incluya ningún método en la directiva <Limit>. Tenga en cuenta que este contenedor no se puede anidar, ni tampoco puede aparecer un contenedor <Directory> dentro de él. Los nombres de los métodos cambian al utilizar mayúsculas o minúsculas.

Sintaxis: `<Limit method method ... > ... </Limit>`

Predefinido: ninguno

Contexto: todos

<LimitExcept>

La directiva del contenedor <LimitExcept> se utiliza de forma opuesta a la directiva <Limit>, <limit> limita los métodos nombrados (por ejemplo, arguments) y <LimitExcept> limita todo lo que no sean argumentos. Todos los métodos que no están en la lista de argumentos están limitados.

Sintaxis: `<LimitExcept metodo metodo ... > ... </LimitExcept>`

Predefinido: ninguno

Contexto: todos

En el siguiente ejemplo, se aplica el límite a todos los métodos HTTP excepto a GET.

```
<LimitExcept GET>
    # directivas
</LimitExcept>
```

LimitRequestBody

La directiva LimitRequestBody le permite asignar un límite en el tamaño de la solicitud HTTP que Apache servirá. El límite por defecto es 0, que significa ilimitado. Puede asignar este límite de 0 a 2147483647 (2GB).

Sintaxis: `LimitRequestBody bytes`

Predefinido: `LimitRequestBody 0`

Contexto: configuración del servidor, host virtual, directorio, configuración en el ámbito de directorio

Se recomienda asignar un límite, sólo si tiene experiencia en evitar ataques basados en la denegación de servicios HTTP, en los que intentan sobrecargar el servidor con grandes solicitudes HTTP. Se trata de una directiva útil a la hora de mejorar la seguridad en el servidor.

LimitRequestFields

La directiva `LimitRequestFields` le permite limitar el número de campos de cabeceras de solicitudes permitidos en una sola solicitud HTTP. Este límite se puede encontrar entre 0 y 32767 (32K). Esta directiva le puede ayudar a implementar las medidas de seguridad contra ataques basados en la denegación de servicios con grandes solicitudes.

Sintaxis: `LimitRequestFields número`

Predefinido: `LimitRequestFields 100`

Contexto: configuración del servidor

LimitRequestFieldsize

La directiva `LimitRequestFieldsize` le permite limitar el tamaño (en bytes) de un campo de cabecera de solicitud. El tamaño por defecto es 8190 (8K) que es más que suficiente para la mayor parte de las situaciones. Sin embargo, si tiene lugar un ataque basado en la denegación de servicios HTTP, puede cambiarlo a un número más pequeño para denegar las solicitudes que excedan el límite. El valor 0 asigna el límite en ilimitado.

Sintaxis: `LimitRequestFieldsize` *`bytes`*

Predefinido: `LimitRequestFieldsize 8190`

Contexto: configuración del servidor

LimitRequestLine

La directiva `LimitRequestLine` asigna el límite de tamaño de la línea de solicitudes. En realidad limita el tamaño de la URL que se puede enviar al servidor. El límite por defecto debería ser suficiente para la mayoría de las situaciones. Si tiene lugar un ataque basado en la denegación de servicios que utilice una URL muy larga para agotar los recursos en su servidor, puede reducir el límite para rechazar este tipo de solicitudes.

Sintaxis: `LimitRequestLine bytes`

Predefinido: `LimitRequestLine 8190`

Contexto: configuración del servidor

Require

Apache determina qué usuarios o grupos pueden acceder a un directorio restringido utilizando la directiva `Require`. Hay tres tipos de nombres de entidad

disponibles: user, group, valid-user. Por ejemplo, `require user joe jenny` le dice a Apache que permita la entrada al área únicamente a joe o a jenny tras una autentificación con éxito. Sólo los usuarios nombrados pueden acceder al directorio.

