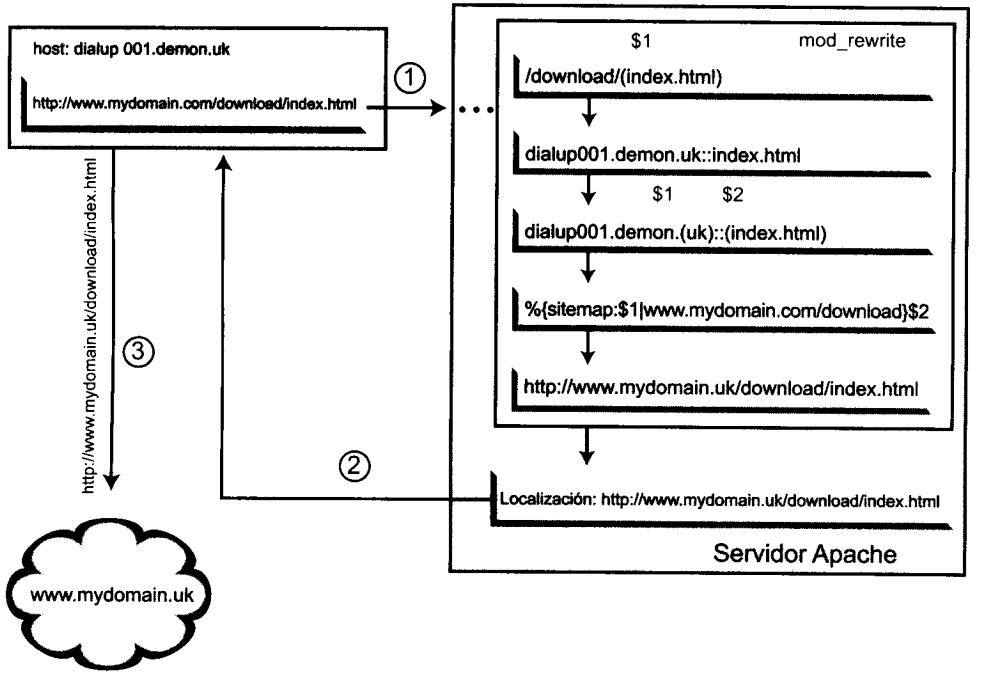


Figura 9.7. Un acceso multiplexor basado en URL

Tal y como muestra la figura 9.7, cuando un host como `dialup001.demon.uk` solicita la página `www.mydomain.com/download/index.html` (1), la primera regla reescribe la solicitud utilizando el nombre del host solicitado:

```
dialup001.demon.uk::index.html
```

Entonces se aplica la siguiente regla de la cadena. Esta regla se aplica cuando el patrón de búsqueda coincide y se crea la URL sustituida colocando el archivo map en el dominio de máximo nivel. Si no se encuentran coincidencias, se utiliza `www.mydomain.com` que es la URL por defecto. Esto lo lleva a cabo el operador | (or) en la cadena URL de sustitución. Quizá es más fácil de entender la segunda regla utilizando el algoritmo tal y como se muestra en el listado 9.1.

Listado 9.1. Algoritmo para la segunda regla de reescritura

```
if(current URL matches a fully-qualified-hostname::anything)
then

# substituye la URL actual utilizando la información del tipo de
# dominio, almacenada en $1, realizando una búsqueda en el
archivo map.

If  (map file has a key that matches the domain type) then
#utiliza el valor de la clave del siguiente modo:

        Substituted URL = value-of-the-key$2
        #donde $2 es cualquier cosa que se encuentre después de
        #fully-qualified-hostname::  pattern

Else
        # Utiliza el valor por defecto www.mydomain.com/
download/$2

        Substituted URL = www.mydomain.com/download/$2
endif

Endif
```

El indicador `R` lo convierte en una redirección URL externa y el indicador `L` lo convierte en la última regla para la URL sustituida. La nueva localización creada por la regla #2 se envía al navegador Web (2) y el navegador obtiene la página en (3) tal y como se muestra en la figura 9.7.

Crear URL dependientes del tiempo

Es posible que alguna vez se haya preguntado si se puede tener una URL que pudiese dirigirse a distintos archivos dependiendo del momento. `mod_rewrite` crea de un modo sencillo este tipo de URL. Hay muchas variables llamadas `TIME_xxx` para reescribir condiciones. Utilizando los patrones de comparación especiales `<STRING`, `>STRING` y `=STRING` puede realizar redireccionamientos dependientes del tiempo, por ejemplo:

```
RewriteCond    %{TIME_HOUR}%{TIME_MIN} >0700
RewriteCond    %{TIME_HOUR}%{TIME_MIN} <1900
```

```
RewriteRule    ^foo\.html$              foo.day.html
RewriteRule    ^foo\.html$              foo.night.html
```

Esto proporciona el contenido de `foo.day.html` bajo la URL `foo.html` desde 07:00 a 19:00, y el resto del tiempo, proporciona el contenido de `foo.night.html`.

Manejar contenido

Los ejemplos de esta sección versan sobre reglas de reescritura de contenido específico. Le mostraré cómo crear las URL de compatibilidad retroactiva, a reescribir contenido basado en el navegador, a crear HTML transparente al usuario final para las redirecciones CGI, y este tipo de cosas.

Añadir compatibilidad retroactiva en las URL

Imagine que ha renombrado recientemente la página `bar.html` a `foo.html` y que ahora quiere proporcionar la antigua URL para tener compatibilidad retroactiva. Además, no quiere que los usuarios de la antigua URL se den cuenta de que la página ha sido renombrada. ¿Cómo puede hacer esto? A continuación le mostramos la manera:

```
RewriteRule    ^foo\.html$  bar.html
```

Si quiere dejar que el navegador se dé cuenta del cambio, puede realizar una reescritura externa de modo que el navegador mostrará la nueva URL. Todo lo que necesita hacer es añadir el indicador `R` del siguiente modo:

```
RewriteRule    ^foo\.html$  bar.html [R]
```

Crear las URL con contenido específico para el navegador

Puede utilizar reglas de reescritura para servir distinto contenido (utilizando subsolicitudes internas) a distintos navegadores. No puede utilizar negociación de contenido para esto, porque los navegadores no proporcionan sus tipos de este modo. En lugar de esto, tiene que actuar en la cabecera HTTP User-Agent. Por ejemplo, si la cabecera User-Agent de un navegador coincide con Mozilla/5 entonces puede enviar una página de características afines a Netscape Navigator 5 (o superior), o puede enviar una página distinta si el navegador es una versión más antigua de Navigator u otro tipo de navegador.

Si la cabecera HTTP User-Agent empieza por Mozilla/5, la página `foo.html` es reescrita a `foo.NS.html` y termina la reescritura. Si el navegador es Lynx o

la versión de 1 a 4 de Mozilla, la URL se convierte en `foo.dull.html`. El resto de los navegadores reciben la página `foo.cool.html`. Esto se lleva a cabo con las reglas siguientes:

```
RewriteCond %{HTTP_USER_AGENT}   ^Mozilla/5.*
RewriteRule ^foo\.html$          foo.cool.html   [L]

RewriteCond %{HTTP_USER_AGENT}   ^Lynx/.*   [OR]
RewriteCond %{HTTP_USER_AGENT}   ^Mozilla/[1234].*
RewriteRule ^foo\.html$          foo.dull.html   [L]
```

Cuando se recibe una solicitud de una URL del tipo `http://hostname/foo.html`, la primera condición comprueba si la variable de entorno `HTTP_USER_AGENT` tiene un valor que contiene la cadena Mozilla/5. o no lo tiene. Si contiene la cadena, entonces se aplica la primera regla. Esta regla sustituye `foo.cool.html` por la URL original y se completa la reescritura; sin embargo, cuando no se aplica la primera regla, se invoca la segunda. Hay dos condiciones OR. En otras palabras, una de estas condiciones deben coincidir antes de que esta regla se aplique.

La primera condición prueba la misma variable de entorno para la subcadena `Lynx/`, y la segunda condición prueba la misma variable de entorno para la subcadena Mozilla/1 a través de Mozilla/4. Si se cumple alguna de estas condiciones, se aplica la regla. La regla sustituye `foo.dull.html`, la URL original. La URL sustituida se convierte en una subsolicitud y Apache la procesa de la forma habitual.

Crear HTML para un puente CGI

Si quiere transformar la página estática `foo.html` en una variante dinámica llamada `foo.cgi`, sin informar ni al navegador ni al usuario, a continuación le mostramos cómo hacerlo:

```
RewriteRule  ^foo\.html$   foo.cgi   [T=application/x-httpd-cgi]
```

La regla rescribe una solicitud para `foo.html` a una solicitud para `foo.cgi`. Además fuerza el tipo correcto de MIME, de modo que se ejecuta como un script CGI. Una solicitud del tipo `http://hostname/foo.html` se traduce internamente en una solicitud para el script CGI. El navegador no sabe que esta solicitud se ha redirigido.

Restricción de acceso

Estos ejemplos muestran temas sobre el control de acceso. En esta sección le muestro cómo controlar el acceso a ciertas áreas de su sitio Web utilizando el módulo para rescribir URL.

Robots de bloqueo

Es sencillo bloquear un programa rastreador de Web (también llamados robots) molesto, para que no pueda obtener páginas de un sitio Web determinado. Puede probar con un archivo `/robots.txt` que contenga entradas de Robot Exclusion Protocol, pero, normalmente, esto no es suficiente para deshacerse de estos tipos de robots. Una solución podría ser:

```
RewriteCond %{HTTP_USER_AGENT}    ^NameOfBadRobot.*
RewriteCond %{REMOTE_ADDR}        ^123\.45\.67\.[8?9]$
RewriteRule ^/not/to/be/indexed/by/robots/.+    -    [F]
```

Esta regla tiene dos condiciones:

```
If  (HTTP_USER_AGENT of the robot matches a pattern
"NameOfBadRobot" ) and
   (REMOTE_ADDR of the requesting host is 123.45.67.8 to
123.45.67.9) then
   No substitution but send a HTTP "Forbidden" header (status
code 403)
endif
```

Como puede ver, coincide la cabecera User-Agent del robot, junto con la dirección IP del host que se utiliza. Las condiciones anteriores permiten que se comprueben varias direcciones IP (`123.45.67.8` y `123.45.67.9`).

Crear deflector URL basado en una referencia HTTP

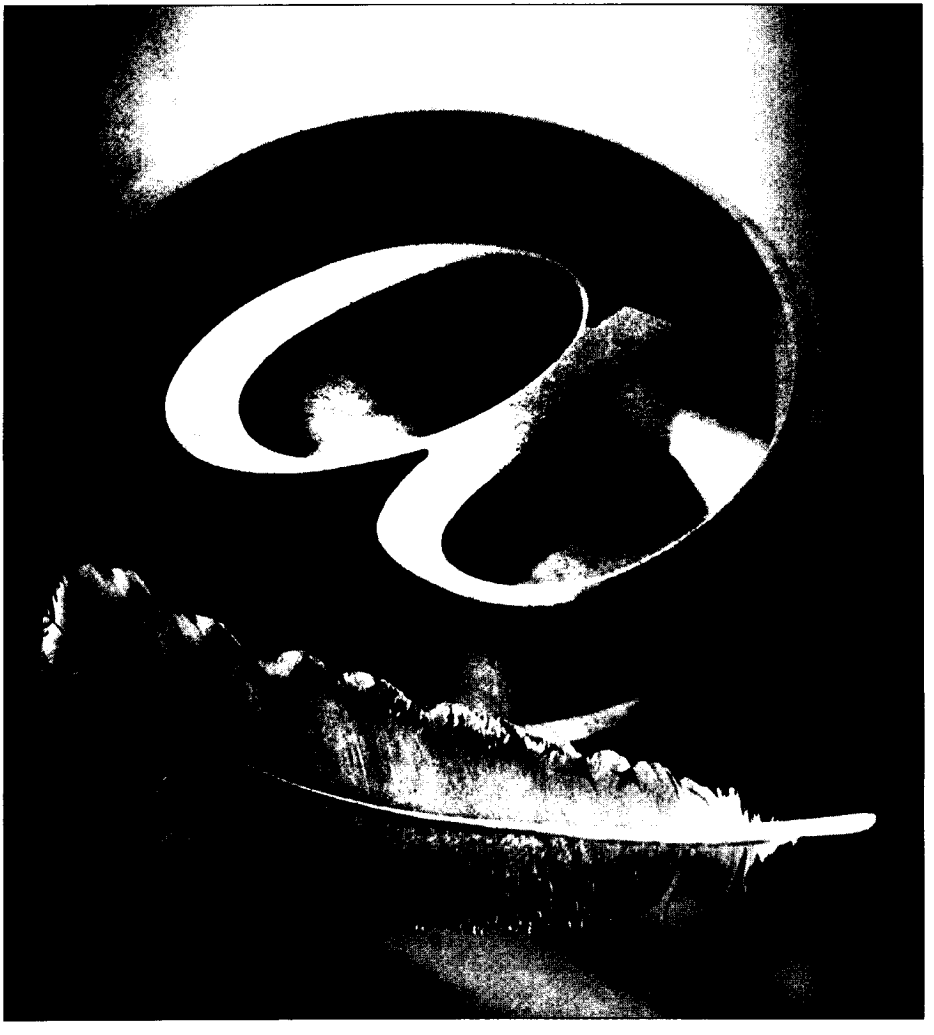
Puede programar un deflector URL flexible que actúe en la cabecera Referer HTTP y que la configure con tantas páginas como quiera. A continuación tenemos el modo de hacerlo:

```
RewriteMap   deflector txt:/path/to/deflector.map
RewriteRule ^/(.*)
${deflector:%{HTTP_REFERER}|/$1}
RewriteRule ^/DEFLECTED    %{HTTP_REFERER} [R,L]
RewriteRule .*  - [PT]
```

Esto se utiliza junto con el mapa correspondiente de reescritura:

```
http://www.badguys.com/bad/index.html     DEFLECTED
http://www.badguys.com/bad/index2.html    DEFLECTED
http://www.badguys.com/bad/index3.html    http://somewhere.com/
```

Esto redirecciona automáticamente las solicitudes a la página de referencia si la URL coincide con el valor DEFLECTED en el archivo map. En todos los casos, las solicitudes se redireccionan a las URL especificadas.



10 Establecer un servidor Proxy

En este capítulo

1. Analizamos los tipos de servidores proxy.
2. Configuramos Apache como un servidor proxy.
3. Preparamos navegadores Web para proxies.

Un servidor proxy es un sistema que se sitúa entre el host del cliente y el servidor al que quiere acceder. Cuando un host solicita un cierto recurso remoto utilizando una URL, el servidor proxy recibe esta solicitud y utiliza el recurso para satisfacer la solicitud del cliente. En términos generales, un servidor proxy actúa como un servidor para los host del cliente y como un cliente para los servidores remotos.

En un escenario típico de proxy, este proceso permite al servidor proxy almacenar el contenido solicitado en un caché. Cualquier solicitud nueva que requiera información que se encuentra en el caché no necesita ser servida trayendo la información desde el servidor remoto. En lugar de eso, la nueva solicitud se sirve desde los datos del caché. Esto permite a los servidores proxy suavizar los cuellos de botella. Sin embargo, esto no es todo lo que un servidor proxy puede hacer. Este capítulo le enseña a convertir Apache en un servidor proxy que puede llevar

a cabo multitud de servicios. Aprenderá a convertir Apache en un servidor proxy de caching (forward). Distribuir este tipo de servidores en un cuello de botella puede reducir retrasos en los tiempos de respuesta, conservar el ancho de banda y, además, le ayuda a reducir el gasto general de sus comunicaciones. Como el proxy se utiliza, normalmente, para redes con grandes comunidades de usuarios, también vamos a tratar los distintos aspectos de la configuración automática del proxy.

¿Quién debería utilizar un servidor proxy?

El propósito de este servidor proxy es traer los recursos solicitados desde el servidor remoto, devolverlos al usuario que los solicita y cachearlos en los drivers locales. El servicio proxy es perfecto para escenarios en los que están accediendo a la red varios usuarios. Muchas organizaciones tienen varios ordenadores host que acceden a Internet mediante una sola conexión Internet, como un router RDSI o cualquier otra conexión dedicada o en demanda. Un proxy puede ser muy útil en tales redes. Utilizando un proxy tanto para Internet como para su intranet puede obtener los siguientes beneficios:

- **Proxying:** si la red interna utiliza direcciones IP no rutables por razones de seguridad o económicas, puede utilizar un servidor proxy para proporcionar recursos de Internet a los host que normalmente no pueden acceder a Internet. Este capítulo le enseña cómo hacerlo.
- **Caching:** utilizando un proxy caching como Apache (con `mod_perl`), puede proporcionar a los usuarios locales de los recursos de Internet un acceso más rápido. Esto no sólo aumentará el rendimiento de la red según la percepción del usuario sino que, además, reducirá los costes de utilización de ancho de banda.
- **Control de registro y acceso:** utilizando un servidor proxy, puede controlar el uso que hacen los empleados o los estudiantes de Internet (o incluso de intranet). Puede bloquear el acceso a determinados sitios Web para proteger a su compañía, y puede evitar que abusen del tiempo de su compañía. Analizando los accesos a su servidor proxy y los registros de error, puede identificar los patrones de uso y realizar una política de utilización de la red más adecuada en el futuro.

Análisis de los tipos de servidores proxy

Antes de hablar de la utilización de Apache como servidor proxy, vamos a discutir los tipos de servidores proxy y su funcionamiento. Hay dos tipos de servidores proxy:

- **Servidores proxy forward:** cuando se utiliza este tipo de servidores proxy, los usuarios pasan sus solicitudes al servidor proxy y éste obtiene la respuesta del host objetivo de la solicitud. Los proxies forward están normalmente definidos explícitamente en los programas usuarios (como en el navegador Web).
- **Servidores proxy reverse:** cuando se utiliza este tipo de servidor, los usuarios no son conscientes de su existencia porque piensan que están accediendo al recurso directamente. Todas las solicitudes que realiza el usuario se envían al proxy reverse, que sirve la respuesta desde su caché o recogiendo información de otro host.

Proxy forward

Un proxy forward normalmente se sitúa entre el host del usuario y los recursos remotos a los que quiere acceder. Un recurso puede ser un recurso de Internet, como muestra la figura 10.1, o puede ser un recurso de intranet. La siguiente solicitud para el mismo recurso será servida desde los datos del caché si los datos no han expirado.

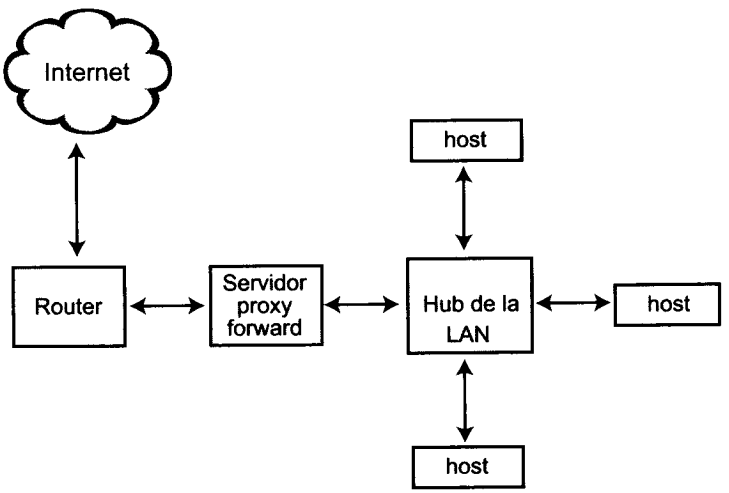


Figura 10.1. Un servidor proxy forward

Los hosts usuarios saben que están utilizando un servidor proxy porque cada host debe ser configurado para utilizar un servidor proxy. Así por ejemplo, es necesario que le diga al navegador Web que utilice un servidor proxy antes de que el navegador pueda utilizarlo. Todas las solicitudes remotas están encauzadas mediante el servidor forward mostrado en la figura 10.1, proporcionando, de este modo, una solución manejable y de coste efectivo para reducir la utilización de ancho de banda e implementar la política de acceso a usuarios. Este tipo de servidores proxy también se conocen con el nombre de *servidor proxy caching*. El

proxy reverse también mete datos en el caché pero funciona de forma contraria a como lo hace el servidor proxy forward.

Proxy reverse

Un proxy reverse se sitúa frente a un recurso de Internet, tal y como muestra la figura 10.2, o frente a un recurso de intranet. En estos sistemas, el proxy reverse recupera los recursos solicitados del servidor original y los devuelve al host usuario.

El host usuario conectado con el servidor proxy no es consciente de estar conectado al servidor proxy, sino que piensa que está conectado directamente al servidor que contiene los recursos, a diferencia de lo que ocurre cuando utilizamos un servidor forward. Desde el punto de vista del usuario, está accediendo al recurso solicitado directamente.