Sintaxis: `Require nombre-entidad nombre-entidad...`

Predefinido: ninguno

Contexto: directorio, configuración en el ámbito de directorio

Invalidar: `AuthConfig`

A continuación tenemos un ejemplo en el que sólo los usuarios de los grupos nombrados pueden acceder al directorio, se trata de un ejemplo con un requisito de acceso basado en grupo:

```
Require group my-group your-group his-group her-group
```

Con la siguiente línea, todos los usuarios válidos pueden acceder al directorio.

```
require valid-user
```

Si la directiva `require` aparece en una sección `<Limit>`, entonces restringe el acceso a los métodos nombrados; en caso contrario, restringe el acceso para todos los métodos. Por ejemplo:

```
AuthType Basic
AuthName "Game Zone Drop Box"
AuthUserFile   /www/netgames/.users
AuthGroupFile  /www/ntgames/.groups

<Limit GET>
   require group coders
</Limit>
```

Si la configuración anterior se encuentra en un archivo `.htaccess` en un directorio, únicamente un grupo llamado `coders` tiene permiso de acceso al directorio para recuperar archivos mediante el método HTTP `GET`. Para que funcione adecuadamente la directiva `Require` debe ir acompañada de las directivas `AuthName` y `AuthType`, y por directivas del tipo `AuthUserFile` y `AuthGroupFile`.

Para obtener los detalles de autentificación, ver los capítulos siguientes.

Satisfy

Si ha creado una configuración básica de autentificación HTTP en la que se utilizan las directivas `Allow` y `Require`, puede utilizar la directiva `Satisfy` para decirle a Apache que los requisitos de autentificación son suficientes.

Sintaxis: `Satisfy Any | All`

Predefinido: `Satisfy all`

Contexto: directorio, nivel de directorios

El valor de la directiva `Satisfy` puede ser `all` o `any`. Si el valor es `all`, entonces la autentificación será exitosa en el caso de que tengan éxito `Allow` y `Require`. Si el valor es `any`, entonces la autentificación será exitosa en el caso de que no lo sean ni `Allow` ni `Require`.

La directiva `Satisfy` es útil sólo si se restringe una determinada zona, tanto con contraseña y nombre de usuario como con la dirección del host cliente. En este caso, el comportamiento por defecto (`all`) requiere que el cliente pase la restricción de acceso de dirección e introduzca una contraseña y un nombre de usuario válidos. Con la opción `any`, se permite el acceso al cliente si el usuario, o bien pasa la restricción del host o bien introduce un nombre de usuario y una contraseña válidos. Esta directiva se puede utilizar para restringir el acceso a una zona utilizando contraseñas, al mismo tiempo, y de forma simultánea, le da acceso a todos los clientes de un conjunto determinado de direcciones IP (es decir, un conjunto de direcciones IP) sin pedirles que introduzcan contraseña.

ScriptInterpreterSource

La directiva `ScriptInterpreterSource` le permite especificar el modo en que Windows encuentra el intérprete para un script. Normalmente, el intérprete de script se detecta utilizando la línea #! encontrada en un script. Sin embargo, asignando esta directiva para que registre, fuerza a Windows a realizar una búsqueda en el registro de las extensiones de los script para encontrar el programa solicitado (es decir, el intérprete).

Sintaxis: `ScriptInterpreterSource Registry | Script`

Predefinido: `ScriptInterpreterSource script`

Contexto: directorio, `.htaccess`

Directivas específicas de MPM threaded

Es igual que en los MPM prefork, pero en lugar de que cada proceso hijo tenga un solo hilo, cada proceso hijo puede tener un número determinado de hilos. Como los hilos son más eficaces en cuanto a recursos que los procesos, este MPM es muy escalable. Cada hilo dentro de un proceso hijo puede servir una solicitud distinta.

Los procesos se añaden o se eliminan controlando su conteo de hilos producidos. Por ejemplo, si un proceso tiene menos hilos producidos que el valor mínimo,

se añade un nuevo proceso. De igual modo, cuando un proceso tiene un número máximo de hilos parados, es asesinado.

NOTA: Todos los procesos se ejecutan bajo el mismo ID de usuario y de grupo, asignado por el servidor Apache.

CoreDumpDirectory

La directiva `CoreDumpDirectory` asigna el directorio que Apache intenta cambiar antes de que se estropee el archivo general. La localización por defecto es el directorio especificado por la directiva `ServerRoot`.

Sintaxis: `CoreDumpDirectory ruta-directorio`

Predefinido: directorio de la ruta del servidor

Contexto: configuración del servidor

Group

La directiva `Group` debería utilizarse en unión con la directiva `User`. `Group` determina el grupo bajo el cual el servidor responde a las solicitudes. Para utilizar esta directiva, el servidor debe ejecutarse inicialmente como raíz. La directiva `Group` puede tener un número de grupo como valor asignado. `Group` busca nombres de grupo y sus correspondientes valores numéricos en su archivo `/etc/group`.

Sintaxis: `Group Unix-group`

Predefinido: `Group #-1`

Contexto: Configuración del servidor, host virtual

NOTA: Todas las advertencias y [illegible] se pueden aplicar a esta directiva, [illegible] tiva `User` más [illegible]

Listen

Por defecto, Apache responde a las solicitudes en todas las direcciones IP adjuntas al servidor, pero sólo por la dirección del puerto especificado por la directiva `Port`. La directiva `Listen` se puede utilizar para hacer esta situación más configurable. Puede utilizar la directiva `Listen` para pedirle a Apache que

responda a cierta dirección IP, a una combinación de dirección IP y de puerto o, simplemente a un puerto.

Sintaxis: `Listen [IP address:] numero-puerto`

Predefinido: ninguno

Contexto: configuración del servidor

A pesar de que se puede utilizar `Listen` en lugar de `BindAddress` y `Port`, podría tener que utilizar la directiva `Port` si su servidor Apache genera URLs que se dirigen a él mismo.

Se pueden utilizar varias directivas `Listen` para especificar el número de direcciones y puertos a los que escuchar. El servidor responderá a las solicitudes desde cualquiera de las direcciones y puertos de la lista. Por ejemplo, para conseguir que el servidor acepte conexiones del puerto 80 y del puerto 8080, utilice:

```
Listen 80
Listen 8080
```

El siguiente ejemplo, consigue que Apache acepte conexiones en dos direcciones IP y en dos números de puerto:

```
Listen 192.168.1.100:80
Listen 192.168.1.101:8080
```

ListenBacklog

La directiva `ListenBacklog` le permite tomar medidas de defensa contra un ataque a la seguridad llamado Denegación de Servicios (DOS), permitiéndole que asigne la longitud máxima de la cola de conexiones pendientes. Auméntelo en caso de que detecte que se encuentra ante un ataque TCP SYN flood (DOS); en caso contrario, déjelo como está.

Sintaxis: `ListenBacklog pendientes`

Predefinido: `ListenBacklog 511`

Contexto: configuración del servidor

LockFile

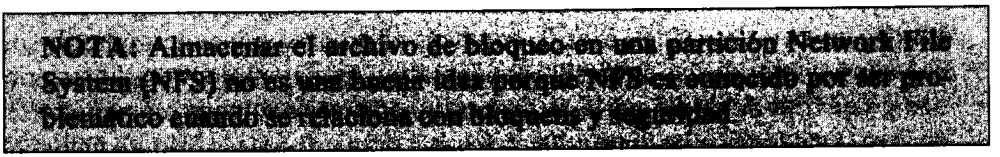
Si compila Apache con la opción `USE_FCNTL_SERIALIZED_ACCEPT` o la opción `USE_FLOCK_SERIALIZED_ACCEPT`, se utiliza un archivo de bloqueo. Puede utilizar la directiva `LockFile` para asignar la ruta al nombre de archivo del archivo de bloqueo.

Asegúrese de que únicamente el servidor Apache tiene acceso para lectura y escritura en el archivo.