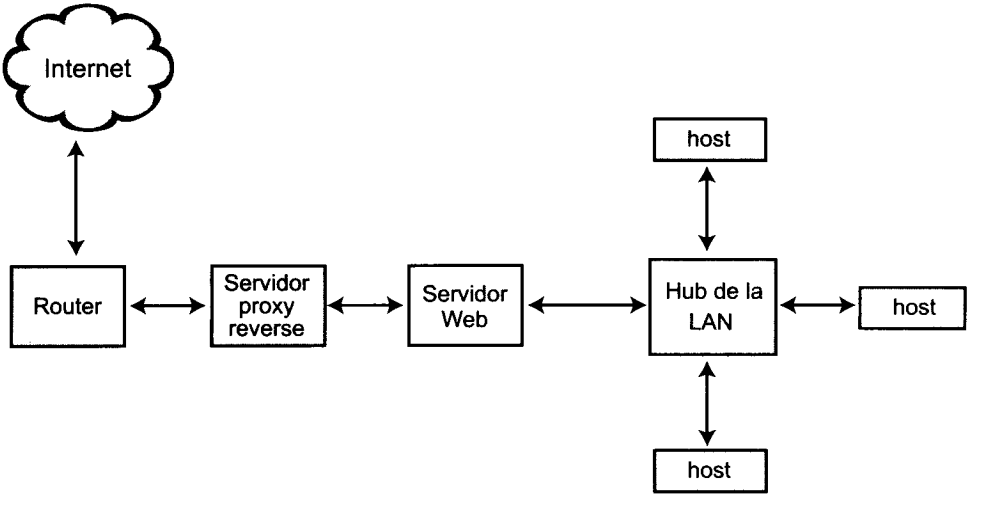


Figura 10.2. Un servidor proxy reverse

La figura 10.2 muestra que los usuarios de Internet tienen que atravesar el proxy reverse para conectarse al servidor Web. Como es un proxy reverse, los usuarios no son conscientes de su existencia y piensan que están conectados con el servidor Web. Los usuarios de una LAN pueden conectar directamente con el servidor Web desde la red interna, tal y como se muestra en la figura.

Por ejemplo, si se utiliza un servidor proxy reverse para un sitio Web llamado `www.csus.edu`, entonces todos los estudiantes de CSUS (California State University, Sacramento) tienen que dirigir su navegador a `www.csus.edu` y no hacer referencia a ninguna configuración proxy en su navegador. El navegador manda una solicitud al servidor `www.csus.edu`. Lo cierto es que al navegador no le sirve de mucho saber que el servidor `www.csus.edu` es en realidad un servidor proxy reverse que traduce internamente las solicitudes a un servidor Web

llamado `internal-www.csus.edu`, para obtener el contenido de la solicitud. ¿Qué ventaja tiene este tipo de sistemas? Como el dato está cacheado para cada solicitud, el servidor proxy puede proporcionar algún soporte de balance de carga para los verdaderos servidores que se encuentran detrás de la escena.

NOTA: Apache no soporta por ahora el servicio proxy reverse; sin embargo, se implementará en la siguiente versión del módulo mod_proxy.

Directivas mod_proxy

El soporte proxy en Apache proviene del módulo `mod_proxy`. Este módulo no está compilado por defecto. Tiene que reconfigurar Apache utilizando `./config.status --enable-module=proxy` del directorio de la distribución de la fuente de Apache y ejecutar `make && make install` para recompilar y reinstalar el servidor Apache actualizado. Actualmente, sólo está implementado un servidor proxy caching en Apache. Es capaz de soportar los protocolos HTTP 1.1, HTTPS (vía CONNECT para SSL) y FTP. El módulo también se puede configurar para conectarse a otros módulos proxy para estos y otros protocolos. Proporciona las directivas que se discuten en las siguientes secciones.

ProxyRequests

`ProxyRequests` le permite activar o desactivar el servicio caching del proxy. Sin embargo, no afecta a la funcionalidad de la directiva `ProxyPass`.

Sintaxis: `ProxyRequests On | Off`

Predefinido: `ProxyRequests Off`

Contexto: configuración del servidor, host virtual

ProxyRemote

`ProxyRemote` le permite a su servidor funcionar de interfaz con otro servidor proxy.

Sintaxis: `ProxyRemote match remote_proxy_server_URL`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

El valor de correspondencia puede ser alguno de los siguientes:

- El nombre de un esquema URL que soporta el servidor remoto.
- Una URL parcial para la que se pueda utilizar el servidor remoto.
- Para indicar el servidor que debería ser contactado para todas las solicitudes.

`remote_proxy_server_URL` puede tener un valor del tipo `http://hostname-:port`. Observe que, por ahora, únicamente se soporta el protocolo HTTP. En otras palabras, puede especificar únicamente un servidor proxy que trate con el protocolo HTTP; sin embargo, puede dirigir las solicitudes FTP desde su servidor proxy a uno que soporte los protocolos HTTP y FTP del siguiente modo:

```
ProxyRemote ftp http://ftp.proxy.evoknow.com:8000
```

envía todas las solicitudes FTP que proceden del servidor proxy local a `ftp://ftp.proxy.evoknow.com`. Las solicitudes se envían vía HTTP, por lo tanto, la verdadera transacción FTP tiene lugar en el servidor proxy remoto.

Si quiere dirigir todas las solicitudes proxy de un determinado sitio Web a este servidor proxy directamente, puede hacerlo con esta directiva. Por ejemplo:

```
ProxyRemote http://www.bigisp.com/ http://web-proxy.bigisp.
com:8000
```

Esto envía todas las solicitudes que coinciden con `www.bigisp.com` a `web-proxy.bigisp.com`. Si quiere dirigir todas sus solicitudes proxy a otro proxy, puede utilizar el asterisco como correspondencia. Por ejemplo:

```
ProxyRemote * http://proxy.domain.com
```

envía todas las solicitudes proxy locales al servidor proxy `proxy.domain.com`.

ProxyPass

`ProxyPass` le permite mapear un árbol de documentos del servidor Web en el espacio de documentos de su servidor proxy.

Sintaxis: `ProxyPass relative_URL destination_URL`

Contexto: configuración del servidor, host virtual

Por ejemplo:

```
ProxyPass /internet/microsoft        www.microsoft.com/
```

Si `ProxyPass` se encuentra en el archivo `httpd.conf` de un servidor proxy llamado `proxy.evoknow`, permitirá a los usuarios del servidor proxy que accedan al sitio Web de Microsoft utilizando la URL:

```
http://proxy.evoknow.com/internet/microsoft
```

actúa como un mirror del sitio Web remoto. Cualquier solicitud que utiliza el `<relative-URL>` se convertirá internamente en una solicitud proxy para él `<destination-URL>`.

Si el sitio remoto incluye referencias absolutas, las imágenes no aparecerán y los enlaces no funcionarán. Tampoco podrá utilizar esta directiva con los servidores SSL.

ProxyBlock

La directiva ProxyBlock le permite bloquear el acceso al host o al dominio.

Sintaxis: `ProxyBlock partial_or_full_hostname [. . .]`

Contexto: configuración del servidor, host virtual

Por ejemplo:

```
ProxyBlock gates
```

bloquea el acceso a cualquier host que tenga la palabra `gates` en su nombre. De este modo, el acceso a `http://gates.ms.com` o a `http://gates.friendsofbill.com` está bloqueado. También puede especificar varios hosts. Por ejemplo:

```
ProxyBlock apple orange.com bannana.com
```

bloquea todos los accesos a cualquier host que coincida con cualquiera de las palabras o nombres de dominio anterior. El módulo `mod_proxy` intenta determinar la dirección IP para estos hosts durante el arranque del servidor, y los cachea para correspondencias posteriores.

Para bloquear el acceso a todos los hosts, utilice:

```
ProxyBlock *
```

Esto efectivamente inutiliza su servidor proxy.

NoProxy

`NoProxy` le proporciona cierto control sobre la directiva `ProxyRemote` en un entorno de intranet.

Sintaxis: `NoProxy Domain_name| Subnet | IP_Address | Hostname`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

Puede especificar que un nombre de dominio, o una subred, o una dirección IP, o un nombre de host no sea servido por el servidor proxy especificado en la directiva `ProxyRemote`. Por ejemplo:

```
ProxyRemote   *   http://firewall.yourcompany.com:8080
NoProxy           .yourcompany.com
```

Aquí todas las solicitudes para `<anything>.yourcompany.com` (como `www.yourcompany.com`) están servidas por el servidor proxy local y el resto se dirigen al servidor proxy `firewall.yourcompany.com`.

ProxyDomain

`ProxyDomain` especifica el nombre de dominio para el servidor proxy.

Sintaxis: `ProxyDomain Domain`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

Cuando esta directiva tiene asignado el nombre de dominio local en una intranet, cualquier solicitud que no incluye a un nombre de dominio, tendrá este nombre de dominio adjunto; por ejemplo:

```
ProxyDomain       .evoknow.com
```

Cuando un usuario del dominio `evoknow.com` envía una solicitud para una URL del tipo `http://marketing/us.html`, la solicitud es regenerada como la siguiente URL:

```
http://marketing.evoknow.com/us.html
```

Observe que el nombre de dominio que se especifica en la directiva `ProxyDomain` debe tener un punto inicial.

CacheRoot

`CacheRoot` le permite cachear el disco. Puede especificar un nombre de directorio en el que el servidor proxy puede escribir archivos cacheados.

Sintaxis: `CacheRoot directory`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

El servidor Apache que está ejecutando el módulo proxy, debe escribir permisos para el directorio, por ejemplo:

```
CacheRoot  /www/proxy/cache
```

le dice a Apache que escriba datos cacheados del proxy en el directorio /www/proxy/cache. Observe que necesitará especificar el tamaño del caché utilizando el directorio CacheSize antes de que el servidor proxy pueda empezar a utilizar este directorio para caching. Necesitará utilizar otras directivas caché (que se discutirán más tarde) para crear una solución proxy con un disco cacheable.

CacheSize

CacheSize determina la cantidad de espacio en el disco (en kilobytes) que se debe utilizar para el cacheado del disco. Los archivos cacheados están escritos en el directorio especificado por la directiva CacheRoot.

Sintaxis: CacheSize kilobytes

Predefinido: CacheSize 5

Contexto: configuración del servidor, host virtual

NOTA: Aunque el servidor proxy puede escribir más datos de los que están especificados en el límite, el esquema de la colección basura del servidor proxy eliminará archivos hasta que la utilización se encuentre en el valor asignado o por debajo de éste. La asignación por defecto (5K) no es realista; recomiendo cualquier asignación entre 10MB y 1GB dependiendo de la carga de usuarios.

CacheGcInterval

CacheGcInterval determina el momento (en horas) en el que Apache debe comprobar los directorios caché para eliminar los archivos que han expirado. Esto también tienen lugar cuando Apache fuerza el límite del espacio de disco utilizado especificado por la directiva CacheSize.

Sintaxis: CacheGcInterval hours

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

CacheMaxExpire

CacheMaxExpire determina el momento (en horas) en el que expiran todos los documentos cacheados. Esta directiva invalida cualquier fecha de expiración especificada en el documento; por lo tanto, si un documento tiene una fecha de expiración posterior al máximo especificado por esta directiva, el documento se elimina igualmente.

Sintaxis: `CacheMaxExpire hours`

Predefinido: `CacheMaxExpire 24`

Contexto: configuración del servidor, host virtual

El valor por defecto permite que los documentos cacheados expiren en 24 horas. Si desea que los documentos expiren más tarde cambie este valor.

CacheLastModifiedFactor

`CacheLastModifiedFactor` determina un factor que se utiliza para calcular el momento de expiración cuando el servidor Web original no distribuye una fecha de expiración para el documento.

Sintaxis: `CacheLastModifiedFactor floating_point_number`

Predefinido: `CacheLastModifiedFactor 0.1`

Contexto: configuración del servidor, host virtual

El cálculo se realiza utilizando la fórmula siguiente:

```
expiry-period =  (last modification time for the document ) *
(floating point number)
```

Por lo tanto, si se modificó un documento hace 24 horas, el factor por defecto de 0.1 produce un cálculo de 2.4 horas para la expiración de ese documento. Si el tiempo de expiración calculado es superior al asignado por `CacheMaxExpire`, el período de expiración que marca `CacheMaxExpire` tiene prioridad.

CacheDirLength

Cuando está activado el caching de un disco, Apache crea subdirectorios en el directorio especificado por la directiva `CacheRoot`. Esta directiva determina el número de caracteres utilizados en la creación de los nombres de los directorios. Realmente no necesita cambiar el valor por defecto de esta directiva. Los usuarios curiosos que quieran saber cómo o por qué se han creado estos directorios, tienen una respuesta simplificada a continuación.

Sintaxis: `CacheDirLength length`

Predefinido: `CacheDirLength 1`

Contexto: configuración del servidor, host virtual

Apache utiliza un esquema de digestión cuando crea la ruta y el nombre de archivo para los datos de la URL que se van a cachear. Por ejemplo, cuando tiene activado el caching y accede a una URL (como `www.microsoft.com`) me-

diante su servidor proxy Apache, el servidor digiere esta URL de modo que más tarde puede recuperar datos rápidamente. Esta digestión es del tipo `1YSRxSmB20Q_HkqkTuXeqvw`. Si se utiliza el valor por defecto para las directivas `CacheDirLength` y `CacheDirLevels`, Apache almacena los datos encontrados en `www.microsoft.com` en un archivo llamado:

```
%CacheRoot%/1/Y/S/RRxSmB20Q_HkqkTuXeqvw
```

Aquí `%CacheRoot%` es el directorio especificado por la directiva `CacheRoot`. Los directorios `1/Y/S` se han creado para el valor por defecto de la directiva `CacheDirLevels`. Cuando este documento se solicita de nuevo utilizando la misma URL, Apache solo necesita recalcular la digestión para recuperar la página de la ruta especificada.

CacheDirLevels

`CacheDirLevels` determina el número de directorios que creará Apache para almacenar los archivos de datos cacheados. Remítase a la sección anterior para obtener la información relacionada.

Sintaxis: `CacheDirLevels levels`

Predefinido: `CacheDirLevels 3`

Contexto: configuración del servidor, host virtual

CacheDefaultExpire

`CacheDefaultExpire` proporciona el tiempo por defecto (en horas) que se utiliza para expirar un archivo cacheado cuando no se sabe el momento de la última actualización de un archivo. `CacheMaxExpire` no invalida esta asignación.

Sintaxis: `CacheDefaultExpire hours`

Predefinido: `CacheDefaultExpire 1`

Contexto: configuración del servidor, host virtual

NoCache

Sintaxis: `NoCache Domain_name | Subnet | IP_Address | Hostname . . .]`

Predefinido: ninguno

Contexto: configuración del servidor, host virtual

La directiva `NoCache` determina una lista de host, nombres de dominio y direcciones IP, separados por espacios, para los que no tiene lugar un cacheado. Esta directiva debería utilizarse para deshabilitar el caching de los servidores Web en una intranet.

Observe que el servidor proxy también relaciona nombres parciales de un host. Si quiere deshabilitar todo el caching, utilice:

```
NoCache *
```

Configurar un servidor proxy Apache

En esta sección le mostraré cómo configurar Apache (con `mod_proxy`) como un servidor proxy forward. Una vez que tiene el módulo `mod_proxy` compilado en Apache (tal y como se ha discutido), establecer un servidor proxy es muy sencillo.

Para activar el servidor proxy, necesita asignarle el valor `On` al `ProxyRequests` en un archivo `httpd.conf`. Cualquier configuración adicional depende de lo que quiera hacer con el servidor proxy.

Independientemente de lo que decida hacer con él, cualquier directiva, que quiera utilizar para controlar el comportamiento del servidor proxy, debe ir dentro de un contenedor `<Directory . . .>` especial que es de la forma siguiente:

```
<Directory proxy:*>
. . .
</Directory>
```

El asterisco es un comodín para la URL solicitada. En otras palabras, cuando el servidor Apache procesa una solicitud para `www.evoknow.com`, lo hace del siguiente modo:

```
<Directory proxy:http://www.evoknow.com/>
. . .
</Directory>
```

También puede utilizar el contenedor `<Directory ~ /RE/>`, que utiliza expresiones regulares que le permiten una flexibilidad mayor en la definición de la configuración proxy.

Por ejemplo:

```
<Directory ~ proxy:http://[^:/]+/.*>
. . .
</Directory>
```

A continuación vamos a ver unas cuantas configuraciones proxy que se utilizan habitualmente.

Escenario 1: conectar una IP privada a Internet

En este escenario, sólo hay un ordenador en la red que tiene una dirección IP de Internet rutable, tal y como se muestra en la figura 10.3. Este ordenador ejecuta el servidor proxy de Apache con `ProxyRequest` fijada en `On`, y no se necesita configuración proxy adicional. El servidor proxy sirve todas las solicitudes.

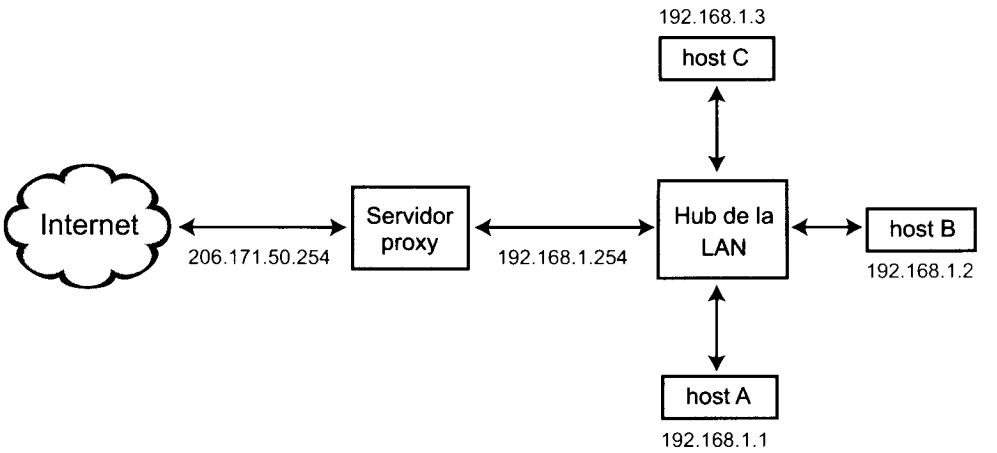


Figura 10.3. Proxy para una red IP privada

En este tipo de configuración, el servido proxy necesita estar multialojado; en otras palabras, necesita tener acceso a la red privada no rutable (`192.168.1.0`) y a la red con IP rutable (`206.171.50.0`). De algún modo, este proxy actúa como un firewall para la red privada aunque ya lo haga el pool de direcciones IP elegido. El proxy permite que el host acceda a los servicios de Internet, como son la Web, FTP, y similares.

Escenario 2: caching sitios web remotos

Como hay una gran cantidad de contenido Web estático en Internet y en intranet, cachearlo en un servidor proxy local puede ahorrar una gran cantidad de ancho de banda. Un servidor proxy que permite cachear, únicamente sirven los documentos solicitados cuando el caché contiene un documento que ha expirado o cuando el documento solicitado no está en el caché. Para permitir el caching en su servidor proxy, tiene que especificar las directivas de caching dentro de un contendor de directorios especial. Por ejemplo:

```
<Directory proxy:*>
    CacheRoot /www/cache
    CacheSize 1024
    CacheMaxExpire 24
</Directory>
```

Esta configuración define un servidor proxy de caching que escribe archivos caché en el directorio `/www/cache`. Está permitido escribir 1024K de datos (1MB) y el caché deberá expirar cada día (24 horas).

Si quiere evitar que alguien abuse de su servidor proxy, puede restringir el acceso mediante autentificación del host o mediante autentificación nombre de usuario/ contraseña.

Para controlar qué hosts tienen acceso al servidor proxy, puede crear una configuración del siguiente tipo:

```
<Directory proxy:*>

    AuthType Basic
    AuthName Proxy
    order deny,allow
    deny from all
    allow from myhost.evoknow.com

</Directory>
```

Esta configuración deniega el acceso a todos excepto a `myhost.evoknow.com`. Si quiere utilizar autentificación nombre de usuario/ contraseña, puede utilizar algo parecido a lo siguiente:

```
<Directory proxy:*>
    AuthType Basic
    AuthName Proxy
    AuthUserFile /path/to/proxy/.htpasswd
    AuthName Proxy
    require valid-user
</Directory>
```

Si no está seguro de cómo crear los archivos de contraseña necesarios, remítase a capítulo anteriores.

Además, es posible restringir el acceso para un protocolo. Por ejemplo:

```
<Directory proxy:http:*>
. . .
</Directory>
```

le permite controlar cómo se procesan las solicitudes HTTP en su servidor proxy. Del mismo modo, puede utilizar el esquema siguiente para controlar cómo maneja el servidor proxy cada protocolo:

```
<Directory proxy:ftp:*>
. . .
</Directory>
```

o

```
<Directory proxy:https:*>
. . .
</Directory>
```

También puede crear un host virtual exclusivamente para servidores proxy. En ese caso, las directivas deberían estar dentro del contenedor `<VirtualHost>`:

```
<VirtualHost proxy.host.com:*>
. . .
</VirtualHost>
```

Escenario 3: crear una copia local de un sitio Web

Un sitio Web mirror es una copia local de un sitio Web remoto. Puede utilizar el servidor proxy para realizar este tipo de copias del sitio Web `www.apache.org` para que sus usuarios puedan conectarse a ese sitio y acceder a la información de Apache rápidamente, puede utilizar el servidor proxy para crearlo, del siguiente modo:

```
ProxyPass /  www.apache.org/
CacheRoot /www/cache
CacheDefaultExpire 24
```

Esto convierte el servidor proxy en un mirror del sitio Web `www.apache.org`. Esta configuración convierte a mi servidor proxy `blackhole.evoknow.com` en un mirrow Apache. Cuando un usuario introduce la URL `http://blackhole.evoknow.com`, el usuario recibe la página de inicio del mirrow Apache como si estuviese accediendo a `www.apache.org`.