Sintaxis: `LockFile nombre archivo`

Predefinido: `LockFile logs/accept.lock`

Contexto: configuración del servidor



MaxClients

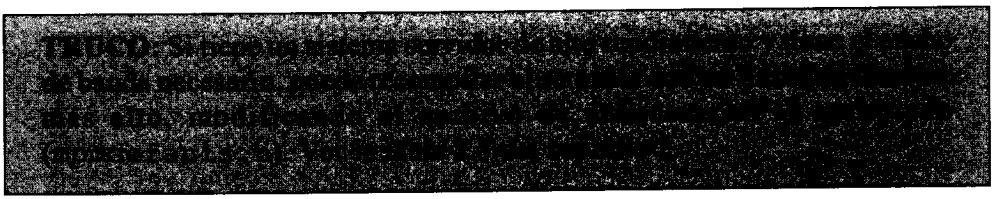
La directiva `MaxClients` limita el número de solicitudes simultaneas que Apache puede servir. Como Apache utiliza un servidor hijo para cada solicitud, éste es también el límite efectivo para el número de servidores hijo que pueden existir al mismo tiempo.

Sintaxis: `MaxClients numero`

Predefinido: `MaxClients 256`

Contexto: configuración del servidor

El límite por defecto es realmente el límite asignado en el archivo `httpd.h` en la distribución fuente de Apache. Esta asignación debería servir para los sitios con carga de normal a moderada. Los programadores de Apache ponen este límite de hardware ahí por dos razones: no quieren que el servidor quiebre el sistema rellenando alguna tabla kernel, y este límite máximo mantiene el archivo marcador lo bastante pequeño como para que se pueda leer con facilidad. Cuando el servidor alcanza el máximo conteo de solicitudes, deja las solicitudes entrantes en estado de espera hasta que queda libre para darles servicio.



MaxRequestsPerChild

Apache lanza un proceso hijo para servir una solicitud; sin embargo, un servidor hijo puede procesar varias solicitudes. El número de solicitudes que puede procesar un servidor hijo está limitado por la directiva `MaxRequestsPerChild`.

Sintaxis: `MaxRequestsPerChild numero`

Predefinido: `MaxRequestsPerChild 0`

Contexto: configuración del servidor

Tras servir el máximo número de solicitudes, el proceso hijo termina. Si `MaxRequestsPerChild` es 0, entonces el proceso nunca termina. Si sospecha que hay bibliotecas en su sistema operativo (por ejemplo, Solaris) que tienen código de filtrado de memoria, debería asignarle a esta directiva un valor distinto de cero. Esto le permite definir un ciclo vital para un proceso hijo, reduciendo las posibilidades de que un proceso consuma memoria filtrada y de que poco a poco gaste la memoria disponible. Además le proporciona una pequeña carga media para su sistema, porque la carga relacionada con Apache se reduce a medida que su servidor Web está menos ocupado.

MaxSpareThreads

La directiva `MaxSpareThreads` fija el número máximo de hilos parados. El MPM `threaded` trata con los hilos parados en la base de un servidor, que significa que si hay demasiados hilos parados en el servidor, comienza a destruir procesos hijo hasta que el número de hilos parados es menor que el número que se especifica aquí.

Sintaxis: `MaxSpareThreads número`

Predefinido: `MaxSpareThreads 10` (para MPM Perchild) o 500 (para MPM threaded)

Contexto: configuración del servidor

El MPM `perchild` cuenta los hilos parados en la base de cada hijo, lo que significa que si hay demasiados hilos parados en un hijo, se destruye el hijo hasta que la cuenta de hilos por hijo sea menor que el número especificado con la directiva `MaxSpareThreads`.

MinSpareThreads

La directiva `MinSpareThreads` determina el mínimo número de hilos parados. El MPM `threaded` negocia con los hilos parados en la base de un servidor, lo que significa que cuando hay menos hilos parados que el número especificado aquí, Apache crea nuevos procesos hijo para alcanzar este número.

Sintaxis: `MinSpareServers número`

Predefinido: `MaxSpareThreads 5` (para MPM Perchild) o 250 (para MPM threaded)

Contexto: configuración del servidor

El MPM `perchild` maneja el conteo de hilos parados en la base de cada hijo; por lo tanto, cuando un hijo tiene un número menor de hilos que los que se especifican aquí, el servidor crea hilos nuevos dentro de esos procesos hijo.

SendBufferSize

La directiva `SendBufferSize` fija el tamaño de buffer TCP enviado en el número de bytes especificado. En una red de gran rendimiento, asignar a esta directiva un valor más alto que el valor por defecto del sistema operativo aumentará el rendimiento del servidor.

Sintaxis: `SendBufferSize bytes`

Predefinido: ninguno

Contexto: configuración del servidor

StartServers

La directiva `StartServers` fija el número de procesos hijo del servidor Apache que se crean en la puesta en marcha. El número de procesos hijo Apache necesarios para un determinado período de tiempo se controla de forma dinámica. El servidor principal de Apache (el proceso demonio) lanza un proceso hijo nuevo a medida que encuentra mayor carga de solicitudes.

Las directivas `MinSpareServers`, `MaxSpareServers` y `MaxClients` controlan el número real de procesos hijo. Por lo tanto, tiene poco que ganar ajustando este parámetro.

Sintaxis: `StartServers número`

Predefinido: `StartServers 5`

Contexto: configuración del servidor

NOTA: La directiva `StartServers` es útil únicamente cuando el servidor Apache se ejecuta como un servidor autónomo. En otras palabras, necesita tener fijado `ServeType` autónomo para que esta directiva sea efectiva.

NOTA: Cuando ejecutamos Microsoft Windows, esta directiva determina el número total de procesos hijo en ejecución. Como la versión Windows de Apache es multihilo, un proceso maneja todas las solicitudes. El resto de los procesos están en reserva hasta que el proceso principal muera.

ThreadsPerChild

La versión Windows de Apache es un servidor multihilo. La directiva `ThreadsPerChild` le dice al servidor cuántos hilos debería utilizar. También determina el número máximo de conexiones que puede manejar el servidor en un momento dado. Por lo tanto, este valor debería asignarse en un valor razonablemente elevado para permitir el número máximo de éxitos.

Sintaxis: `ThreadsPerChild numero`

Predefinido: `ThreadsPerChild 50`

Contexto: configuración del servidor (Windows)

User

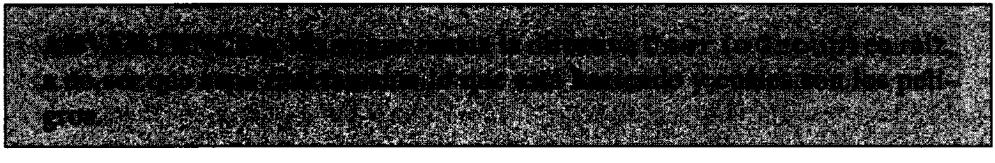
La directiva `User` asigna el ID de usuario que utilizan los hijos Apache que sirven solicitudes HTTP. Una vez que se inicia el servidor Apache, lanza procesos hijo para responder a las solicitudes. Sin embargo, estos procesos hijo se ejecutan como raíz. El proceso padre de Apache (a menudo llamado demonio) cambia el ID del usuario del proceso hijo a cualquiera de los asignados en la directiva `User`, siempre que se trate de un ID de usuario válido.