Antes de que haga una copia local de un sitio Web, es importante que obtenga permiso, ya que suelen estar implicados asuntos relacionados con el copyright.

Preparar un navegador Web para utilizar un proxy

Una vez que tiene configurado su servidor proxy, está preparado para establecer el navegador Web en sus host clientes. Los navegadores Web populares hacen muy sencilla la utilización de los servidores proxy. En las siguientes secciones, le mostraré cómo configurar Netscape Navigator 6 y Microsoft Internet Explorer (IE) 5.5 para proxy. Hay dos modos de establecer un servidor proxy para estos navegadores: configuración manual y configuración automática.

La configuración manual de los navegadores Web para proxy no es difícil. Sin embargo, si tiene que configurar muchos ordenadores de usuarios, este proceso se

puede convertir en un gran lío cada vez que tenga que cambiar su configuración proxy. Aquí es donde tiene sentido la configuración automática de proxy en los navegadores.

Configuración manual del proxy

Se utiliza cuando tiene únicamente unas cuantas máquinas clientes y sus configuraciones de proxy no cambian con frecuencia. Si sus necesidades son distintas, por ejemplo, tiene cientos de máquinas clientes, debe saltarse esta sección y pasar a "Configuración automática del proxy."

Configurar Netscape manualmente

Los siguientes pasos le guían en la configuración manual de Netscape:

1. Elija Editar>Preferencias del menú de Navigator. Debería ver una ventana de diálogo como la que se muestra en la figura 10.4.

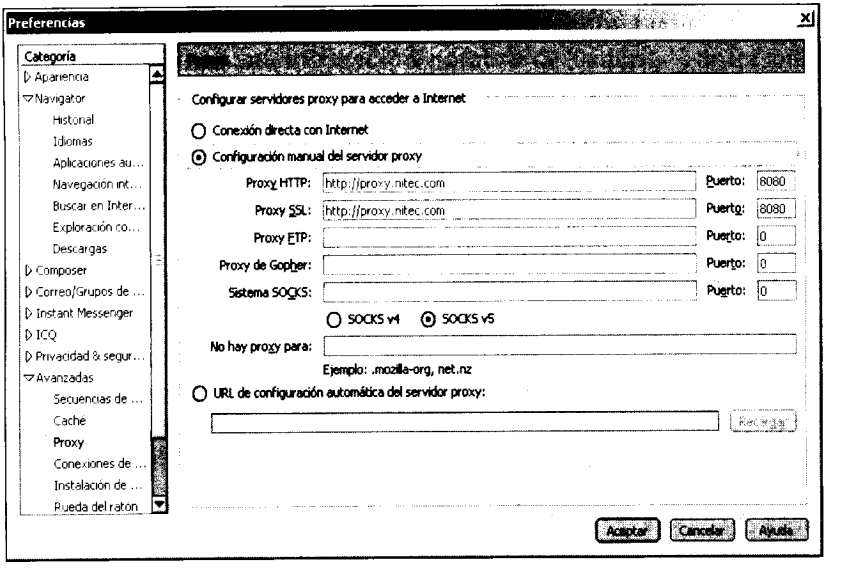


Figura 10.4. La ventana de establecimiento manual de un proxy en Netscape Navigator

2. Haga clic en la categoría Avanzadas.
3. Haga clic en la categoría Proxy.
4. Haga clic en la opción Configuración manual del servidor proxy.
5. Introduzca las URL del servidor proxy para los campos de entrada de datos HTTP, FTP y Security (HTTPS), junto con la información del puerto. Como estoy utilizando un solo servidor proxy para estos protocolos, la

URL (`http://proxy.evoknow.com /`) y el puerto (8080) son los mismos. Si tiene distintos servidores proxy para cada uno de estos servicios, tiene que especificarlos.

6. Una vez que ha introducido la información, realice una solicitud de un documento remoto para saber si su proxy está funcionando. Un buen modo de determinar qué es lo que está ocurriendo es monitorizar el acceso al servidor proxy y los registros de error. En la mayoría de los sistemas Unix puede utilizar un comando como el que tiene a continuación para ver las entradas de registro tal y como se han escrito en el archivo:

```
tail -f  /path/to/access/log
```

Configurar Internet Explorer manualmente

Vamos a configurar Microsoft Internet Explorer para Windows. Para configurar manualmente Internet Explorer para proxy, siga los siguientes pasos:

1. Elija Herramientas>Opciones de Internet. Esto abrirá la caja de diálogo que se muestra en la figura 10.5.

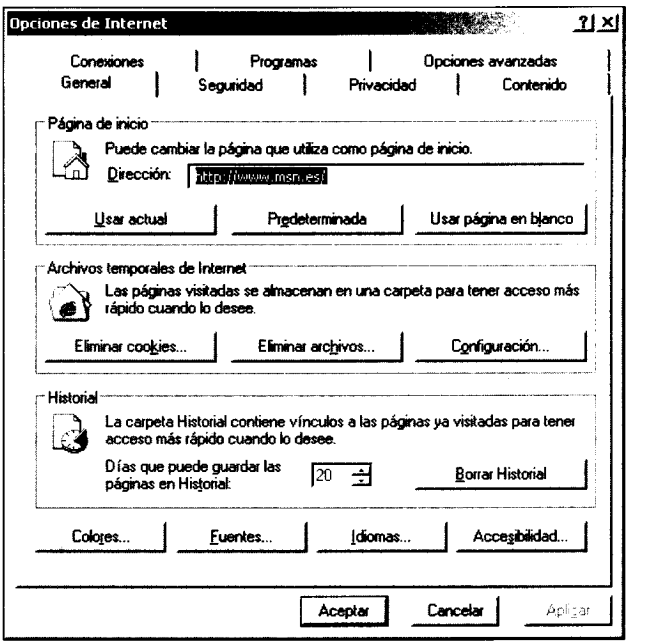


Figura 10.5. La ventana de establecimiento manual de un proxy en Microsoft Internet Explorer

2. Haga clic en la pestaña de Conexiones y haga clic en el botón de Configuración de LAN. Supongo que está utilizando una conexión LAN gateway/router a Internet.

3. Seleccione la opción Utilizar un servidor proxi para su LAN...
4. Introduzca la URL del servidor proxy y el número de puerto en los campos proporcionados.
5. Haga clic en Aceptar para completar la configuración.

TRUCO: Si quiere especificar información sobre el servidor proxy distinta para los distintos protocolos, puede utilizar el botón Opciones avanzadas para obtener la ventana que se muestra en la figura 10.6.

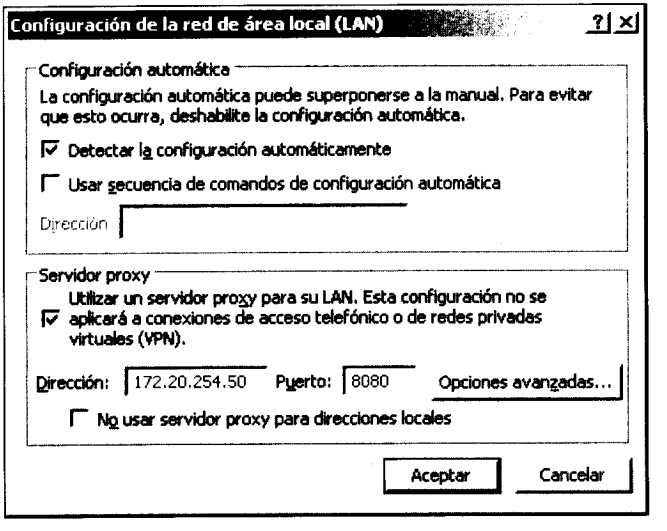


Figura 10.6. La ventana de establecimiento avanzado de un proxy en Internet Explorer

En este caso, al igual que en el Netscape Navigator, puede especificar diferentes ajustes para los servidores proxy. Cuando haga clic en Aceptar y en el botón Aplicar para aplicar los nuevos ajustes, su navegador se configurará para utilizar el servidor proxy.

Configuración automática del proxy

Los expertos en Netscape Communications pensaron en los problemas implicados en la configuración manual de un proxy para varios ordenadores clientes, y sugirieron un modo de superar este problema. Cuando se inicia el navegador Web, carga la función desde el archivo JavaScript (más tarde se discute cómo el navegador accede a este archivo) y llama a `FindProxyForURL` para cada solicitud URL. El navegador suministra los argumentos host y URL a la función para que pueda devolver la mayor parte de la configuración proxy.

NOTA: Microsoft también tiene opciones de autoconfiguración para Internet Explorer. Por desgracia, la incorporación de la autoconfiguración en el navegador en Microsoft es algo más complicada. Debe obtener el Internet Explorer Administrator Kit (IEAK) para crear archivos de autoconfiguración. Como obtener IEAK requiere un acuerdo de licencia que informe trimestralmente a Microsoft sobre la utilización de IEAK, este autor no ha obtenido el kit. Sin embargo, tengo la confirmación de una buena fuente de que la documentación IEAK tiene un escenario de configuración de proxy parecido al de Netscape y puede incluso utilizar los mismos scripts. Esta sección se aplica tanto a IE como a Navigator. La única diferencia es que si quiere utilizar IE, debe tener en cuenta la creación de archivos apropiados utilizando IEAK.

La autoconfiguración proxy se realiza utilizando un JavaScript especial. Esto es así tanto para Netscape Navigator como para IE. El JavaScript especial cumple estos requisitos:

- El JavaScript de autoconfiguración proxy debe implementar una función llamada `FindProxyForURL`. Esta función tiene el siguiente esqueleto:

```
function FindProxyForURL(url, host) {

   // código java script
   return "proxy to use for servicing the URL";
}
```

- Los argumentos que recibe esta función son `url` y `host`. El argumento `url` es la URL completa que se ha solicitado y el argumento `host` es el nombre del host extraído de la URL. Por ejemplo, cuando el navegador Web detecta una solicitud para una página Web, llama a la función:

```
ret = FindProxyForURL(", )
```

NOTA: El argumento del host en la función es realmente una subcadena entre el `://` y el primer `:` o el primer `/`. El número de puerto no está incluido en este parámetro.

- La función debe devolver una cadena con la configuración proxy necesaria para una solicitud URL determinada. Los valores aceptables de las cadenas que representan una configuración proxy se muestran en la tabla 10.1.

Tabla 10.1. Valores de cadenas aceptables para la configuración del proxy

Cadena	Significado
`NULL`	Cuando se devuelve un valor NULL (no la cadena NULL), le dice al navegador que no utilice un proxy para esta solicitud.
`DIRECT`	Las conexiones se podrían realizar directamente, sin proxies.
`PROXY host:port;`	Debería utilizarse el proxy especificado.
`SOCKS host:port;`	Debería utilizarse el servidor SOCKS especificado.

Asignar valores de retorno para FindProxyForURL

Tal y como se discute en la tabla 10.1, se pueden devolver cuatro valores. Obviamente, los valores interesantes son `DIRECT` y `PROXY`. Cuando tiene varios servidores proxy o servidores SOCKS, puede devolver una lista en lugar de un solo par `host:port`. Por ejemplo, la siguiente configuración proxy:

```
PROXY best-proxy.evoknow.com:8080; PROXY good-
proxy.evoknow.com:8081; PROXY soso-proxy.evoknow.com:8082
```

le dice al navegador que primero lo intente `best-proxy.evoknow.com`, y si falla, que lo intente el siguiente (`good-proxy.evoknow.com`), y así sucesivamente. Observe que cada par `host:port` está separado por un punto y coma y que la palabra clave PROXY se repite en cada par. Si fallan todos los servidores proxy, se le preguntará al usuario antes de intentar una conexión directa. Cuando fallan todos los proxies y no está especificada ninguna opción `DIRECT`, el navegador le pregunta al usuario si deberían ignorarse los proxies de forma temporal e intentar conexiones directas. Para evitar la interacción con el usuario la configuración anterior se convierte en la siguiente:

```
PROXY best-proxy.evoknow.com:8080; PROXY good-
proxy.evoknow.com:8081; PROXY soso-proxy.evoknow.com:8082;
DIRECT
```

Como la conexión directa ya está especificada como último recurso, no se le preguntará al usuario antes de realizar tal tipo de conexión en caso de fallo total del proxy. También puede mezclar `PROXY` y `SOCKS`. Por ejemplo:

```
PROXY best-proxy.evoknow.com:8080; SOCKS
socks4.evoknow.com:1080; DIRECT
```

En este caso, se utilizará el proxy basado en SOCKS, cuando el servidor proxy principal `best-proxy.evoknow.com` falle en la respuesta.

Cuando un proxy falla en la respuesta, el navegador Web vuelve a utilizar el proxy pasados 30 minutos. Si la subsolicitud posterior falla, el intervalo se alarga otros 30 minutos.

Utilizar funciones predefinidas en FindProxyForURL

Para ayudar a los administradores Web (que además deben programar en JavaScript), hay disponible un conjunto de funciones predefinidas. En la tabla 10.2 se muestran estas funciones y su descripción.

Tabla 10.2. Funciones predefinidas para el script de configuración automática de programación de un proxy

Nombre de la función	Explicación	Ejemplos
isPlainHostName(host)	Devuelve true si no hay un punto en el nombre del host. En otras palabras, si no está incluido el nombre del dominio.	isPlainHostName("blackhole") devuelve true. isPlainHostName("blackhole.evoknow.com") devuelve false. evoknow.com.
DnsDomainIs(host, domain)	Devuelve true si el host pertenece al dominio. Observe que el nombre de dominio debe contener un punto inicial.	dnsDomainIs("www.evoknow.com", ".evoknow.com") devuelve true. dnsDomainIs("www.apache.org", ".evoknow.com") devuelve false. evoknow.com.
localHostOrDomainIs (host, fqdnhost)	Devuelve true si el host que forma parte de fqdnhost (fully qualified host name) coincide con el host.	localHostOrDomainIs ("a.b.com", "a.b.com") devuelve true. LocalHostOrDomainIs ("a.b", "a.b.com") devuelve true. LocalHostOrDomainIs ("a.b.org", "a.c.com") devuelve false.
isResolvable(host)	Si un servidor DNS puede traducir el nombre del host a una dirección IP, devuelve true; en caso contrario, devuelve false. La utilización de esta función puede hacer más lentos los navegadores porque se necesita una consulta DNS para llevar a cabo la prueba.	isResolvable("{hyperlink}") devuelve true (porque {hyperlink} tiene registros DNS).

Nombre de la función	Explicación	Ejemplos
isInNet(host, IP address pattern, netmask)	Devuelve true si la dirección IP del host coincide con el patrón especificado en el segundo argumento. La búsqueda de coincidencia se realiza utilizando la máscara de red del siguiente modo: si uno de los octetos de la máscara es un 255, debe coincidir el mismo octeto de la dirección IP del host. Si un octeto de la máscara es 0, se ignora el mismo octeto de la dirección IP del host. La utilización de esta función puede hacer más lentos los navegadores porque se necesitará una consulta DNS para realizar la prueba.	Si el host tiene una dirección IP 206.171.50.51, isInNet(host, "206.171.50.50", "255.255.255.0") devuelve true porque, de acuerdo con la máscara de red, únicamente deben coincidir los tres primeros octetos y se debe ignorar el último.
dnsResolve(host)	Devuelve la dirección IP del host, en caso de éxito. Observe que la utilización de esta función puede hacer los navegadores más lentos porque es necesaria una consulta DNS para realizar la prueba.	dnsResolve("proxy.evoknow.com") devuelve "206.171.50.50".
myIpAddress()	Devuelve la dirección IP del host que está ejecutando el navegador Web. Observe que la utilización de esta función puede hacer los navegadores más lentos porque es necesaria una consulta DNS para realizar la prueba.	var hostIP = myIpAddress() devuelve el IP del host del navegador Web y almacena en él una variable llamada hostIP.
dnsDomainLevels (host)	Devuelve el número de niveles de dominio en el nombre del host.	dnsDomainLevels("www.nitec.com") devuelve 2.
shExpMatch(string, shellExpression)	Devuelve true si la cadena coincide con la expresión shell.	shExpMatch("path/to/dir", "*/to/*") devuelve true. shExpMatch("abcdef", "123") devuelve false.
WeekdayRange(weekday1, weekday2, gmt)	Únicamente es necesario el primer argumento weekday1. Devuelve true si el día en el que se ejecuta esta función es igual a weekday1 o si se encuentra en el rango de weekday1 a weekday2. Si el tercer parámetro, gmt, es GMT entonces se utiliza la hora GMT en lugar de la hora local. Los valores aceptables para weekday1 o weekday2 son SUN, MON, TUE, WED, THU, FRI o SAT.	weekdayRange("FRI") devuelve true si estamos a Friday (viernes) según la hora local. weekdayRange("MON", "FRI", "GMT") devuelve true si estamos en el rango de Monday-Friday (lunes a viernes) en la hora GMT.

Nombre de la función	Explicación	Ejemplos
dateRange(day) dateRange(day1, day2) dateRange(month) dateRange(month1, month2) dateRange(year) dateRange(year1, year2) dateRange(day1, month1, day2, month2) dateRange(month1, year1, month2, year2) dateRange(day1, month1, year1, day2, month2, year2) dateRange(day1, month1, year1, day2, month2, year2, gmt)	Devuelve true si el día, el año y el mes actuales, o los tres, se encuentran en el rango. El valor de day (día) puede ser 1-31; el de month (mes) puede ser JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV o DEC; el valor de year (año) es un número de cuatro dígitos; gmt es "GMT" o nada (hora local).	dateRange(31) devuelve true si el día actual es el 31. dateRange("JAN", "APR") devuelve true si el año actual se encuentra en el rango de January (enero) a April (abril). dateRange(1995) devuelve true si el año actual es 1995.
timeRange(hour) timeRange(hour1, hour2) timeRange(hour1, min1, hour2, min2) timeRange(hour1, min1, sec1, hour2, min2, sec2) timeRange(hour1, min1, sec1, hour2, min2, sec2, gmt	Devuelve true si la hora, los minutos o los segundos especificados son los actuales. Si se especifica un rango, entonces devuelve true cuando la unidad de tiempo correspondiente se encuentra dentro del rango especificado. El valor de hour (hora) puede ser 0-23; el valor de min (minutos) puede ser 0-59; el valor de second (segundos) puede ser 0?59; y gmt es "GMT" o nada (hora local).	timeRange(9, 17) devuelve true si la hora actual se encuentra entre las 9 a.m. y las 5 p.m.

Con la ayuda de funciones predefinidas y con sus funciones personalizadas, puede escribir `FindProxyForURL` de modo que devuelva la cadena de configuración de proxy apropiada para cada solicitud.

A continuación proponemos unos cuantos ejemplos de escenarios en los que se puede escribir la función `FindProxyForURL` de distintos modos.

Escenario 1: utilizar un proxy únicamente para solicitudes URL remotas

En este escenario, la idea es decirle al navegador Web que el proxy debería ser únicamente para solicitudes URL remotas, tal y como se muestra en la figura 10.7. En este caso, se evalúa una solicitud a `http://www.microsoft.com`, desde el host A, mediante el método `FindProxyForURL()`, el cual devuelve `proxy.nitec.com:8080` como el servidor proxy y entonces también le instruye para que intente acceder directamente si se ha caído el proxy. De igual

modo, un navegador Web en el host B utiliza el método `FindProxyForURL()` para evaluar cómo acceder a `http://www.nitec.com`. La figura muestra que el método `FindProxyForURL()` instruye al navegador para que acceda al sitio directamente.

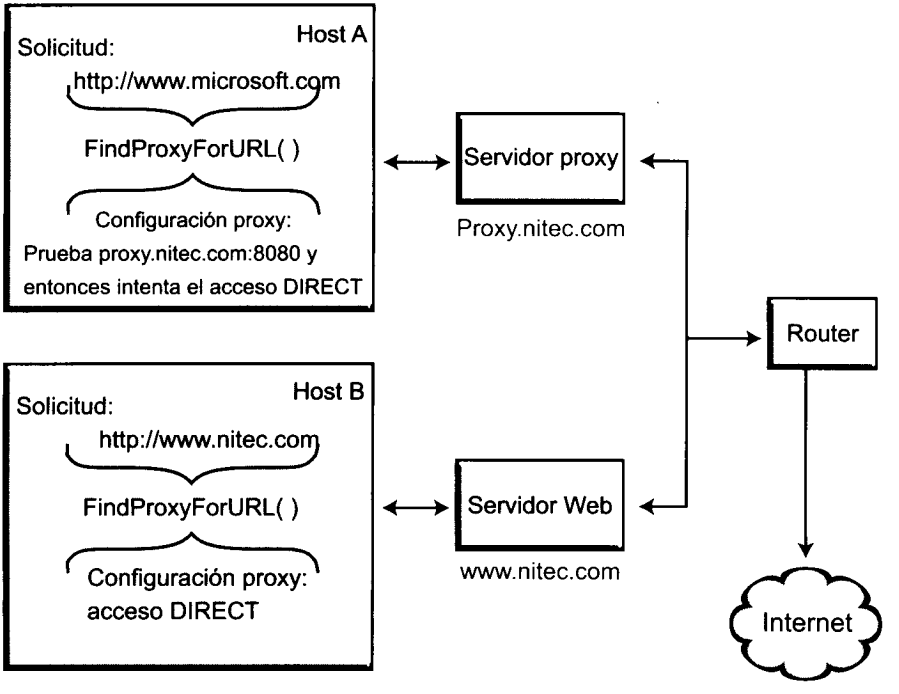


Figura 10.7. Utilizar un proxy únicamente para solicitudes URL remotas

El listado 10.1 es un ejemplo de una función `FindProxyForURL`.