Sintaxis: `User unix-userid`

Predefinido: `User #-1`

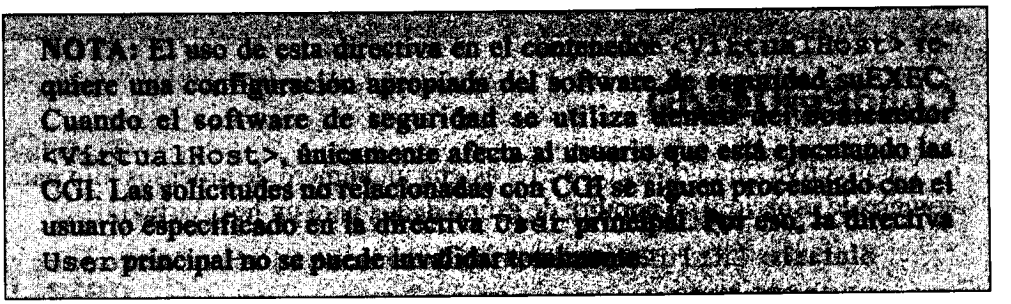
Contexto: configuración del servidor, host virtual

Si no inicia el servidor como usuario raíz, no podrá cambiar el ID de usuario especificado por la directiva `User`, y en lugar de eso, continuará ejecutándose como el usuario original. Si comienza el servidor como raíz, entonces es normal que el proceso padre de Apache se quede ejecutándose como raíz; sin embargo, ejecuta el proceso hijo como el usuario especificado por la directiva `User`.



También puede utilizar los números ID de los usuarios, que puede encontrar normalmente en el archivo `/etc/password`. Si piensa utilizar un valor numérico en lugar del nombre de usuario real, el número debe ir precedido del signo #. Muchos administradores de Apache utilizan el usuario por defecto `nobody` para sus sitios Web. Este usuario no está disponible en todos los sistemas Unix, y además no siempre es deseable. Recomiendo encarecidamente que emplee un solo

ID de usuario y de grupo (ver la directiva `Group`) en su servidor Apache. El hacerlo le dará un mayor control sobre los accesos al servidor. El ID que decida utilizar para los procesos hijo de Apache debería tener muy pocos privilegios. No debería permitirse el acceso a archivos que no tiene intención de mostrar fuera de su sitio Web, y de igual modo, el usuario no debería ser capaz de ejecutar aplicaciones que no tienen sentido para las solicitudes HTTP.



Directiva específicas de MPM perchild

En este modelo MPM, se inicia un conjunto de procesos hijo con un número determinado de hilos. A medida que aumenta la carga de solicitudes, los procesos añaden nuevos hilos según los van necesitando. Cuando se reduce el conteo de solicitudes, los procesos disminuyen sus conteos de hilos utilizando un máximo y un mínimo asignado de conteo.

AssignUserID

La directiva `AssignUserID` asigna un nombre de usuario y un nombre de grupo a un host virtual.

Sintaxis: `AssignUserID nombreusuario nombregrupo`

Predefinido: ninguno

Contexto: host virtual

Por ejemplo, a continuación podemos ver cómo un host virtual llamado `www.afactcat.com` es asignado a un usuario `mrbert` y a un grupo llamado `wheel`. Debe utilizar `ChildPerUserID` para especificar el número de procesos hijo que puede servir este host virtual.

```
<VirtualHost  192.168.1.100>
    ServerName www.afatcat.com
    AssignUserID mrbert wheel

    #
```

```
    # Aquí van otras directivas
    #
</VirtualHost>
```

NOTA: `username` y `groupname` deben existir en su sistema. Por ejemplo, en un sistema Linux `username` y `groupname` deben estar en los archivos `/etc/passwd` y `/etc/group`, respectivamente.

ChildPerUserID

La directiva `ChildPerUserID` asigna un número de procesos hijo para un nombre de usuario y un nombre de grupo dados para un host virtual.

Sintaxis: `ChildPerUserID número de hijos nombre de usuario nombre de grupo`

Predefinido: ninguno

Contexto: host virtual

En el siguiente ejemplo, el host virtual `www.afatcat.com` será servido por los procesos hijo de Apache que se ejecutan bajo el nombre de usuario `mrbert` y el nombre de grupo `wheel`:

```
    ChildPerUserID 10 mrbert wheel
<VirtualHost  192.168.1.100>
    ServerName www.afatcat.com
    AssignUserID mrbert wheel
    #
    # Aquí van otras directivas
    #
</VirtualHost>
```

NOTA: `username` y `groupname` deben existir en su sistema para que Apache sea capaz de utilizarlos. Por ejemplo, en un sistema Linux `username` y `groupname` deben existir en los archivos `/etc/passwd` y `/etc/group`, respectivamente.

ConnectionStatus

La directiva `ConnectionStatus` determina si la información del estado se almacena internamente o no. Cuando se le asigna el valor `on`, los módulos que utilizan la información del estado funcionarán adecuadamente.

Sintaxis: `ConnectionStatus On | Off`

Predefinido: `ConnectionStatus On`

Contexto: configuración del servidor

CoreDumpDirectory

Ver la directiva `CoreDumpDirectory` en la sección "Directivas específicas de MPM threaded".

Group

Ver la directiva `Group` en la sección "Directivas específicas de MPM threaded".

Listen

Ver la directiva `Listen` en la sección "Directivas específicas de MPM threaded".

ListenBacklog

Ver la directiva `ListenBacklog` en la sección "Directivas específicas de MPM threaded".

LockFile

Ver la directiva `LockFile` en la sección "Directivas específicas de MPM threaded".

MaxRequestsPerChild

Ver la directiva `MaxRequestsPerChild` en la sección "Directivas específicas de MPM threaded".

MaxSpareThreads

Ver la directiva `MaxSpareThreads` en la sección "Directivas específicas de MPM threaded".

MaxThreadsPerChild

La directiva `MaxThreadsPerChild` determina el número máximo de hilos por hijo. El número por defecto es el límite de harware. Si desea cambiarlo tiene

que modificar el archivo de cabecera apropiado. Ver la tabla 2.3 del capítulo 2 para obtener los detalles.

Sintaxis: `MaxThreadsPerChild number`

Predefinido: `MaxThreadsPerChild 64`

Contexto: configuración del servidor

MinSpareThreads

Ver la directiva `MinSpareThreads` en la sección "Directivas específicas de MPM threaded".

NumServers

La directiva `NumServers` determina el número de procesos hijo simultáneos que crea Apache.

Sintaxis: `NumServers number`

Predefinido: `NumServers 2`

Contexto: configuración del servidor

El MPM per-child utiliza el valor asignado por esta directiva para determinar el número de hijos simultáneos. El valor por defecto 2 no es el apropiado en todos los sistemas.

Puede cambiarlo a un valor más alto como 15 o 20. Cuanto mayor es el número, más procesos Apache se producen, lo que significa que se pueden manejar más conexiones simultaneas.

PidFile

Ver la directiva `PidFile` en la sección "Directivas específicas de MPM threaded".

ScoreBoardFile

Ver la directiva `ScoreBoardFile` en la sección "Directivas específicas de MPM threaded".

SendBufferSize

Ver la directiva `SendBufferSize` en la sección "Directivas específicas de MPM threaded".

StartThreads

La directiva `StartThreads` determina el conteo inicial de hilos por hijo. Como los conteos de hijos se controlan dinámicamente, asignar un número superior que el asignado por defecto no suele ser necesario.

Sintaxis: `StartThreads número`

Predefinido: `StartThreads 5`

Contexto: configuración del servidor

User

Ver la directiva `User` en la sección "Directivas específicas de MPM threaded".

Directivas específicas de MPM

Este es el MPM para todas las versiones de la plataforma Windows, incluido Windows NT/2000/XP y Windows 9x/ME. Este módulo es multihilo; utilizando este módulo Apache creará un proceso padre y un proceso hijo. El proceso hijo crea todos los hilos que sirven la solicitud. Este módulo saca partido de algunas llamadas a funciones nativas sólo de Windows, lo que le permite funcionar mejor que versiones anteriores del servidor Apache en la plataforma Windows.