Listado 10.1. Utilizar un proxy únicamente para solicitudes URL remotas

```
function FindProxyForURL(url, host) {

    // Comprueba si el host es un host local.
    // Si es un host local
    // especifica una conexión DIRECT (es decir, no proxy)
    // o utiliza el proxy.

if (isPlainHostName(host) || dnsDomainIs(host, ".nitec.com"))

        return "DIRECT";

else

       return "PROXY proxy.nitec.com:8081; DIRECT";

}
```

Cuando un usuario del servidor Web realiza una solicitud a URL `http://www.domain.com`, el navegador llama a `FindProxyForURL` con el argumento `url` asignado a `http://www.domain.com` y el host asignado a `www.domain.com`. La función llama primero a la función `isPlainHostName` para ver si la solicitud es sólo para el host (solo `www`) o no. Como no es así, `isPlainHostName` devuelve false. Ahora llama a la función `dnsDomainIs` para comprobar si se encuentra en el dominio `.nitec.com`. Esto también devuelve false. Como ambas pruebas devuelven false, se ejecuta la parte `else` de la sentencia condicional. En otras palabras, la solicitud URL de `http://www.domain.com` devuelve la siguiente configuración de proxy al navegador Web:

```
PROXY proxy.nitec.com:8081; DIRECT
```

Esto le dice al navegador Web que utilice el servidor proxy llamado `proxy.nitec.com` en el puerto 8081 si este no ha caído. Si ha caído, la solicitud debería servirse con una solicitud HTTP directa a `http://www.domain.com`. Para la mayor parte de las instalaciones proxy, esta configuración es suficiente. Vamos a ver un escenario más complicado.

Escenario 2: utilizar varios servidores proxy

En este escenario, hay varios servidores proxy. La figura 10.8 nos muestra una red en la que hay tres servidores proxy: `http-proxy.nitec.com` que se utiliza para todas las solicitudes HTTP a las URL remotas; `ftp-proxy.nitec.com` que se utiliza para todas las solicitudes FTP a URL remotas; y `ssl-proxy.nitec.com` que se utiliza para todas las solicitudes HTTPS a URL remotas. El resto de las solicitudes URL que utilizan otros protocolos como GOPHER, NEWS, y similares, se conectan directamente. Todos los tipos de solicitudes locales se sirven también directamente. Para implementar esta configuración, `FindProxyForURL` se convierte en una función algo más compleja, algo parecido a lo que se muestra en el listado 10.2.

Listado 10.2. FindProxyForURL para una configuración de servidores multiproxy

```
function FindProxyForURL(url, host) {

    //
    // ¿Es esta la URL local? Si es así, utiliza una
    // conexión DIRECT
    //
    if (isPlainHostName(host) ||

         dnsDomainIs(host, ".nitec.com")) {

        return "DIRECT";

    } else {
```

```
        // De acuerdo, la URL es remota, por lo tanto, determina
cuál es
        // proxy que hay que utilizar.

        if (url.substring(0, 5) == "http:") {

           return "PROXY http-proxy.nitec.com:8080";

        } else if (url.substring(0, 4) == "ftp:") {

            return "PROXY ftp-proxy.nitec.com:8080";

        } else if (url.substring(0, 6) == "https:") {

             return "PROXY ssl-proxy.nitec.com:8080";

        } else{

            return "DIRECT";

        }
    }
}
```

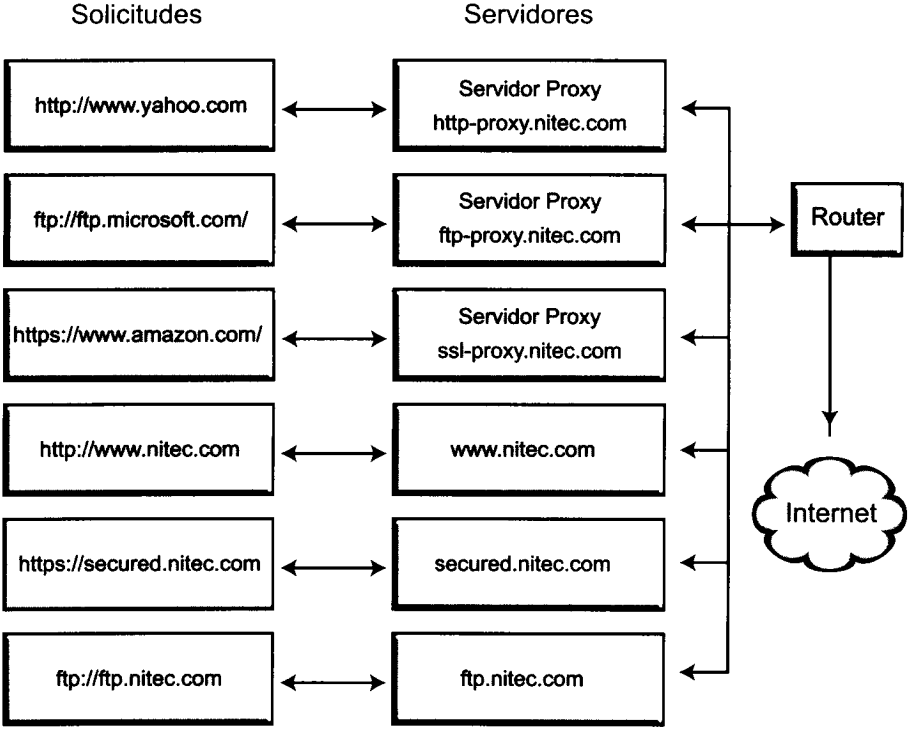


Figura 10.8. Utilizar varios servidores proxy

Esta función comprueba primero si la solicitud URL es local. Si es local, entonces se sirve directamente. Si es una solicitud para un servidor remoto, se realiza una búsqueda de coincidencia del protocolo de la URL para localizar el servidor proxy adecuado. Sin embargo, sólo se reconocen los protocolos HTTP, FTP y HTTPS, y las URL que solicitan recursos remotos utilizando dichos protocolos son dirigidas a servidores proxy. Cuando una solicitud URL remota no coincide con ninguno de estos protocolos de estado, se conecta directamente.

Escenario 3: generar FindProxyForURL dinámicamente utilizando un script CGI

También es posible personalizar la configuración de su servidor proxy basándose en el host al que está accediendo el servidor proxy. Esto se puede realizar utilizando un script CGI que realice una salida de `FindProxyForURL` de forma distinta dependiendo del `REMOTE_HOST` (el host del navegador). El listado 10.3 muestra uno de estos script, `proxy.pl`, escrito en Perl.

Listado 10.3. proxy.pl

```
#!/usr/bin/perl
#
# Un script Perl que da lugar a la configuración de un servidor
proxy.
# $Author$
# $Revision$
# $Id$

# Obtiene la IP del host remoto de la variable de entorno CGI
# REMOTE_HOST
my $client = $ENV{REMOTE_HOST};

# Imprime el tipo de contenido necesario para permitir que el
# navegador sepa que se trata de una configuración proxy.
print "Content-type: application/x-ns-proxy-autoconfig\n\n";

# Si la solicitud procede de un host con la dirección IP
# 206.171.50.51 entonces saca la configuración del proxy
# desde una subrutina &specialClient
#
if ($client =~ /206\.171\.50\.51/){

    &specialClient;

} else {

    # Si la solicitud procede de cualquier otro cliente,
entonces
    # envía la configuración proxy al resto de los clientes

    &otherClients;
```

```
}

exit 0;

sub specialClient{
#
# Esta subrutina saca una configuración del servidor proxy
#

   print <<FUNC;

        function FindProxyForURL(url, host)
        {
            if (isPlainHostName(host) ||
                dnsDomainIs(host, ".nitec.com"))
                return "DIRECT";
            else if (shExpMatch(host, "*.com"))
                return "PROXY com-proxy.nitec.com:8080; "

            else if (shExpMatch(host, "*.edu"))
                return "PROXY edu-proxy.nitec.com:8080; "

            else
                return "DIRECT";
        }
FUNC

}

sub otherClients{
#
# Esta subrutina saca una configuración del servidor proxy
#

   print <<FUNC;

        function FindProxyForURL(url, host)
        {
            return "DIRECT";
        }

FUNC

}
```

Este script da lugar a una configuración especial del servidor proxy para el host con la dirección IP `206.171.50.51`; el resto de los host obtienen una configuración distinta. Para acceder a esta configuración proxy, puedo preparar el Netscape Navigator o el IE para dirigir este script a `http://www.nitec.com/cgi-bin/proxy.pl`. Por ejemplo, en IE puede especificar una URL igual a la anterior como la dirección del script de configuración

automática en Tools>Internet Options>Connections>LAN Settings>Use automatic configuration, excepto si le ha pedido al navegador que solicite un script CGI en lugar de un archivo `.pac`. Pero como el script envía el tipo de contenido de un archivo `.pac`, el navegador no tiene que preguntarse por qué obtiene la configuración proxy desde un script CGI en vez de desde un archivo `.pac`. Aunque el script del ejemplo no tiene muchas funciones, puede utilizar scripts parecidos para configuraciones proxy más complejas.

11 Ejecutar sitios Web perfectos

En este capítulo

1. Creamos un ciclo Web para su organización.
2. Generamos sitios Web basados en plantillas utilizando el makepage.
3. Publicamos en una intranet utilizando el método PUT HTTP.
4. Estandarizamos sus estándares.
5. Hacemos sus Web más intuitivas.
6. Promocionamos su sitio Web en Internet.

En este momento, tendrá probablemente, uno o más sitios Web preparados y ejecutándose en su nuevo servidor Web Apache. Toda su empresa le está felicitando por su maravilloso trabajo. Está usted en el cielo, ¿verdad? ¡Incorrecto! Muy pronto sus compañeros le preguntarán cómo actualizar sus páginas en el sitio Web. Por ejemplo, el departamento de marketing podría llamar y preguntarle cómo actualizar la información referente a los precios, o el departamento legal le podría preguntar cómo puede añadir más contenido legal en uno de los sitios Web.

Esto es lo que les ocurre a los administradores Web de las organizaciones medianas y grandes. Estos administradores pronto se encuentran en el medio de un caos de solicitudes actualizadas y una lista de deseos. Por lo tanto, ¿cómo manejamos ahora la Web? En este capítulo, aprenderá a crear un entorno de gestión profesional de la Web que mantendrá a sus desarrolladores Web y a usted cuerdos y en sincronía con la Web.

Este capítulo trata varios aspectos relacionados con el desarrollo de un sitio Web perfecto. Un sitio Web perfecto exhibe las características siguientes:

- Contenido de alta calidad: por supuesto, si no tiene un contenido útil y entretenido, no van a visitar su página. Sin embargo, el tipo de contenido que funciona depende del propósito de su sitio Web.
- Una presencia y un contenido consistentes: los sitios Web que tienen un tema consistente a lo largo de todas las páginas son más atractivos y suelen indicar un proceso elaborado. Crear una presencia y un contenido consistentes requiere herramientas y un proceso sistemático. Este capítulo le introduce en un proceso llamado *ciclo Web*, que requiere que utilice tres fases (desarrollo, prototipo y producción) para manejar sus sitios Web.
- Publicación automatizada: mi experiencia es que desarrollar constantemente contenido nuevo y excitante es un gran desafío en sí mismo. Si añade tareas manuales de presentación de contenido, muy pronto tendrá las cosas fuera de control. Por ejemplo, si tiene tres autores de contenido escribiendo páginas HTML para un sitio, debería plantear una presentación y un objetivo comunes, podría utilizar plantillas HTML e integrar el contenido utilizando un proceso automatizado. En este capítulo se discuten algunos de estos procesos.
- Prácticas estándar: para que el sitio Web sea intuitivo, hay una serie de guías que hay que seguir. Discutiremos algunas de las más importantes en este capítulo.

Ciclo de desarrollo Web

Por desgracia, los típicos proyectos de desarrollo Web no comienzan con el diseño de un sitio Web manejable. En la mayoría de los proyectos, gran parte del tiempo se emplea en ejecutar y desarrollar el contenido; y no es habitual preocuparse sobre aspectos de gestión a largo plazo. Irónicamente, tan pronto como todo parece estar funcionando, las cosas empiezan a fallar debido a la falta de claridad, a la falta de un ciclo sostenible. En esta sección, trataremos el ciclo Web, el cual le permite crear una solución Web muy manejable.

Un ciclo Web consiste en tres fases: desarrolo, prototipo y producción. Implementando cada una de estas fases, puede crear un sitio Web manejable y sostenible. La figura 11.1 muestra un diagrama de alto nivel de un ciclo Web.

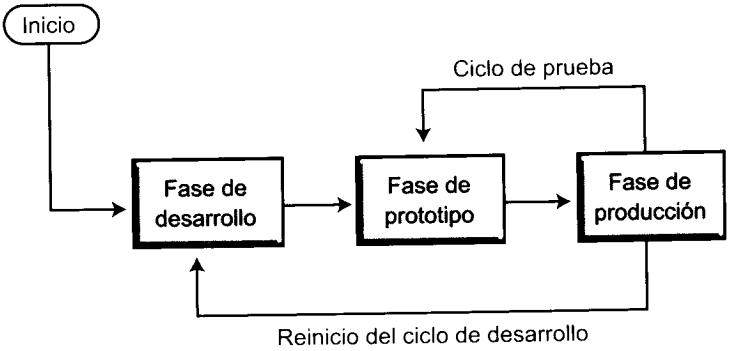


Figura 11.1. Un diagrama de alto nivel de un ciclo Web

Tal y como muestra la figura, un ciclo Web se inicia con la fase de desarrollo, continúa con la fase prototipo y termina con la fase de producción. Cuando se reinicia el ciclo, sin embargo, se inicia con la fase de producción y repite la ruta previa del ciclo. Las fases del ciclo son:

- **Fase de desarrollo:** en esta fase, comienza a desarrollar su contenido Web. El contenido, ya sea documentos HTML o scripts CGI o cualquier otra cosa, queda totalmente desarrollado y probado en esta fase. Una vez que los desarrolladores están totalmente seguros de que su trabajo está listo para integrarse con el sitio o sitios Web, el nuevo contenido desarrollado pasa a la siguiente fase.
- **Fase prototipo:** la fase de prototipo permite la integración del nuevo contenido desarrollado con el contenido existente, y permite el desarrollo de ciclos de prueba. Una vez en la fase de prototipo, los desarrolladores no siguen participando en el proceso. En este proceso, se introducen sujetos prueba que no son desarrolladores, para eliminar los posibles prejuicios a la hora de evaluar la corrección del contenido. En este momento, podrá localizar algunos problemas o quizá obtenga un conjunto satisfactorio de pruebas. En este último caso, está preparado para pasar el nuevo contenido, desarrollado, practicado y comprobado, a la fase de producción. Si se han encontrado problemas en este contenido nuevo, tendrá que comenzar desde la fase de desarrollo una vez que los desarrolladores hayan solucionado el problema en el área de desarrollo. No permita que los desarrolladores solucionen los problemas en el área prototipo.
- **Fase de producción:** esta fase consiste en la realización de una copia de seguridad (hacer un backup) y en las tareas de despliegue de contenido. Primero, ha de realizar una copia de seguridad de su contenido existente (funcional), y luego tiene que pasar el contenido de la fase de prototipo al espacio de producción Web. Esto tiene que suceder tan rápido como sea posible para reducir las desconexiones de visitantes y prevenir pérdidas de datos Web recolectados.

Cuando esté listo para empezar otro ciclo de desarrollo (reiniciar el proceso completo), copie el contenido de la fase de producción y páselo a la fase de desarrollo, de modo que los desarrolladores puedan trabajar en él. El ciclo continúa del mismo modo cada vez que lo necesite.

¿Qué ventajas tiene todo esto para usted? Le ofrece opciones de seguridad y de manejabilidad. Por ejemplo, si está desarrollando contenido y lo está descargando directamente en su sistema de producción antes de realizar la totalidad de las pruebas, está trabajando en condiciones límite. En la mayor parte de los casos, los desarrolladores de contenido querrán tener su contenido probado en el entorno local, y se darán prisa en terminar. Como un entorno de desarrollo local normalmente carece de integración de contenido actual con el contenido nuevo, las pruebas no son siempre realistas. Únicamente integrando el contenido existente con el nuevo, puede detectar las posibles incompatibilidades. Una carga directa en el sistema de producción desde la fase de desarrollo, sin la fase de prototipo, puede dar lugar a cualquiera de los siguientes errores:

- El nuevo contenido puede invalidar los archivos en el sistema de producción. Esto ocurre cuando tenemos archivos de imágenes, debido a la falta de una convención estándar a la hora de nombrar los archivos o debido a la utilización de directorios comunes para archivos de imagen.
- Los archivos de datos en el sistema de producción pueden ser invalidados, porque los desarrolladores de CGI utilicen los archivos antiguos de datos cuando desarrollan el contenido.
- Cuando hay implicados varios desarrolladores, algunos de los antiguos archivos pueden reaparecer en el servidor de producción, porque cada desarrollador podría empezar a trabajar con una copia en distintos momentos. Un desarrollador descarga su copia y luego lo hace el otro con el consiguiente error en el resultado.

Pueden aparecer muchos otros problemas si hay varios desarrolladores implicados y sus proyectos están interconectados. Si no quiere correr riesgos en la fase de producción, necesita pasar por la fase prototipo. Apache puede ayudarle a implementar estas fases.

Poner en marcha el ciclo Web

Está preparado para poner en marcha el ciclo Web. De forma ideal, no deberíamos realizar ningún trabajo de desarrollo en el sistema del servidor de producción. Si su presupuesto no permite desplegar gran cantidad de máquinas para su Web, lo que si podría hacer es utilizar su servidor para implementar el ciclo.

Lo primero que necesita es preparar el servidor o los servidores para el ciclo Web. Aunque hay varios modos de hacerlo, vamos a discutir sólo tres. A continuación puede encontrar una breve descripción de cada uno de ellos.

- **Un solo ordenador con dos hosts virtuales para desarrollo y prototipo.** El servidor de producción es el servidor principal de Apache. Tenga cuidado cuando modifique cualquier configuración de Apache en este sistema, ya que los cambios podrían afectar al comportamiento de los servidores de producción.
- **Un solo ordenador con tres servidores principales Apache para desarrollo, prototipo y producción.** Con este método, puede crear configuraciones separadas para cada servidor principal de Apache de modo que puede experimentar con las configuraciones Apache en el sitio de desarrollo sin perturbar la configuración de producción.
- **Al menos tres ordenadores distintos como servidores Apache de desarrollo, prototipo y producción.** Todos estos ordenadores ejecutan servidores Apache en el puerto 80.

Establecer el ciclo Web

Puede establecer el ciclo Web de dos modos: puede utilizar dos host virtuales nuevos para implementar los sitios de desarrollo y de prototipo en su servidor de producción, o puede crear tres configuraciones Apache separadas para sus servidores de producción, de desarrollo y prototipo.

Si su trabajo de desarrollo incluye tocar los archivos de configuración Apache o probar un servidor Apache nuevo, debe utilizar archivos de configuración separados para el servidor de producción Apache y los otros dos servidores Apache. Si su desarrollo normal de Web no incluye cambios relacionados con Apache puede utilizar la aproximación de host virtual.

Un buen ciclo Web requiere una estructura de directorios perfectamente planeada. La figura 11.2 muestra esta estructura de directorios para un ciclo Web.

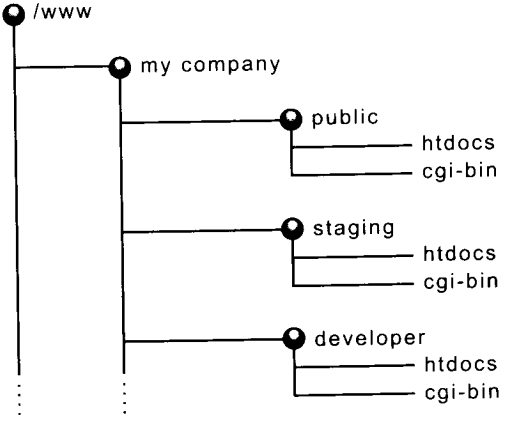


Figura 11.2. La estructura de directorios utilizada para los sitios public, stage y developer para "my company"

Esta figura muestra un buen ejemplo de una estructura de directorios porque le permite mantener los sitios público (public), de prototipo (staging) y de desarrollo (developer) bajo un solo directorio de máximo nivel (en este caso my company). Añadir un nuevo sitio Web significa crear una estructura de directorios parecida a ésta para él. En los ejemplos de configuración que vamos a discutir en las secciones siguientes, voy a suponer que tiene la estructura de directorios anteriores. También supongo que su host servidor Web se llama `www.mycompany.com`, y que tiene la dirección IP `206.171.50.50`. Asegúrese de que reemplaza estos valores con los adecuados para su propia configuración.

Crear un host virtual para cada fase

Si ha decidido modificar los archivos de configuración de Apache como parte de su proceso de desarrollo, no utilice este esquema; necesita una sola asignación de archivos de configuración de Apache en este sistema, y cambiar los archivos para experimentación puede afectar a su servidor de producción. En ese caso, puede seguir utilizando una sola máquina, pero necesita ejecutar varios servidores Apache (principales). Esta aproximación se describe en la siguiente sección.

Si decide que no necesita realizar cambios relacionados con Apache en sus archivos de configuración, puede utilizar este esquema para crear un host virtual para cada fase. Para hacerlo, debería crear dos host virtuales que tengan el mismo nombre de servidor pero que se ejecuten en direcciones de puerto diferentes. La tabla 11.1 muestra una asignación de puerto para cada sistema.

Tabla 11.1. Asignaciones de puertos en servidores Apache para un ciclo Web

Puerto	Tipo de servidor
80	Servidor de producción (servidor principal).
1080	Servidor de prototipo (host virtual).
8080	Servidor de desarrollo (host virtual).

Puede elegir cualquier otra asignación de puertos, siempre que no utilice una dirección de puerto que ya se esté utilizando o que sea superior a 65535. El servidor de producción debería tener asignado el puerto 80, porque las solicitudes HTTP se envían por defecto a esta dirección de puerto. Para crear los host virtuales para la asignación de puerto mostrada en la tabla 11.1, necesita editar el archivo `httpd.conf` del servidor Apache del siguiente modo.

1. Para hacer que el servidor Apache escuche estos puertos, utilice la directiva `Listen`:

```
Listen 80
Listen 1080
Listen 8080
```

2. Entonces, puede crear dos host virtuales del siguiente modo:

```
# No olvide cambiar la dirección IP, los valores de las
directivas
# ServerName,DocumentRoot, ScriptAlias,
# TransferLog y ErrorLog con los valores apropiados en su
# caso, para su propio sistema de configuración.
#
<VirtualHost 206.171.50.50:1080>
    ServerName  www.mycompany.com
    DocumentRoot  "/www/mycompany/staging/htdocs"
    ScriptAlias   /cgi-bin/ "/www/mycompany/staging/cgi-bin/"
    TransferLog  logs/staging-server.access.log
    ErrorLog     logs/staging-server.error.log
    </VirtualHost>

    <VirtualHost 206.171.50.50:8080>
    ServerName  www.mycompany.com
    DocumentRoot  "/www/mycompany/developer/htdocs"
    ScriptAlias   /cgi-bin/ "/www/mycompany/developer/cgi-
bin/"
    TransferLog   logs/developer-server.access.log
    ErrorLog      logs/developer-server.error.log
    </VirtualHost>
```

NOTA: En el ejemplo anterior, se utiliza la misma dirección IP en ambos host, pero se especifican puertos distintos en el contenedor <VirtualHost . . .>. La dirección IP es la misma que la del servidor principal, www.mycompany.com. La directiva ServerName también está asignada al nombre del servidor principal. La configuración de su servidor principal será la habitual.

La URL `http://www.mycompany.com:1080` puede utilizarse para acceder al sitio prototipo; para acceder al sitio de desarrollo, esta URL puede utilizar: `http://www.mycompany.com:8080`.

Utilizar varios procesos (principales) del servidor Apache

Podría utilizar más de un proceso de un servidor (principal) si ha decidido experimentar con Apache fuera de la fase de desarrollo. Tiene que crear tres conjuntos de archivos de configuración, con cada conjunto dirigido a un `DocumentRoot` y un `ScriptAlias` distintos. Una vez que lo ha hecho, puede arrancar los tres servidores (principales) Apache desde el directorio principal en el que tenga instalado Apache (ejemplo: `/usr/local/apache`) del siguiente modo:

```
httpd -f conf/httpd.conf
httpd -f conf/staging/httpd.conf
httpd -f conf/developer/httpd.conf
```

Cuando decida compilar una nueva versión de Apache y la ejecute bajo el servidor de desarrollo, puede simplemente cebarlo en el archivo de configuración para el servidor de desarrollo. Por ejemplo, si ha decidido añadir un nuevo módulo y quiere ver el efecto de este módulo en su contenido, puede simplemente ejecutar los servidores de desarrollo y de prototipo utilizando ese ejecutable en lugar del ejecutable de su servidor de producción (`httpd`). Una vez que ha compilado el nuevo ejecutable tendrá que volver a nombrarlo con algo parecido a `httpd-xx80`, para asegurar que no invalida accidentalmente el ejecutable del servidor de producción.

Para implementar el ciclo, siga las siguientes instrucciones.

1. Ha de crear dos subdirectorios en el directorio de configuración de Apache llamados `staging` y `developer`, del siguiente modo:

```
mkdir  /path/to/Apache/server/root/conf/staging
mkdir  /path/to/Apache/server/root/conf/developer
```

NOTA: No olvide reemplazar `/path/to/Apache/server/root/conf` con la ruta actual del directorio de configuración de su servidor.

2. Copie todos los archivos `*.conf` en los subdirectorios `staging` y `developer`, del siguiente modo:

```
cp  /path/to/Apache/server/root/conf/*.conf  /path/to/Apache/
server/root/conf/staging/*
cp  /path/to/Apache/server/root/conf/*.conf  /path/to/Apache/
server/root/conf/developer/*
```

3. Modifique el archivo `httpd.conf` en el subdirectorio `staging` para que escuche el puerto 1080 en lugar del puerto 80, que es el puerto por defecto. Puede utilizar la directiva `Port` o la directiva `Listen` para realizarlo. Del mismo modo, tiene que modificar `httpd.conf` en el subdirectorio `developer` para que la directiva `Port` y la directiva `Listen` tengan asignado el puerto 8080.

4. Modifique los archivos `srm.conf` (o los `httpd.conf`) en los subdirectorios `staging` y `developer` para que dirijan sus directivas `DocumentRoot` y `ScriptAlias` a la ruta adecuada. Los cambios necesarios aquí para la estructura de directorios mostrada en la figura 11.2 son:

```
DocumentRoot    "/www/mycompany/staging/htdocs"
ScriptAlias     /cgi-bin/   "/www/mycompany/staging /cgi-bin/"
```

para la configuración del sitio prototipo. Por otro lado, para el archivo de configuración del sitio de desarrollo, serán:

```
DocumentRoot    "/www/mycompany/developer/htdocs"
ScriptAlias     /cgi-bin/    "/www/mycompany/developer/cgi-bin/"
```

NOTA: Si utiliza una configuración especial para su servidor de producción, el cual utiliza información de la ruta absoluta, tendrá que editar los nuevos archivos de configuración, para los subdirectorios `staging` y `developer`.

Utilizar varios ordenadores servidores Apache para el ciclo Web

Si puede tener varios ordenadores servidores de Apache (es decir, uno para desarrollo, uno para prototipo y uno para producción) para crear el entorno del ciclo Web, no necesita crear configuraciones Apache especiales. Simplemente instale Apache en todos los host implicados, y trate un host como el host del sitio de desarrollo, un segundo host como el host del sitio de prototipo y el tercer host como el sitio de producción. Como ahora tiene servidores Apache ejecutándose en tres host distintos, puede ejecutar también cada servidor en el puerto 80. Eso es todo lo que necesita para un ciclo con varios host.

Implementar el ciclo Web

Para iniciar su ciclo Web, copie el contenido de producción de su directorio raíz de documentos del servidor de producción en su sitio de desarrollo. Por ejemplo, si su configuración es una de las dos primeras de la lista anterior, puede copiar fácilmente todo su contenido Web en el sitio de desarrollo utilizando los siguientes comandos Unix:

```
cd   /path/to/production/docroot/dir
tar cvf  -   .   | (cd /path/to/development/site/docroot/dir ;
tar xvf - )
```

Estos comandos copian todos los archivos y directorios de su servidor de producción en la raíz de documentos de su servidor de desarrollo. Simplemente tiene que asegurarse de cambiar la información de la ruta donde sea necesario en su sistema.

Para un entorno con varios ordenadores para el ciclo Web, puede crear un archivo `tar` en su servidor de producción y copiarlo en su sitio de desarrollo vía FTP.

Asigne los permisos de archivos de modo que Apache pueda leer todos los archivos y ejecutar scripts CGI. Si tiene directorios en los que Apache podría tener acceso de escritura (para scripts CGI que escriban datos), debería asignar también estos permisos. Una vez que lo ha hecho, inicie o reinicie Apache para servir el sitio de desarrollo.

Ahora, tiene que asegurarse de que aparece el sitio de desarrollo (en el navegador Web) igual que el sitio de producción. Realice algunas comparaciones manuales y algunas pruebas. Asegúrese de que los script también funcionan.

Si cualquiera de sus scripts CGI produce las URL codificadas por hardware para su servidor de producción, seguirán haciendo lo mismo para su sitio de desarrollo. Puede ignorar estas URL o dejarlas fijas de modo que utilicen la variable de entorno `SERVER_NAME` y la dirección de puerto de `SERVER_PORT`.

Probar el ciclo Web

Cuando todo funciona como debe funcionar, significa que ha creado con éxito un entorno de ciclo Web. Ahora puede pedirle a sus desarrolladores que pongan nuevo contenido y scripts en el sitio de desarrollo y que lo prueben. Cada vez que se completa un desarrollo de contenido, debe probarlo primero en el área de desarrollo. Las pruebas suelen enfocarse en los siguientes eventos:

- ¿Cumple su propósito? En otras palabras, ¿la funcionalidad que proporciona el nuevo contenido cumple sus especificaciones?
- ¿Tiene efectos secundarios el nuevo contenido? Por ejemplo, si el contenido nuevo es realmente un nuevo script CGI, debería utilizar el soporte de depuración de errores de Apache para controlar cómo funciona el script.

Mover el sitio nuevo al servidor de producción

Una vez que está satisfecho con los resultados de las pruebas, evite tener que realizar otro conjunto de pruebas de funcionalidad suspendiendo cualquier desarrollo más allá de su nuevo contenido. Cuando llegue el momento de actualizar un sitio de producción, realice una copia de su sitio de producción y colóquela en el sitio prototipo. A continuación tenemos algunos trucos para hacerlo:

- Asegúrese de que el sitio prototipo es exactamente el mismo que el sitio de producción. Una vez que ha realizado algunas comprobaciones manuales para asegurarse de que todo tiene la misma presentación y el mismo enfoque, puede mover contenido y scripts nuevos al sitio prototipo e integrarlos.
- Mueva un proyecto cada vez, de modo que pueda encontrar y resolver los problemas de prototipo. Por ejemplo, si añade tres scripts CGI nuevos a su sistema, mueva un script cada vez al área prototipo. Realice tanto las pruebas de funcionalidad como las de integración del sitio. Si el script supera la prueba, mueva el siguiente script al área prototipo. Una vez que ha movido todo el contenido nuevo, puede realizar las pruebas de nivel de integración del sitio. Monitorice con cuidado los registros del sitio de prototipo. ¿Nota algo raro en los registros de error? Si no es así, entonces está preparado para realizar una actualización de su sitio de producción. Ha de tener cuidado al llevar esto a cabo. Por ejemplo, si tiene algún script CGI

en el servidor de producción que crea archivos de datos en el área de producción, no debería invalidar ninguno de estos archivos de datos con lo que tiene en el área prototipo.

- El mejor momento para actualizar su sitio de producción es cuando piensa que el servidor de producción está menos ocupado. En este momento, puede coger los archivos de datos de su servidor de producción y colocarlos en los directorios apropiados en la versión prototipo del sitio. Todo esto deja a su sitio prototipo en sincronía con el sitio de producción. En este momento, tiene que descargar rápidamente su sitio prototipo dentro del área de producción. Esto podría ser muy delicado porque el sitio de producción está vivo, y nunca va a saber cuándo podría acceder un visitante a una página o cuándo va a utilizar un script CGI que necesita para leer o escribir archivos de datos.

Para minimizar el tiempo (al menos en un sistema de un solo servidor) puede crear un script shell que haga lo siguiente:

1. Copie todos los archivos de datos en las áreas apropiadas de su sitio prototipo.
2. Vuelva a nombrar el directorio de producción de máximo nivel (como el directorio `public` de la figura 11.2) a algo del tipo `public.old`.
3. Vuelva a nombrar su directorio prototipo de máximo nivel (como el directorio `staging` de la figura 11.2) como acostumbre llamar a su directorio de producción de máximo nivel, por ejemplo, `public`.
4. Vuelva a nombrar el viejo directorio de producción (como `public.old`) como acostumbre llamar a su directorio prototipo de máximo nivel, por ejemplo, `staging`.

De este modo, el sitio prototipo se convierte en el sitio de producción en sólo unos cuantos pasos, sin necesidad de realizar un gran número de operaciones de copias de archivos. En la lista 11.1 tenemos un ejemplo de un script que corresponde al entorno mostrado en la figura 11.2.

Listado 11.1. script stage2production.sh

```
#!/bin/sh
# Propósito: un simple script shell para copiar archivos de datos
# en el área prototipo y para volver a nombrar el área prototipo
# en un sitio vivo de producción. También vuelve a nombrar el
# área antigua de producción en el área prototipo.
#
# Copyright © 2001 Mohammed J. Kabir
# License: GNU Public License

# Necesitará cambiar estas variables para utilizar este script.
```

```
DATA_FILES="/www/mycompany/public/htdocs/cgi-data/*.dat";
TEMP_DIR="/www/mycompany/public.old";
PRODUCTION_DIR="/www/mycompany/public";
STAGE_DIR="/www/mycompany/staging";

# Copie los datos en el directorio staging.
/bin/cp $DATA_FILES $STAGE_DIR

# Nombre de forma temporal el directorio de producción actual
TEMP_DIR
/bin/mv PRODUCTION_DIR    TEMP_DIR

# Nombre el sitio prototipo actual como el directorio de
# producción
/bin/mv STAGE_DIR    PRODUCTION_DIR

# Nombre de forma temporal el directorio (antiguo) de producción
# como el directorio staging
/bin/mv TEMP_DIR   STAGE_SITE

# Para estar seguros, cambie los permisos asignados del directorio
# de producción para que el usuario Apache (httpd)
# y el grupo Apache (httpd) puedan leer todos los archivos.
# Si utiliza algún otro usuario y grupo para Apache, tiene
# que modificar este comando de acuerdo con su sistema.

/bin/chown -R httpd.httpd   $PRODUCTION_DIR

# Cambie el permiso de archivo de modo que el dueño
# (httpd en este caso) tenga permiso de lectura, escritura y de
# ejecución, el grupo (httpd en este caso)
# tenga permiso de lectura y de ejecución, y el resto tenga
# permiso para ver los archivos de
# producción

/bin/chmod - R 750   $PRODUCTION_DIR
```

Una vez que ejecutamos este script, debería realizar una prueba rápida para asegurarse de que todo está en orden. En caso de problemas, puede volver a nombrar el directorio de producción actual, y cambiar el nombre del directorio prototipo al nombre de su directorio de producción para restablecer su ultimo sitio de producción.

Construir un sitio Web utilizando plantillas y el makepage

Mantener un ciclo Web estricto le proporciona un proceso que se puede repetir para la publicación de grandes sitios Web. Sin embargo, sigue necesitando un

proceso de presentación de contenido que está altamente automatizado y que requiera muy poca participación por nuestra parte.

Existe una gran cantidad de programas en el mercado que nos pueden ayudar en este proceso.

Algunos desarrolladores utilizan Microsoft Front Page para gestionar el desarrollo de contenido; otros utilizan el Dreamweaver; y algunos utilizan otros productos. Algunas personas utilizan métodos más robustos de compañías de desarrollo Web que cuestan cientos o miles de dólares. A continuación tenemos una solución que me ha funcionado (y lo hará durante años) en el mantenimiento de sitios Web. Los requisitos para esta solución son:

- Crear un mecanismo sencillo para que los autores de contenido puedan publicar páginas Web con una presentación y un objetivo consistentes.
- Requiere una cantidad de trabajo mínima para el autor de contenido de modo que la mayor parte del trabajo está automatizado.
- Supone que el desarrollador de contenido sabe muy poco HTML y prefiere enviar el contenido en formato de texto.

Para implementar esta solución, he escrito el `makepage`, un script que está incluido en el CD-ROM de este libro. Este script utiliza un conjunto de plantillas HTML y una página con el texto del cuerpo (contenido) para construir cada página en el sitio Web. Cuando empecé este proyecto, quería generar cada página al vuelo utilizando CGI o `mod_perl`, pero más tarde decidí generar el contenido una vez al día porque mis sitios iban a ser actualizados únicamente una o dos veces al día. Sin embargo, es muy sencillo aumentar la frecuencia de actualización, como aprenderá en esta sección. El script `makepage` supone que cada página consiste en:

- Una barra de navegación a la izquierda.
- Una barra de navegación a la derecha.
- Un menú de navegación central.
- Un área que aloja el contenido de la página.

Cada vez que se ejecuta el script `makepage` en un directorio determinado, busca todos los archivos terminados con la extensión `.txt` y crea las páginas `.html` correspondientes. Por ejemplo, si ejecuta el script `makepage` en un directorio con un archivo de texto llamado `index.txt`, el script se ejecutará y producirá una salida parecida a esta:

```
Processing ./index.txt
RSB template ./.-rsb.html chosen: /home/mjkabir/www/default-
rsb.html
LSB template ./.-lsb.html chosen: /home/mjkabir/www/default-
lsb.html
```

```
Backed up ./index.html as ./index.html.bak
        Template: /home/mjkabir/www/default-tmpl.html
       BODY file: ./index.txt
         LSB file: /home/mjkabir/www/default-lsb.html
         RSB file: /home/mjkabir/www/default-rsb.html
Bottom Nav file: /home/mjkabir/www/default-bottom.html
    Top Nav file: /home/mjkabir/www/default-top.html
       HTML file: ./index.html
```

Esta salida del script muestra que se está procesando el archivo `index.txt` en el directorio actual (indicado por el punto). Entonces muestra cuál es el archivo de la plantilla de la barra de navegación de la derecha, RSB (Right Side Navigation Bar), que se está utilizando para `index.html`. El script busca primero el archivo de plantilla RSB llamado `index-rsb.html` y si no encuentra un archivo RSB específico para el archivo de texto, utiliza el archivo RSB por defecto para el directorio completo, que va a ser el `default-rsb.html`. Repite el mismo proceso para la plantilla de la barra de navegación de la izquierda, LSB (Left Side Navigation Bar), seleccionada.

Entonces realiza una copia del actual `index.html` (la salida de la última ejecución) a `index.html.bak` y utiliza la plantilla con el cuerpo por defecto `default-tmpl.html` para crear la página `index.html`. Si encuentra una plantilla de cuerpo llamada `index-tmpl.html`, utiliza esta plantilla en lugar de la plantilla de cuerpo por defecto del directorio.

Esto le ofrece la flexibilidad en el diseño de cada página Web. Puede crear simplemente un gran directorio de plantillas y tener todas las páginas con el mismo aspecto.

O puede personalizar una sola página en el directorio con sus propias plantillas RSB, LSB y BODY.

Si ejecuta el script en el directorio raíz de documentos utilizando el comando `makepage path_to_document_root`, el script crea páginas automáticamente en todos los subdirectorios en la raíz de documentos. De este modo, puede establecer este script como un trabajo del `cron` que se ha de ejecutar cada hora, o a diario, o a la semana, o incluso cada minuto según sus necesidades de actualización.

Los autores de contenido simplemente lanzan sus archivos de texto y las páginas se crean automáticamente.

Cuando se lanza un nuevo archivo de texto y no se proporciona la plantilla RSB, LSB o BODY específica para la página, ésta se crea con la plantilla por defecto del directorio, lo que facilita enormemente el que se añadan nuevas páginas. Simplemente escriba una página en su editor de texto favorito y FTP el archivo en su directorio correcto de su sitio Web y éste será publicado en el siguiente `makepage` ejecutado vía `cron`.

El paquete `makepage` suministrado en el CD-ROM incluye las plantillas por defecto que puede estudiar para construir las suyas propias.

Utilizar HTTP PUT para publicaciones Web en una Intranet

Apache soporta el método PUT, que le permite publicar una página Web. Sin embargo, esta característica tiene asociados grandes riesgos de seguridad en el caso de que no se implemente con cuidado extremo. Es evidente que no desea que nadie pueda cambiar su sitio Web. Únicamente recomiendo utilizar esta característica para intranets que no son accesibles desde Internet.

Necesita el módulo `mod_put`, que implementa los métodos PUT y DELETE encontrados en HTTP 1.1. El método PUT le permite cargar contenidos en el servidor y el método DELETE le permite eliminar recursos del servidor. Puede bajar este módulo de `http://hpwww.ec-lyon.fr/~vincent/apache/mod_put.html`.

Las directivas del módulo mod_put

El módulo `mod_put` proporciona tres directivas para controlar la publicación basada en PUT y en DELETE.

EnablePut

`EnablePut` activa o desactiva el método PUT. Para utilizar el método PUT, debe activarlo asignándole el valor `On` a esta directiva.

Sintaxis: `EnablePut On|Off`

Predefinido: `EnablePut Off`

Contexto: directorio, localización

EnableDelete

`EnableDelete On | Off` activa o desactiva el método DELETE, que le permite eliminar una página Web vía HTTP. Para utilizar el método DELETE, debe activarlo asignando el valor `On` a esta directiva.

Sintaxis: `EnableDelete On | Off`

Predefinido: `EnableDelete Off`

Contexto: directorio, localización

umask

`umask octal_` determina la máscara de permiso por defecto (es decir, `umask`) para un directorio. El valor por defecto de `007` asegura que cada archivo dentro del directorio se crea con permiso 770, que únicamente permite al dueño del archivo y al grupo leer, escribir y ejecutar el archivo.

Sintaxis: `umask octal_value`

Predefinido: `umask octal_007`

Contexto: directorio, localización

Compilar e instalar mod_put

Una vez que ha bajado `mod_put`, tiene que realizar los siguientes pasos para compilarlo e instalarlo:

1. Extraiga la fuente `mod_put.tar.gz` y mueva el directorio que ha creado dentro del subdirectorio de módulos de su distribución fuente de Apache.
2. Añada el módulo `mod_put` a Apache utilizando el script `configure` (o `config.status` si ya lo tiene compilado en Apache). Ejecute el script con la opción `add --enable-module=put`.
3. Compile e instale Apache utilizando el comando `make && make install`.
4. Reinicie Apache utilizando el comando `/usr/local/apache/bin/apachectl`.

Establecer un directorio Web que permita el método PUT

Los clientes Web como Netscape, AOLPress y Amaya, pueden publicar páginas Web mediante el método PUT. Esta sección le enseña cómo establecer `httpd.conf` para permitir publicaciones basadas en PUT para un solo directorio Web bajo su árbol raíz de documentos.

ADVERTENCIA: Tenga cuidado con el método PUT si tiene pensado utilizarlo fuera de su intranet. Utilizar PUT en un sitio Web accesible al resto del mundo en Internet podría aumentar enormemente los riesgos de seguridad porque alguien puede desfigurar su sitio Web si el proceso de autentificación basado en la Web que se ha descrito aquí, se ve comprometido. Sólo recomiendo utilizar el método PUT para uso interno.

1. Tiene que crear la configuración siguiente en `httpd.conf`:

```
Alias location_alias
"physical_directory_under_document_root"

<Location location_alias>
```

```
    EnablePut On
    AuthType Basic
    AuthName "Name_of_the_Web_section"
    AuthUserFile path_to_user_password_file
    <Limit PUT>
       require valid-user
    </Limit>

</Location>
```

A continuación vamos a describir lo que está ocurriendo en el código anterior (para aprender más sobre estas directivas relacionadas con la autentificación, lea el capítulo 7):

- Un alias llamado `loc_alias` se asocia con una ruta física llamada `physical_directory_under_document_root`.
- El directorio `<Location>` asigna directivas para este alias.
- La directiva `EnablePut` activa el módulo `mod_put`.
- La directiva `AuthType` asigna el tipo de autentificación en autentificación HTTP `Basic`.
- La directiva `AuthName` asigna una etiqueta para esta sección. Esta etiqueta se despliega en la caja de diálogo de autentificación que los navegadores Web muestran al usuario, por lo que debe asegurarse de que es significativa.
- `AuthUserFile` determina el archivo de contraseña de usuario que utiliza para autentificar al usuario.
- El contenedor `<Limit>` fija límites para el método PUT. Le dice a Apache que necesita usuarios válidos cuando un cliente Web envía una solicitud PUT.

A continuación se muestra un ejemplo de la configuración definida anteriormente:

```
Alias /publish/ "/www/mysite/htdocs/publish/"

<Location /publish>
    EnablePut On
    AuthType Basic
    AuthName "Web Publishing Section"
    AuthUserFile /www/mysite/secrets/.users

    <Limit PUT>
       require valid-user
    </Limit>

</Location>
```

En este ejemplo, el directorio físico `/www/mysite/htdocs/publish` tiene activado un método PUT para todos los usuarios del archivo `/www/mysite/secrets/.users`.

2. Reinicie el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl restart` y utilice su navegador Web con soporte PUT para publicar un documento en el directorio `http://your_web_server/loc_alias`. Para la configuración del ejemplo, esta URL es `http://server/publish`.

   Cuando se publica un archivo utilizando el método PUT, tendrá el permiso asignado utilizando la directiva `umask` para el módulo `mod_put`. El archivo pertenecerá al usuario bajo el que se está ejecutando Apache. Por ejemplo, si asigna las directivas `User` y `Group` en `httpd.conf` para que sean `httpd`, entonces el archivo es propiedad del usuario `httpd`, y el propietario del grupo es también dueño de `httpd`.

Establecer un host virtual para utilizar el módulo mod_put

El usuario determina en la directiva `User` en `httpd.conf` archivos propios creados por `mod_put`. Esto supone un problema en los sitios con varios usuarios distintos porque ahora todo el mundo puede invalidar los archivos de otra persona utilizando el método PUT. Puede resolver este problema fácilmente utilizando un host virtual para cada usuario, tal y como se muestra a continuación.

1. Añada las líneas siguientes a `httpd.conf`:

   ```
   ChildPerUserID number_of_chid_servers username1 groupname1
   ```

   en el que el par `username1 groupname1` son el usuario y el grupo que hay que utilizar para un host virtual. Cambie estos nombres con los nombres de usuario y de host que está utilizando. Ha de crear tantas líneas `ChildPerUserID` como necesite. `num_of_chid_servers` es un número que utiliza Apache para lanzar procesos hijo asociados con este host virtual. Por ejemplo, si tiene dos usuarios llamados carol y john y quiere asignar 10 hijos Apache por cada host virtual, entonces añada las siguientes líneas en `httpd.conf`:

   ```
   ChildPerUserID 10 carol carol_group
   ChildPerUserID 10 john john_group
   ```

   Asegúrese de que los usuarios y grupos existen realmente en `/etc/passwd` y en `/etc/group`, respectivamente.

2. Ha de crear un `VirtualHost` para cada usuario que necesite publicación PUT del siguiente modo:

```
NameVirtualHost  IP_Address

<VirtualHost  IP_Address>

    ServerName vhost_domain_name
    AssignUserID user_name group_name

    Alias location_alias
"physical_directory_under_document_root"

    <Location location_alias>
       EnablePut On
       AuthType Basic
        AuthName "Name_of_the_Web_section"
        AuthUserFile path_to_user_password_file

       <Limit PUT>
            require username
       </Limit>

    </Location>

    # Otras directivas

</VirtualHost>
Example:
NameVirtualHost  192.168.1.100

<VirtualHost  192.168.1.100>
    ServerName carol.domain.com
    AssignUserID carol carol_group
    DocumentRoot /www/intranet/htdocs/carol

    Alias /publish/ "/www/intranet/htdocs/carol/publish/"

    <Location /publish>
       EnablePut On
       AuthType Basic
       AuthName "Carol's Publishing Site"
        AuthUserFile /www/intranet/secrets/.users

       <Limit PUT>
          require carol
       </Limit>

    </Location>

</VirutalHost>
```

Se puede publicar el usuario carol en el directorio `http://carol.domain.com/publish` utilizando su propia cuenta de usuario. También podrá acceder a los archivos creados por el servidor vía FTP, ya que los archivos son propiedad del usuario carol.

3. Una vez que ha creado un host virtual para cada usuario, reinicie Apache utilizando el comando `/usr/local/apache/bin/apachectl restart` y probando el sistema de cada usuario publicando una página de prueba utilizando la URL apropiada.

Mantenimiento de su sitio Web

Una vez que ha implementado el ciclo Web y que tiene un proceso localizado de generación de contenido, es importante mantener su Web. Las tareas habituales de mantenimiento incluyen la monitorización del servidor, y el registro y la copia de seguridad de los datos. Los aspectos de monitorización y registro en el servidor se discuten en el tema 8. Esta sección trata la realización de copias de seguridad de los datos. Debería tener, si es posible, dos tipos de backup, backup online y backup offline.

Backup online

El backup online es útil en caso de emergencia. Puede acceder rápidamente a los datos copiados y, en la mayor parte de los casos, realizar en pocos minutos las tareas necesarias de restitución. Para obtener una solución backup online, buscar un fabricante de backup online comercial o hablar con su ISP. Si está alojando su servidor o servidores Web en su propia red, sin embargo, puede mantener backups de otro host en su red. En la mayoría de los sistemas Unix, puede ejecutar un programa llamado `rdist` para crear directorios mirror en sus sitios Web en otros host Unix (en el capítulo 23 tiene un ejemplo de una aplicación de mirrow de un sitio basado en `rdist`).

Podría ser una buena idea mantener una versión comprimida de los datos Web en el servidor en sí. En los sistema Unix, puede establecer un trabajo `cron` para crear un archivo `tar` comprimido con los datos de la Web con una frecuencia determinada. Por ejemplo:

```
# Para sistemas V-ish Unix, rango laborable 0?6 donde 0=Domingo
# Para sistemas BSD-ish utiliza rango laborable 1?7 donde
1=Lunes
# Es un ejemplo es para un sistema Linux (System V-ish cornd)
30 2 * * 0,1, 3, 5,  root  /bin/tar czf /backup/M-W-F-Sun.tgz
/www/*
30 2 * * 2, 4, 6     root  /bin/tar czf /backup/T-TH-Sat.tgz  /
www/*
```

Si estas dos entradas de `cron` se guardan en `/etc/crontab`, entonces se crean dos archivos. Cada lunes, miércoles, viernes y domingo, el primer trabajo del `cron` se ejecutará a las 2:30 a.m. para crear un backup de todo lo que hay en `/www`, y almacenará el archivo backup comprimido en el archivo `/backup/M-`

W-F-Sun.tgz. Del mismo modo, los martes, los jueves y los sábados por la mañana (a las 2:30 a.m.), la segunda entrada del `cron` creará un archivo con los mismos datos llamado `T-TH-Sat.tgz` en el mismo directorio backup. Tener dos backups le garantiza que tiene al menos dos backups de dos días en dos archivos comprimidos.

Backup offline

También puede realizar backups en un medio transportable y guardarlas en localizaciones seguras. Este tipo de backup suele ser una operación que consume bastante tiempo. Para realizar este backup puede utilizar unidades de cintas, drives duros transportables (como los discos Jaz). Yo prefiero un backup basado en cintas de 8mm porque proporciona 8GB de capacidad de almacenaje de datos; además, llevan en el mercado mucho más tiempo que el nuevo medio transportable compacto.

A medida que sus sitios Web se enriquecen de contenido, el espacio Web disponible se llena rápidamente. Esto suele ser una consecuencia de archivos que están sin usar pero que nunca se han eliminado por miedo a que algo (como un enlace) se rompa en algún sitio. Si piensa que esto está sucediendo en su sitio Web, y está en una plataforma Unix, debería considerar ejecutar la utilidad `find` para localizar archivos a los que no se haya accedido durante mucho tiempo. Por ejemplo:

```
find /www -name "*.bak" -type f -atime +10 -exec ls -l {} \;
```

Esto da lugar a una lista con todos los directorios `/www` que terminan con la extensión `.bak` y a los que no se ha accedido desde hace 10 días. Si quiere eliminar estos archivos, puede reemplazar el comando `ls -l` y realizar una búsqueda como esta:

```
find /www -name "*.bak" -type f -atime +10 -exec rm -f {} \;
```

Si le sirve de ayuda, puede crear una entrada `cron` que ejecute este comando cada cierto tiempo.

Definir estándares

Con un ciclo Web en marcha, tiene un entorno que puede acomodar muchos desarrolladores; sin embargo, únicamente creando un ciclo Web no está asegurando alta calidad de producción Web. Una alta calidad Web requiere contenido de alta calidad, y hay una serie de guías que puede seguir en cuanto al desarrollo de contenido. Se trata de definir sus estándares.

Cada sitio Web debería ofrecer contenido único para hacerlo atractivo a potenciales visitantes. Todos los tipos de contenido Web pueden clasificarse en

contenido estático y contenido dinámico. El contenido estático se crea normalmente con archivos HTML, y el contenido dinámico suele ser la salida de un CGI o de otras aplicaciones del lado del servidor o del lado del cliente.

La mayoría de los sitios Web utilizan una mezcla de ambos tipos de contenido para publicar su información; por lo tanto, son necesarios los estándares tanto para el desarrollo de contenido estático como para el desarrollo de contenido dinámico

Política de desarrollo de documentos HTML

Aunque pueda proporcionar contenido estático de muchos modos, como un plain-text o un archivo PDF, la mayor parte de los sitios Web utilizan documentos HTML como almacén principal de información. Para guiar a sus autores de HTML, debería crear una política de desarrollo de HTML. A continuación tenemos una serie de guías que puede adaptar para su organización.

Utilice siempre etiquetas HTML estándar

Los desarrolladores de HTML deberían utilizar siempre el último estándar de HTML. La utilización de HTML dependiente de navegador hará que la página tenga un aspecto formidable en un tipo de navegador y horrible en otro.

Por ejemplo, el siguiente código muestra un esqueleto HTML que satisface el estándar HTML mínimo.

```
<HTML>
<HEAD><TITLE> Título del documento </TITLE> </HEAD>
<BODY>
   Cuerpo del documento
</BODY>
</HTML>
```

Cada uno de sus documentos debería contener al menos estas etiquetas HTML.

Guarde imágenes in-line junto con los documentos

Las imágenes in-line de un documento deberían residir en un subdirectorio del directorio de documentos. Las referencias de la fuente de estas imágenes deben ser relativas, de modo que si se mueve el documento de una localización a otra junto con el directorio de imágenes, la imagen sigue rindiendo exactamente igual que antes.

NOTA: Hay una excepción a esta regla: si alguna de sus imágenes es reutilizable, debería considerar colocarla en un directorio central de imágenes. Un ejemplo de este tipo de casos es una barra de navegación estándar implementada utilizando archivos de imagen. La barra de navegación se

puede reutilizar en varios documentos, de modo que debería guardar estas imágenes en un directorio central en lugar de guardarlas con cada documento. Esto proporciona mejor control y ahorra espacio en el disco.

El siguiente ejemplo le muestra cómo crear un documento HTML transportable que tiene varios archivos de gráficos unidos a él. Imagine que quiere publicar dos documentos HTML (`mydoc1.html` y `mydoc2.html`) que contienen tres imágenes (`image1.gif`, `image2.gif` y `image3.gif`). Puede crear primero un subdirectorio significativo bajo su directorio raíz de documentos o bajo cualquier otro directorio que sea apropiado. Vamos a suponer que crea este directorio bajo el directorio raíz de documentos del servidor (`/www/mycompany/htdocs`) y que lo llama `mydir`.

Ahora, tiene que crear un subdirectorio de imágenes llamado `images` bajo el directorio `mydir` y almacenar sus tres imágenes en este directorio. Edite sus documentos HTML de modo que todos los enlaces a las imágenes utilicen el atributo SRC del siguiente modo:

```
SRC="images/image1.gif"
SRC="images/image2.gif"
SRC="images/image3.gif"
```

Un ejemplo de un enlace a una imagen in-line para `image3` sería:

```
<IMG SRC="images/images3.gif" HEIGHT="20" WIDTH="30" ALT="Image
3 Description">
```

Los atributos SRC de la línea anterior no contienen ninguna información sobre rutas absolutas. Si se moviesen los documentos desde `mydir` a `otherdir` junto con el subdirectorio `images`, no habría ningún problema con las imágenes. Sin embargo, si los enlaces contienen información de rutas del estilo a:

```
<IMG SRC="mydir/images/images3.gif" HEIGHT="20" WIDTH="30"
ALT="Image 3 Description">
```

o

```
<IMG SRC="/mydir/images/images3.gif" HEIGHT="20" WIDTH="30"
ALT="Image 3 Description">
```

tendríamos que determinar estos documentos una vez movidos. Muchos sitios guardan sus imágenes en un directorio central de imágenes (como `images`) bajo la raíz de documentos y enlazan los documentos utilizando etiquetas IMG como:

```
<IMG SRC="/images/images3.gif" HEIGHT="20" WIDTH="30"
ALT="Image 3 Description">
```

Esto resulta perfecto, pero cuando quiere eliminar el documento HTML, necesita asegurarse de que también elimina la imagen apropiada en el directorio cen-

tral de imágenes. Si falla en este proceso, a la larga desaparecerá una gran cantidad de espacio en el disco en su cantera de imágenes. Por lo tanto, no es una buena idea mantener imágenes en un directorio central. Debería guardar imágenes en un subdirectorio con sus enlaces.

Desplegar mensajes copyright en cada documento

Cada documento debería contener un mensaje copyright embebido (comentado) que nombre, con claridad, al dueño del documento y a todas sus imágenes. Debería aparecer también un mensaje parecido de copyright en cada página. Para facilitar la actualización del mensaje de copyright, debería considerar la utilización de una directiva SSI como la siguiente:

```
<!--#include file=/copyright.html" -->
```

Ahora, todo lo que necesita hacer es crear una página HTML llamada `copyright.html`, y colocarla bajo su directorio raíz de documentos. Como el contenido de esta página HTML se inserta en el documento compatible con `SSI` que hace esta llamada, no necesita utilizar las etiquetas `<HTML>`, `<HEAD>`, `<TITLE>` o `<BODY>`. Utilizar las llamadas `SSI` le hará la vida más sencilla cuando necesite actualizar el año en el mensaje copyright, o cuando necesite realizar cualquier otro cambio.

Política de desarrollo de aplicaciones dinámicas

El contenido dinámico se produce normalmente con scripts CGI o con otras aplicaciones que implementen CGI o alguna interfaz del lado del servidor. Una gran mayoría de contenido se produce utilizando scripts CGI basados en Perl. Como los scripts y las aplicaciones CGI tienen normalmente poca vida de expansión, muchos desarrolladores de CGI no son partidarios de producir aplicaciones de alta calidad.

Si tiene pensado utilizar los scripts y las aplicaciones basadas en `FastCGI` o en `mod_perl`, es importante que se desarrollen del modo adecuado. Debería considerar las siguientes políticas cuando implemente scripts y aplicaciones para su contenido dinámico.

Utilice siempre un control de la versión

Los desarrolladores de CGI deben utilizar un control de versión, que les permita volver a una versión o a una aplicación antiguas en caso de que la nueva versión contenga un error.

En la mayoría de los sistemas Unix, puede utilizar el software Concurrent Versions System (CVS) para implementar un entorno de versiones controladas. Puede encontrar la última versión del software CVS en `ftp://prep.ai.mit.edu`.

No utilice nombres de rutas absolutos en los scripts ni en las aplicaciones CGI

No se deben utilizar nombres de rutas en los scripts CGI. Esto asegura que los scripts se pueden utilizar en varios sitios Web sin modificación ninguna. Si se necesitan los nombres de ruta para un propósito determinado, se puede suministrar un archivo de configuración para el script; de este modo, las rutas se pueden actualizar modificando el archivo de configuración.

Proporcionar documentación de usuario y de código

Se necesita código fuente para estar bien documentado de modo que los desarrolladores futuros puedan actualizar los scripts sin perder mucho tiempo tratando de imaginar cómo funciona.

Evitar las etiquetas HTML embebidas en scripts o en aplicaciones

Las salidas de los scripts CGI deberían estar manejados por plantillas. En otras palabras, un script CGI lee una salida de una plantilla y reemplaza los campos de datos dinámicos (que se pueden representar utilizando etiquetas personalizadas). Esto hace sencilla la actualización de las salidas de páginas para los desarrolladores de HTML, ya que el HTML no está dentro del script CGI. De hecho, los scripts CGI deberían contener la menor cantidad de HTML como sea posible.

No confiar en los datos introducidos por el cliente

Para reducir los riesgos de seguridad, compruebe los datos introducidos por el usuario antes de utilizarlos. Puede aprender más sobre la verificación de entradas en el capítulo 18, en el que se discuten los riesgos relacionados con las entradas y las soluciones en detalle.

Evitar las variables globales en los scripts CGI basados en Perl

Cuando desarrollamos scripts CGI en Perl, debería evitar las variables globales. Limitar el alcance de una variable es un modo de eliminar comportamientos impredecibles en los scripts. Los programadores de Perl deberían utilizar el siguiente código para las declaraciones de variables:

```
my $variable;
```

en lugar de :

```
local $variable;
```

porque la primera declaración crea una variable que sólo está disponible en el ámbito en el que se ha creado. La última definición simplemente crea una instancia local de una variable global, lo que supone una gran confusión. Perl 6 proba-

blemente cambiará la palabra clave 'local' por 'temp' para aclarar este concepto a los programadores.

Proporcionar a su sitio Web una interfaz intuitiva

Utilizar HTML estándar y scripts y aplicaciones CGI bien escritas puede garantizar que su sito Web sea mejor que muchos otros sitios Web. Sin embargo, hay otro aspecto del diseño de un sitio Web que tiene que considerar, la interfaz de usuario.

Piense en un sitio Web como si fuera una aplicación interactiva con un Graphical User Interface (GUI) visible en un navegador Web. La GUI necesita ser intuitiva para que la gente tenga una experiencia agradable mientras que están visitando su sitio Web.

En esta sección se discuten los puntos claves en el desarrollo de una GUI intuitiva. Además de hacer intuitiva su GUI, tiene que estar atento a enlaces rotos o a solicitudes a archivos eliminados. Utilice los registros de error de su servidor para detectar este tipo de problemas. Debería facilitar a los visitantes la interacción con su sitio. La mayor parte de los sitios utilizan una interacción sencilla basada en un formulario HTML o con un script CGI. Puede desarrollar cualquiera de estas interacciones. Proporcionar interacción es un buen modo de aprender lo que piensan sus visitantes sobre su sitio Web.

Facilite la navegación en su sitio

Los usuarios deben ser capaces de ir de una página a otra con facilidad. Deberían ser capaces de localizar botones o barras de menú que les permitan moverse atrás y delante, o saltar a información relacionada.

Muchos diseñadores de páginas Web opinan que los navegadores Web populares incluyen botones de atrás y delante, por lo que es una redundancia incluirlos en la página. Esto es incorrecto. Imagine que un usuario encuentra una de sus páginas (distinta de la página de inicio) desde los resultados de un buscador. El usuario simplemente estaba realizando una búsqueda con una o más palabras clave, y el buscador le proporcionó una URL a una página de su sitio. El usuario está muy interesado en conocer algo más sobre su sitio, de modo que quiere empezar desde el principio del documento, pero no hay modo de que el usuario pueda hacerlo, porque el botón de Atrás del navegador le devuelve al buscador y le saca de la página.

Si esta página tuviese un enlace (o un botón) a una página anterior, el usuario podría navegar por su sitio Web sin problemas. Los diseñadores de páginas Web a los que no les gusta los botones extra insisten en que el usuario podría simple-

mente manipular la URL para llegar a la página de inicio y empezar desde allí. Bien, para llevar esto a cabo hay que suponer que existe un enlace claro a esa página (la que coincide con la palabra buscada) desde la página de inicio, lo cual no es siempre cierto.

Es una buena idea implementar una barra de menú que permita al usuario ir hacia atrás y hacia delante, y que también permita al usuario saltar a una localización relacionada, o incluso a la página de inicio.

Crear un diseño atractivo

Piense en los sitios Web como en presentaciones llenas de colorido e interactivas, que están activas las 24 horas del día. Si la presentación no es correcta, sus visitantes abandonarán su sitio. Considere las siguientes guías para el desarrollo de un diseño Web atractivo.

Colores apropiados

Asegúrese de no excederse con la elección de colores. La utilización de colores extremos hace de su sitio Web un sitio poco profesional y poco lustroso. Ha de preocuparse por el colorido y utilizar el esquema apropiado de colores. Por ejemplo, si su sitio Web está relacionado con juguetes para niños, debería ser bastante colorido. Si su sitio está relacionado con precios de procesadores de señales digitales, no va a necesitar colores brillantes ni llamativos.

Tamaño apropiado de texto

Intente que el contenido principal aparezca en una fuente normal. La utilización de una fuente especial mediante `<FONT FACE="mifuenteespecial">` puede hacer que la página tenga una buena presentación en su navegador (porque tiene la fuente), pero en otro navegador, la página puede ser totalmente distinta y podría ser difícil de leer. Además, ha de tener cuidado con el tamaño del texto; no puede ser ni muy pequeño ni muy grande. Recuerde que si sus visitantes no pueden leer lo que tiene que decir en su página Web, no serán capaces de entender lo que quiere decir.

Mínima utilización de imágenes y animaciones

Tenga cuidado con las imágenes innecesarias. Las imágenes en sus páginas Web hacen que se carguen con más dificultad. Recuerde que no todo el mundo está conectado a su sitio Web mediante una ADSL o una RDSI; la mayor parte de la gente sigue utilizando modems de 56K o 28.8K para su conexión a Internet. Una descarga lenta de una página Web puede hacer que un cliente potencial abandone su sitio.

Además, es necesario ser prudente con la utilización de información. Incluso las animaciones más sugerentes se convierten en aburridas después de las primeras visitas, de modo que asegúrese de no super poblar sus páginas con ellas.

Elimine los mensajes de error en clave

Configure Apache con la directiva `ErrorDocument`, de modo que los usuarios no reciban mensajes de error en el servidor que sean difíciles de entender (al menos para el usuario medio). Por ejemplo, cuando no se encuentra una solicitud URL en el servidor, el servidor desplegará un mensaje de error en clave. Para hacer este mensaje de error más intuitivo, puede añadir una directiva `ErrorDocument` como la siguiente:

```
ErrorDocument  404  /sorry.html
```

en el archivo `httpd.conf`, de modo que los visitantes de la Web puedan entender el mensaje de error.

Pruebe su GUI Web

Uno de los mejores modos de probar su interfaz Web es utilizar un sistema que se parezca al ordenador medio de los usuarios de Internet, o quizá al ordenador de un cliente potencial. Si piensa que sus clientes van a tener ordenadores de alto rendimiento con conexiones rápidas, no tiene que preocuparse por la utilización de pocos gráficos o de aplicaciones del lado del cliente como son los applets de Java y las animaciones Shockwave.

En la mayoría de los casos, no conoce las especificaciones del ordenador ni de la red de sus potenciales clientes, de modo que debería centrarse en el sistema del usuario medio. Utilice un ordenador Pentium low-end con 16MB de RAM y una conexión vía modem de 28.8K para probar su sitio Web desde una cuenta ISP. Pruebe con bajas resoluciones para el monitor, como 640x480 o 800x600 píxeles; si sus visitantes objetivo utilizan sistemas Web-TV, pruebe con una resolución de 550x400 píxeles.

Si disfruta navegando por su sitio Web, otros probablemente también disfrutarán. Por otro lado, si no le gusta lo que ve, a otros tampoco les gustará.

TRUCO: Si lo prefiere, puede realizar una prueba de su sitio Web con productos de terceras partes. Por ejemplo, `Netscape.com` proporciona un servicio gratuito de ajuste basado en la Web, que puede encontrar en `http://websitegarage.netscape.com`. Esta aplicación de Netscape puede examinar cualquier sitio Web en relación con el tiempo de descarga de páginas, calidad del HTML, enlaces que terminan en puntos muertos, errores de ortografía, calidad del diseño HTML y la popularidad del enlace. Para ponerlo a prueba, simplemente diríjase al sitio Web anterior e introduzca su propia dirección del sitio Web y su dirección de correo electrónico y espere de unos segundos a unos minutos. Recibirá un diagnóstico gratuito de su sitio Web.

Promocionar su sitio Web

¿Qué tiene de bueno un sitio Web perfecto si nadie lo conoce? Debería pensar en promocionar su sitio Web en la Web. Puede alquilar agencias de publicidad para ayudarle en este asunto, aunque los anuncios en la Web pueden resultar caros. Si su presupuesto no es muy alto puede hacer algo para promocionarse a sí mismo. La siguiente lista le ofrece algunos puntos para promocionar su sitio adecuadamente.

- **Buscadores:** antes de hacer nada para promocionar su sitio Web, pregúntese, "¿cómo encuentro información en la Web?" La respuesta es: a través de los buscadores. ¿Está su compañía en una lista de un buscador? Si la respuesta es no, este es el primer paso en la promoción de su sitio Web.

  Casi todos los buscadores le permiten introducir su URL en la base de datos del robot de búsqueda, de modo que este robot puede atravesar su Web en el futuro. Debería realizar una lista de los buscadores que considere más importantes, e introducir la URL de su sitio Web en estos buscadores. Este proceso puede llevarle días o semanas.

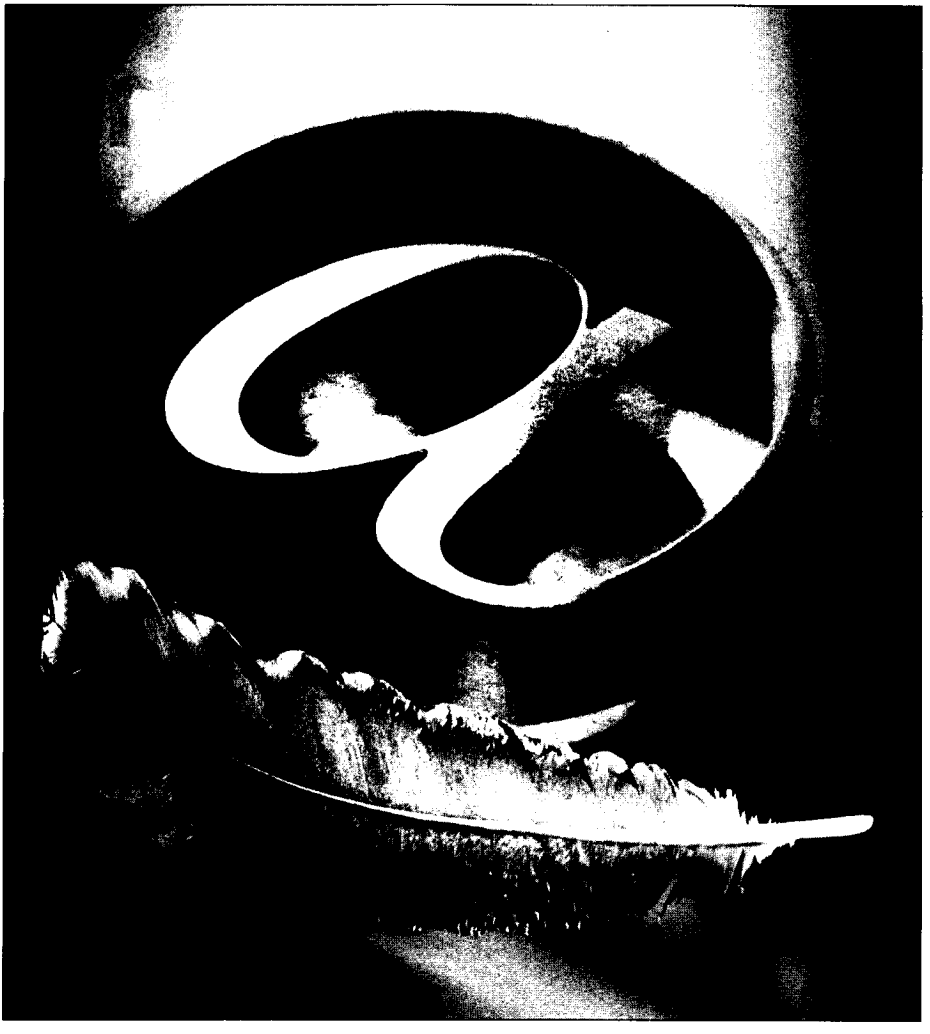
- **Etiquetas META:** puede añadir información META en su contenido para que su URL aparezca en una posición decente cuando un cliente potencial realiza una búsqueda. Por ejemplo, puede añadir información META del tipo:

```
<META NAME="KEYWORD" CONTENT="palabraclave1 palabraclave2
palabraclave3 ...">
<META NAME="DESCRIPTION" CONTENT="Descripción de su
compañía">
```

- **Intercambio de enlaces:** para incrementar el tráfico en su sitio web, puede participar en el intercambio de enlaces en sitios como `www.linkexchange.com`. Los intercambios de enlaces requieren que ponga un conjunto especial de etiquetas HTML en sus páginas Web; estas etiquetas introducen archivos de gráficos de publicidad (banner) en sus páginas Web. Como contrapartida, sus gráficos de publicidad se mostrarán en otros sitios Web. Este tipo de compartición de publicidad es muy popular entre personas y en los negocios de pequeño tamaño.

  Si compra un espacio de publicidad en sitios Web de alto perfil como Yahoo, AltaVista o Netscape, o utiliza el método de intercambio de enlaces, debería comprobar periódicamente que su sitio se encuentra en las salidas de los buscadores generando sus propias peticiones.

Parte III
Ejecutar aplicaciones Web



12 Ejecutar scripts CGI

En este capítulo

1. Entendemos los conceptos básicos del Common Gateway Interface.
2. Configuramos Apache para CGI.
3. Proporcionamos acceso a `cgi-bin` a usuarios individuales.
4. Ejecutamos aplicaciones CGI utilizadas habitualmente.
5. Configuramos Apache para depurar las aplicaciones CGI.

El contenido dinámico dirige la Web. Sin contenido dinámico y personalizable, la Web resultaría un sitio del tipo "se encuentra en tal sitio, haga tal cosa". Después de todo, la gente no va a buscar y a experimentar el mismo contenido caduco una y otra vez. El contenido dinámico se está convirtiendo en una realidad gracias a la ayuda de una especificación llamada Common Gateway Interface (CGI). La especificación CGI le dice al servidor Web cómo interaccionar con aplicaciones externas. Un servidor Web que ejecuta aplicaciones CGI, le permite, prácticamente a cualquiera, ejecutar a demanda una lista seleccionada de programas en su servidor. Este capítulo trata los conceptos básicos del CGI para

ofrecerle una compresión clara sobre el tema, y los detalles la preparación de Apache para soportar ejecuciones CGI.

¿Qué es CGI?

Para proporcionar contenido dinámico e interactivo en la Web, muchos sitios Web populares utilizan aplicaciones CGI. Lo cierto es que usted ha utilizado ya algunas aplicaciones CGI en la Web. Por ejemplo, cuando rellena un formulario Web es muy probable que éste se procese mediante un script CGI escrito en Perl o en algún otro lenguaje.

Por supuesto, a medida que surgen más tecnologías Web, aparecen nuevas formas de distribuir contenido dinámico en la Web. La mayoría de estas soluciones son específicas para un lenguaje determinado, o dependen de un sistema operativo o del desarrollo de un software comercial. CGI, por su parte, es una especificación de una interfaz de puente (gateway), independiente del lenguaje que se puede implementar utilizando de forma cualquier lenguaje de desarrollo de aplicaciones conocido, incluyendo C, C++, Perl, lenguajes de script shell y Java.

Esta sección le ofrece un resumen del funcionamiento de un programa CGI (ver la figura 12.1). La idea principal es que el servidor Web obtiene una URL determinada, que de forma casi mágica, al menos por ahora, le dice al servidor que debe ejecutar una aplicación externa llamada `helloworld.cgi`. El servidor Web lanza la aplicación, espera a que se complete y devuelve un resultado. Entonces, transmite el resultado de la aplicación al cliente Web del otro lado.

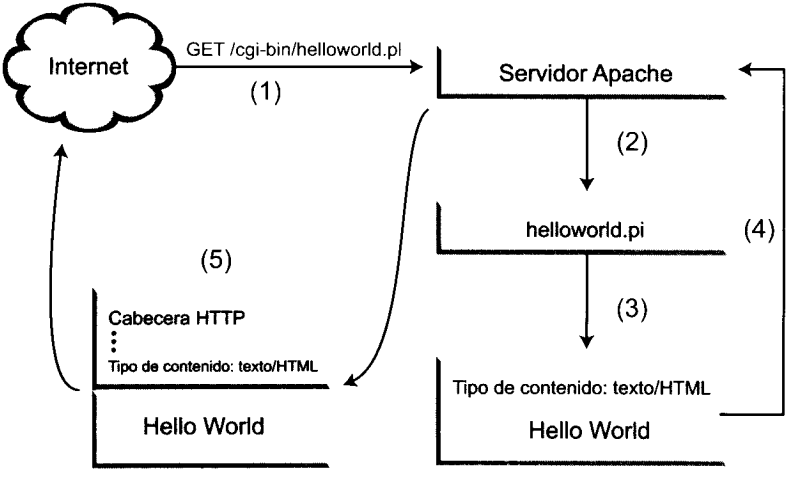


Figura 12.1. Funcionamiento de un programa CGI

En ocasiones vamos a necesitar que el cliente sea capaz de interaccionar con la aplicación. La aplicación debe suministrar los datos introducidos por el cliente. Por otro lado, cuando una aplicación produce una salida, ¿cómo saben el servidor

o el cliente el tipo de resultado que devuelve? Un programa puede producir un mensaje de texto, un formulario HTML para entradas, una imagen, y otro tipo de contenido. Como puede ver, el resultado puede variar mucho de una aplicación a otra, de modo que debe existir una forma para que las aplicaciones informen al servidor Web y al cliente sobre el tipo de resultado.

CGI define un conjunto de significados estándar para que éste le pase al cliente las salidas de las aplicaciones externas, y además define las formas en las que una aplicación externa puede devolver un resultado. Cualquier aplicación que se adhiera a estos estándares, se pueden etiquetar como una aplicación, programa o script CGI. Por simplicidad, utilizo el término programa CGI para referirme a cualquiera de ellos (como un script Perl o un programa C) que sea compatible con una especificación CGI. En la siguiente sección, veremos cómo trabaja un proceso de entrada/salida CGI.

Input y Output CGI

Hay muchas formas en las que un servidor Web puede recibir información de un cliente (un navegador Web, por ejemplo). El protocolo HTTP define el modo en el que un servidor Web y un cliente pueden intercambiar información. Los métodos más comunes para trasmitir solicitudes a un servidor Web son las solicitudes `GET` y las solicitudes `POST`, que se van a describir en las secciones siguientes.

Solicitudes GET

Las solicitudes `GET` son el método más sencillo de enviar solicitudes HTTP. Cada vez que introduce una dirección de un sitio Web en su servidor, genera una solicitud `GET` y la envía al servidor Web solicitado. Por ejemplo, si introduce `http://www.hungryminds.com` en su navegador Web, envía una solicitud HTTP como la siguiente:

```
GET /
```

al servidor Web `www.hungryminds.com`. Esta solicitud `GET` le pide al servidor Web de Hungry Minds que devuelva el documento de máximo nivel del árbol de documentos de la Web. Este documento se llama normalmente página de inicio, y normalmente se refiere a la página `index.html` del directorio Web de máximo nivel. Además, HTTP le permite codificar información adicional en una solicitud `GET`. Por ejemplo:

```
http://www.mycompany.com/cgi-bin/search.cgi?books=cgi&author=kabir
```

Aquí, la solicitud `GET` es:

```
GET www.mycompany.com/cgi-bin/search.cgi?books=cgi&author=kabir
```

Esto le dice al servidor que ejecute el programa CGI `/cgi-bin/search.cgi` y que le pase los datos introducidos, `books=cgi` y `author=kabir`.

Cuando un servidor compatible con CGI como Apache, recibe este tipo de solicitud, sigue las especificaciones CGI y pasa los datos introducidos a la aplicación (en este caso, `search.cgi` en el directorio `cgi-bin`). Cuando un recurso CGI se solicita mediante el método `GET` de solicitudes HTTP, Apache:

1. Asigna las variables de entorno al programa CGI, que incluye almacenaje del nombre del método de solicitudes HTTP en una variable de entorno llamada `REQUEST_METHOD`, y el dato recibido del cliente en una variable de entorno llamada `QUERY_STRING`.
2. Ejecuta el programa CGI solicitado.
3. Espera a que se complete el programa y devuelva una salida.
4. Analiza la salida del programa CGI si se trata de un programa de cabecera no analizada (un programa CGI con la cabecera sin analizar crea sus propias cabeceras HTTP de modo que el servidor no necesita analizar las cabeceras).
5. Crea la cabecera o las cabeceras HTTP necesarias.
6. Envía las cabeceras y el resultado del programa al cliente que las ha solicitado. La figura 12.2 ilustra este proceso.

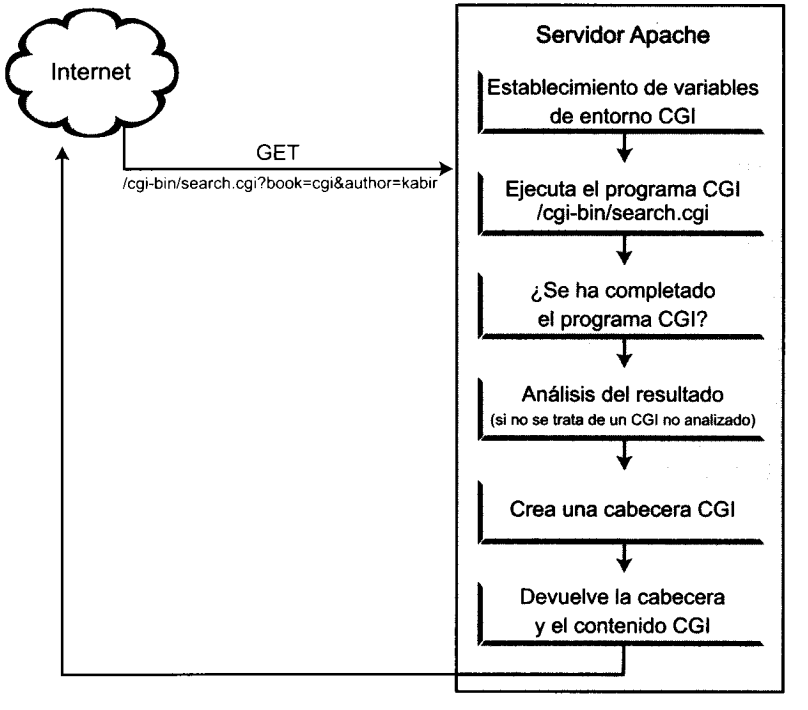


Figura 12.2. Proceso CGI en un servidor

A continuación vamos a ver lo que tiene que hacer un programa CGI para recuperar los datos introducidos para utilizarlos para propósitos internos.

Tal y como muestra la figura 12.3, un programa CGI.

1. Lee la variable de entorno REQUEST_METHOD.
2. Determina si se utiliza o no el método GET utilizando la variable almacenada en la variable REQUEST_METHOD.
3. Si se utiliza el método GET, recupera el dato almacenado en la variable de entorno QUERY_STRING, .
4. Codifica el dato.
5. Procesa el dato codificado.
6. Escribe el tipo de contenido del resultado en su herramienta de salida estándar (STDOUT), tras completar el proceso.
7. Escribe el dato resultante en el STDOUT y sale.

El servidor Web lee el STDOUT de la aplicación y lo analiza para localizar el tipo de contenido de la salida. Entonces transmite las cabeceras HTTP y el Content-Type apropiados antes de transmitir la salida al cliente. El programa CGI se abandona y se completa la transacción CGI.

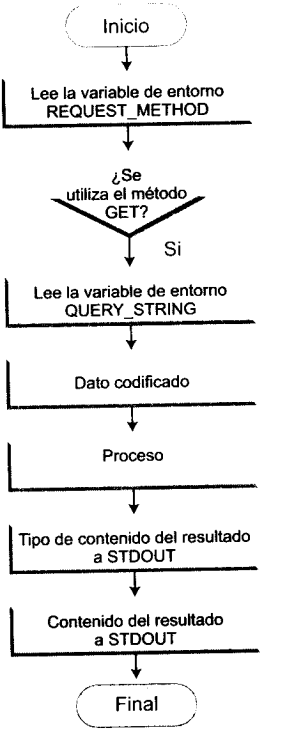


Figura 12.3. Proceso CGI en el servidor

NOTA: Si tenemos un programa CGI para proporcionar toda la información sobre las cabeceras HTTP y el tipo de contenido, hay que ponerle el prefijo nph (para cabeceras las cabeceras no analizadas) a su nombre. El servidor no analiza la salida nph de un programa CGI y la transmite directamente al cliente; la mayoría de los programas CGI dejan que el servidor escriba la cabecera HTTP y, por lo tanto, se trata de programas de cabeceras analizadas.

La utilización del método GET para pasar datos introducidos en un programa CGI, está limitado en muchos sentidos, incluidos los que se exponen a continuación:

- El tamaño total de datos que se pueden transmitir como parte de una URL está limitado por el límite de la longitud de la URL del cliente. Muchos navegadores Web tienen límites de hardware para la longitud de una URL, y por lo tanto, el número total de datos que se puede enviar mediante una URL codificada es muy pequeño. Sin embargo, en ocasiones puede ser una gran idea pasar datos a los programas CGI mediante una URL. Por ejemplo, si tiene un formulario HTML que utiliza el método GET para enviar datos a un programa CGI, esta URL se puede introducir en la carpeta de favoritos para utilizarla más tarde, sin necesidad de volver a introducir los datos de entrada. Esto puede resultar muy cómodo para las aplicaciones de consulta a bases de datos.
- La longitud del valor de una sola variable de entorno (QUERY_STRING) está limitada. Muchos, sino todos, sistemas operativos tienen límites en el número de bytes que puede contener el valor de una variable de entorno. Esto limita el número total de bytes que se pueden almacenar como datos de entrada.

Estos límites no influyen en los programas CGI que necesitan pocas entradas de usuario o que no las necesitan. Para aquellos programas que necesitan una gran cantidad de datos introducidos por el usuario, sin embargo, es mejor utilizar otro método (el método POST) para las solicitudes HTTP. El método POST se discute en la siguiente sección.

Solicitudes POST

El método POST se utiliza para pasar datos a programas CGI. Suele encontrar este método en los formularios HTML que rellena en las páginas Web. Como ejemplo, tenemos el listado 12.1.

Observe que hay una sección <FORM> </FORM> en el listado. Un formulario HTML tiene normalmente una etiqueta de inicio <FORM> que define la acción ACTION y el método METHOD para el formulario. En el ejemplo anterior,

la acción es el programa CGI /cgi-bin/search.cgi y el método es el método POST.

Listado 12.1. Un formulario HTML utilizando el método POST

```
<HTML>
  <HEAD>
        <TITLE> Apache Server 2.0 - Listado capítulo 12 12.1 </
TITLE>
  </HEAD>

<BODY>
<H1>Listing 12-1</H1>
<H2>Un ejemplo de un formulario HTML utilizando el método POST </
H2>
<HR>
<FORM ACTION="/cgi-bin/search.cgi" METHOD="POST">

<PRE>
 Tipo de libro <INPUT TYPE="TEXT" NAME="book" SIZE="10"
MAXSIZE="20">
Nombre del autor <INPUT TYPE="TEXT" NAME="author" SIZE="10"
MAXSIZE="20">
</PRE>
<INPUT TYPE=SUBMIT VALUE="Search Now">

</FORM>

</BODY>
</HTML>
```

Después de la etiqueta <FORM>, normalmente hay una o más entidades INPUT; las entidades INPUT deben incluir cajas de texto, menús desplegables y listados. En nuestro ejemplo, hay tres entidades de introducción de datos. La primera permite al usuario introducir un valor para la variable book. La siguiente es parecida y permite al usuario introducir un valor para la variable author, y la última es algo distinta y permite al usuario enviar el formulario. Cuando el usuario envía el formulario, el software del cliente transmite una solicitud POST al servidor para el recurso ACTION (es decir, /c/s.dll/search.cgi), y además transmite los valores book=<valor introducido por el usuario> y author=<valor introducido por el usuario> en un formato codificado.

Comparar GET y POST

¿Cuál es la diferencia entre las solicitudes GET y POST? Los datos enviados con el método POST no se almacenan en la variable de entorno QUERY_STRING de un programa CGI. Por el contrario, se almacenan en la entrada estándar

(STDIN) del programa CGI. La variable REQUEST_METHOD se asigna a POST, mientras que los datos codificados se almacenan en el STDIN del programa CGI, y se asigna una nueva variable de entorno llamada CONTENT_LENGTH al número de bytes almacenados en el STDIN.

El programa CGI debe comprobar el valor de la variable de entorno REQUEST_METHOD. Si está fijado en POST para las solicitudes HTTP POST, el programa debería determinar primero el tamaño de los datos introducidos con el valor de la variable de entorno CONTENT_LENGTH y, entonces, leer los datos desde el STDIN. Observe que el servidor Web no es responsable de la inserción de un marcador End-of-File (EOF) en el STDIN, que es por lo que la variable CONTENT_LENGTH está asignada a la longitud de los datos, en bytes, haciendo más fácil que el programa CGI determine el número total de bytes en los datos.

Es posible utilizar GET y POST al mismo tiempo. A continuación tenemos un ejemplo de un formulario HTML que utiliza oficialmente el método POST, pero que además utiliza una cadena de consulta, username=joe, como parte del ACTION CGI.

```
<FORM ACTION="/cgi-bin/edit.cgi?username=joe" METHOD=POST>
<INPUT TYPE=TEXT NAME="PhoneNumber">
</FORM>
```

En este ejemplo, la consulta username=joe formará parte de la URL, pero el otro campo (PhoneNumber) formará parte del dato POST. El efecto: el usuario final puede introducir la URL en la carpeta de favoritos y ejecutar el script edit.cgi como joe sin asignar valores para ningún otro campo. Esto resulta perfecto para las aplicaciones de bases de datos online y para los buscadores.

Tanto si utiliza GET, o POST, o ambas, los datos son codificados y devueltos al programa CGI para decodificarlos. La siguiente sección discute los conceptos implicados en la descodificación de los datos.

Decodificación de los datos introducidos

Los diseñadores del protocolo HTTP planearon una implementación sencilla del protocolo en cualquier sistema. Además, realizaron un esquema de descodificación de los datos muy sencillo. Los esquemas definen ciertos caracteres como caracteres especiales. Por ejemplo, el signo igual (=) facilita la realización de pares clave=valor; el signo más (+) reemplaza el carácter espacio, y el carácter ampersand (&) separa dos pares clave=valor.

Si el dato contiene caracteres con un significado, debería preguntarse qué es lo que se está transmitiendo. En este caso, se utiliza un esquema de codificación de tres caracteres. Un signo de porcentaje (%) indica el comienzo de una secuencia codificada de caracteres que consiste en dos dígitos hexadecimales.

El sistema hexadecimal es un sistema de numeración en base 16 en el que los números del 0 al 9 representan los mismos valores que los números del 0 al 9 en

el sistema decimal, pero tiene un conjunto extra de dígitos. Estos dígitos extra son A (=10), B (=11), C (=12), D (=14) y F (=15). Por ejemplo, el 20 en el sistema hexadecimal es igual al 32 en el sistema decimal. El sistema de conversión es:

```
20 = 2 x (16^1 ) + 0 x (16^0)
```

Estos dos dígitos consisten en el valor que puede integrar en la tabla ASCII (para el inglés) para obtener el carácter. Por ejemplo, %20 (hexadecimal) es 32 (decimal) y se corresponde al carácter espacio en la tabla ASCII.

Variables CGI Apache

Hay dos modos en los que Apache puede implementar soporte CGI. La distribución estándar de Apache incluye un módulo CGI que implementa el soporte CGI tradicional; sin embargo, hay un nuevo módulo (`FastCGI`) que implementa soporte para aplicaciones CGI de alto rendimiento. Esta sección discute el soporte estándar CGI.

En las secciones anteriores, vimos que un servidor Web que es compatible con CGI, utiliza variables de entorno, entradas estándar (`STDIN`) y salidas estándar (`STDOUT`) para transferir información a y desde los programas CGI. Apache proporciona un conjunto flexible de variables de entorno para los desarrolladores de programas CGI.

Utilizando estas variables de entorno, un programa CGI no sólo recupera datos de entrada, sino que, además, reconoce el tipo de cliente y servidor con el que está tratando.

En las siguientes secciones, veremos las variables de entorno que están disponibles desde el módulo CGI estándar compilado en Apache.

NOTA: La distribución del código fuente de la versión 2.x.x de Apache, soporta la opción `--enable-cgid` para el script `configure`. Esta opción fuerza a Apache a utilizar un servidor de scripts (llamado demonio CGI) para manejar los procesos del script CGI, los cuales aumentan el rendimiento general de Apache.

Variables del servidor

Estas variables son asignadas por Apache para informar a los programas CGI sobre Apache. Utilizando variables del servidor, un programa CGI puede determinar información específica sobre varios servidores, como la versión del software de Apache, la dirección de correo electrónico del administrador, e información de este tipo.

SERVER_SOFTWARE

SERVER_SOFTWARE está asignado por Apache, y el valor se encuentra habitualmente de la siguiente forma:

```
Apache/Version (OS Info)
```

En este caso, Apache es el nombre del software del servidor que está ejecutando el programa CGI, y version es el número de la versión de Apache. Un valor de ejemplo sería:

```
Apache/2.0.14 (Unix)
```

Esto resulta útil cuando un programa CGI se utiliza para sacar partido de una característica nueva que se encuentra en la última versión de Apache, y sigue siendo capaz de rendir en versiones antiguas.

GATEWAY_INTERFACE le dice al programa CGI qué versión de la especificación CGI soporta actualmente el servidor. Un valor de ejemplo sería:

```
CGI/1.1
```

Un programa CGI puede determinar el valor de esta variable y, de forma condicional, hacer uso de las distintas características disponibles en las diferentes versiones de las especificaciones CGI. Por ejemplo, si el valor es CGI/1.0, el programa no utilizará ninguna característica CGI/1.1, o viceversa.

El primer entero antes de la coma decimal se llama número principal, y el entero que se encuentra detrás de la coma es el número secundario. Como estos dos enteros se tratan como números separados, CGI/2.2 es una versión más antigua que CGI/2.15.

SERVER_ADMIN

Si utiliza la directiva ServerAdmin en el archivo httpd.conf para determinar la dirección de correo electrónico del administrador del sistema, esta variable será asignada para reflejarlo. Además, observe que si tiene una directiva ServerAdmin en un contenedor de configuración de un host virtual, la variable SERVER_ADMIN se asigna a esa dirección en el caso de que el programa CGI al que se está accediendo sea parte del host virtual.

DOCUMENT_ROOT

Esta variable está asignada al valor de la directiva DocumentRoot del sitio Web al que se está accediendo.

Variables para las solicitudes del cliente

Apache crea un conjunto de variables de entorno desde la cabecera de la solicitud HTTP que recibe desde una solicitud de un programa CGI. Proporciona

esta información al programa CGI creando el siguiente conjunto de variables de entorno.

SERVER_NAME

Esta variable le dice al programa CGI a qué host se está accediendo. El valor es una dirección IP o el nombre completo del host:

```
SERVER_NAME = 192.168.1.100
SERVER_NAME = www.domain.com
```

HTTP_HOST

Ver la variable SERVER_NAME.

HTTP_ACCEPT

Esta variable está asignada a la lista de tipos MIME que el cliente puede aceptar, incluidos los siguientes:

```
HTTP_ACCEPT = image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
image/png, */*
```

Aquí, el cliente dice que es capaz de manejar imágenes GIF, JPEG, PNG, y otro tipo de imágenes. Esto le permite al programa CGI determinar el tipo de salida ideal para el cliente.

Por ejemplo, un programa CGI podría producir GIF o JPEG y recibir una HTTP_ACCEPT:

```
HTTP_ACCEPT = image/gif,  */*
```

Entonces, puede enviar un resultado GIF en lugar de uno JPEG porque el cliente lo prefiere de ese modo.

HTTP_ACCEPT_CHARSET

Esta variable determina el conjunto de caracteres aceptable para el cliente, por ejemplo:

```
HTTP_ACCEPT_CHARSET = iso-8859-1,*,utf-8
```

HTTP_ACCEPT_ENCODING

Esta variable determina los esquemas de codificación aceptables para el cliente, por ejemplo:

```
HTTP_ACCEPT_ENCOMDING = gzip
```

En este caso, el cliente acepta archivos `gzip` (comprimidos). De ese modo, un script CGI puede comprimir una página grande utilizando la compresión `gzip` y enviándola después, y el cliente Web será capaz de descomprimirlo.

HTTP_ACCEPT_LANGUAGE

Esta variable determina el lenguaje aceptable para el cliente, por ejemplo:

```
HTTP_ACCEPT_LANGUAGE = en
```

En este caso, el cliente acepta el contenido en inglés.

HTTP_USER_AGENT

Esta variable determina qué software del cliente y qué sistema operativo está ejecutando el sistema, por ejemplo:

```
HTTP_USER_AGENT = Mozilla/4.04 [en] (WinNT; I)
```

Este código es equivalente al siguiente:

```
Client Software = Netscape Navigator ()
Client Software Version = 4.04 (English version)
Operating System = Windows NT (Intel)
```

Observe que Mozilla es una palabra clave utilizada por Netscape para la base de código de Navigator. Aunque únicamente los navegadores Netscape utilizan la palabra Mozilla, muchos otros fabricantes han comenzado a utilizar Mozilla como parte de la cabecera HTTP. Por ejemplo, Microsoft Internet Explorer (IE) 5.5 produce el siguiente dato `HTTP_USER_AGENT` cuando se ejecuta en la misma máquina:

```
HTTP_USER_AGENT = Mozilla/5.5 (compatible; MSIE 5.5; Windows NT)
```

Esta información sobre el agente usuario, se utiliza en muchos sitios Web. Un sitio que está optimizado para Netscape Navigator (es decir, que utiliza una característica de HTML, o JavaScript, o un a plug-in, que funciona adecuadamente en Netscape Navigator) debería utilizar la información `HTTP_USER_AGENT` para devolver una página distinta para los usuarios que navegan en el sitio con IE, o para cualquier otro navegador menos conocido. Sin embargo, recomiendo que se adhiera al estándar HTML (la especificación HTML para el estándar actual, se encuentra disponible en `www.w3.org`), y que no implemente ninguna característica específica de navegador.

Aunque optimizar sus páginas para un navegador concreto dará lugar a unas páginas perfectas en ese navegador, recuerde que no todo el mundo utiliza el mismo navegador. Esto significa que sus etiquetas HTML o sus plug-ins específicos dificultarán la visita a su sitio Web a todos aquellos que no utilicen su navegador Web preferido.

HTTP_REFERER

Esta variable está asignada al Uniform Resource Identifier (URI) que dirige las solicitudes al programa CGI al que se ha llamado. Utilizando esta variable, puede decidir si la solicitud proviene del enlace de una de sus páginas Web o de

una URI remota. Observe que un error de ortografía en el nombre de la variable da lugar a confusión en los desarrolladores de CGI, que lo están deletreando correctamente en sus aplicaciones y scripts, al descubrir que no están funcionando. Por eso, si ha pensado utilizar esta variable, deletréelo tal y como se indica aquí.

HTTP_CONNECTION

La variable HTTP_CONNECTION está asignada al tipo de conexión que utiliza el cliente y el servidor. Por ejemplo:

```
HTTP_CONNECTION = Keep-Alive
```

Esto indica que el cliente es capaz de manejar conexiones utilizando Keep-Alive y que las está utilizando.

SERVER_PORT

El valor de la variable SERVER_PORT le dice al programa CGI qué puerto del servidor se está utilizando para acceder al programa. Un ejemplo sería:

```
SERVER_PORT = 80
```

Si un programa CGI crea unas URL que se dirigen al servidor, sería útil incluir la dirección del puerto, que se encuentra como el valor de esta variable, en la URL.

REMOTE_HOST

La variable REMOTE_HOST informa a un programa CGI sobre la dirección IP o el nombre IP del cliente, del siguiente modo:

```
REMOTE_HOST = dsl-666.isp24by7.net
```

Observe que si el servidor Apache está compilado con la opción MINIMAL_DNS, no se asigna esta variable.

REMOTE_PORT

El cliente utiliza este número de puerto en el lado del cliente de la conexión socket:

```
REMOTE_PORT = 1163
```

No he encontrado aún la utilidad a esta variable.

REMOTE_ADDR

La variable REMOTE_ADDR es la dirección IP del sistema del cliente:

```
REMOTE_ADDR = 192.168.1.100
```

Observe que si el cliente está tras un firewall o un servidor proxy, la dirección IP almacenada en esta variable no tiene por qué ser la dirección IP del sistema del cliente.

REMOTE_USER

La variable REMOTE_USER será asignada únicamente cuando el acceso al programa CGI requiera autentificación HTTP basic. El nombre de usuario utilizado en esta autentificación, se almacena en esta variable para el programa CGI. El programa CGI, sin embargo, no tendrá modo de identificar la contraseña utilizada para acceder a él. Si esta variable está asignada al nombre de usuario, el programa CGI puede suponer con seguridad que el usuario suministró la contraseña adecuada para el acceso.

SERVER_PROTOCOL

SERVER_PROTOCOL es el protocolo y el número de versión que utiliza el cliente para enviar la solicitud para el programa CGI:

```
SERVER_PROTOCOL = HTTP/1.1
```

REQUEST_METHOD

La variable REQUEST_METHOD asigna el método de solicitudes HTTP utilizado por el cliente para solicitar un programa CGI. Los valores habituales son: GET, POST y HEAD.

```
REQUEST_METHOD=GET
```

La entrada se almacena en la variable QUERY_STRING cuando el método es GET. Cuando el método es POST, la entrada se almacena en el STDIN del programa CGI.

REQUEST_URI

La variable REQUEST_URI determina el URI de la solicitud.

```
REQUEST_URI = /cgi-bin/printenv2
```

REMOTE_IDENT

REMOTE_IDENT será asignada sólo si se asigna la directiva IdentityCheck. Esta variable almacena la información de identificación devuelta por el identd (demonio de identificación) remoto. Como muchos sistemas no ejecutan este tipo de procesos demonio, REMOTE_IDENT no debería considerarse seguro en la identificación de usuarios. Recomiendo utilizar esta variable en un entorno de intranet o de extranet en el que usted o su organización esté ejecutando un servidor identd.

AUTH_TYPE

Si un programa CGI se almacena en una sección del sitio Web en la que se requiere la autentificación para acceder, esta variable se asigna para especificar el método de autentificación utilizado.

CONTENT_TYPE

Esta variable especifica el tipo MIME de cualquier dato adjunto a la cabecera de la solicitud. Por ejemplo:

```
CONTENT_TYPE = application/x-www-form-urlencoded
```

Cuando utilizamos un formulario HTML y el método `POST`, puede especificar el tipo de contenido en el formulario HTML utilizando el atributo `TYPE` de la etiqueta `<FORM>`, del siguiente modo:

```
<FORM ACTION="/cgi-bin/search.cgi"
      METHOD="POST"
      TYPE= "application/x-www-form-urlencoded">
```

CONTENT_LENGTH

Cuando se utiliza el método `POST`, Apache almacena los datos introducidos (adjuntos a la solicitud) en el `STDIN` del programa CGI. El servidor no inserta un marcador End-of-File (`EOF`) en el `STDIN`. Sin embargo, a esta variable se le asigna el número de bytes. Por ejemplo, si:

```
CONTENT_LENGTH = 21
```

entonces el programa CGI en cuestión leería 21 bytes de datos desde su `STDIN`.

SCRIPT_NAME

`SCRIPT_NAME` es el URI del programa CGI:

```
SCRIPT_NAME = /cgi-bin/search.cgi
```

SCRIPT_FILENAME

`SCRIPT_FILENAME` es el nombre completo de la ruta del programa CGI:

```
SCRIPT_FILENAME = /www/kabir/public/cgi-bin/search.cgi
```

QUERY_STRING

Si un cliente Web, un navegador Web por ejemplo, utiliza el método `GET` y proporciona los datos introducidos tras una interrogación (?), el dato se almacena como el valor de esta variable. Por ejemplo, una solicitud para el siguiente programa CGI:

```
http://apache.domain.com/cgi-bin/search.cgi?key1=value1&key2=
value2
```