CoreDumpDirectory

Ver la directiva `CoreDumpDirectory` en la sección "Directivas específicas de MPM threaded".

Listen

Ver la directiva `Listen` en la sección "Directivas específicas de MPM threaded".

ListenBacklog

Ver la directiva `ListenBacklog` en la sección "Directivas específicas de MPM threaded".

MaxRequestsPerChild

Ver la directiva `MaxRequestsPerChild` en la sección "Directivas específicas de MPM threaded".

PidFile

Ver la directiva `PidFile` en la sección "Directivas específicas de MPM threaded".

SendBufferSize

Ver la directiva `SendBufferSize` en la sección "Directivas específicas de MPM threaded".

ThreadsPerChild

Ver la directiva `ThreadsPerChild` en la sección "Directivas específicas de MPM threaded".

Directivas específicas de MPM prefork

El MPM prefork crea un grupo de procesos hijo para servir solicitudes. Cada proceso hijo tiene un solo hilo. Por ejemplo, si Apache inicia 30 procesos hijo, puede servir 30 solicitudes simultáneamente. Si algo sale mal y el proceso hijo muere, únicamente se pierde una solicitud. El número de procesos hijo se controla utilizando un mínimo y un máximo fijos. Cuando el número de solicitudes aumenta, se añade un nuevo proceso hijo hasta que se alcanza el máximo. De igual modo, cuando falla la solicitud, se eliminan todos los procesos hijo extra.

CoreDumpDirectory

Ver la directiva `CoreDumpDirectory` en la sección "Directivas específicas de MPM threaded".

Group

Ver la directiva `Group` en la sección "Directivas específicas de MPM threaded".

Listen

Ver la directiva `Listen` en la sección "Directivas específicas de MPM threaded".

ListenBacklog

Ver la directiva `ListenBacklog` en la sección "Directivas específicas de MPM threaded".

LockFile

Ver la directiva `LockFile` en la sección "Directivas específicas de MPM threaded".

MaxClients

Ver la directiva `MaxClients` en la sección "Directivas específicas de MPM threaded".

MaxRequestsPerChild

Ver la directiva `MaxRequestsPerChild` en la sección "Directivas específicas de MPM threaded".

MaxSpareServers

Esta directiva le permite determinar el número de procesos hijo de Apache inactivos en su servidor.

Sintaxis: `MaxSpareServers número`

Predefinido: `MaxSpareServers 10`

Contexto: configuración del servidor

Si el número de procesos hijo de Apache inactivos excede el número máximo especificado por la directiva `MaxSpareServers`, entonces el proceso padre asesina el exceso de procesos. Sólo es necesario ajustar este parámetro en el caso de sitios realmente visitados. A no ser que sepa lo que hace, no cambie el valor por defecto.

MinSpareServers

La directiva `MinSpareServers` determina el número que deseamos tener de procesos hijo inactivos en el servidor. Un proceso inactivo es aquel que no está manejando ninguna solicitud. Si hay menos procesos Apache inactivos que el número especificado en la directiva `MinSpareServers`, entonces el proceso padre crea un nuevo hijo a una velocidad máxima de 1 por segundo. Sólo es necesario ajustar este parámetro en el caso de sitios realmente visitados. A no ser que sepa lo que hace, no cambie el valor por defecto.

Sintaxis: `MinSpareServers número`

Predefinido: `MinSpareServers 5`

Contexto: configuración del servidor

PidFile

Ver la directiva `PidFile` en la sección "Directivas específicas de MPM threaded".

ScoreBoardFile

Ver la directiva `ScoreBoardFile` en la sección "Directivas específicas de MPM threaded".

SendBufferSize

Ver la directiva `SendBufferSize` en la sección "Directivas específicas de MPM threaded".

StartServers

Ver la directiva `StartServers` en la sección "Directivas específicas de MPM threaded".

User

Ver la directiva `User` en la sección "Directivas específicas de MPM threaded".

5 Módulos Apache

En este capítulo

1. Aprendemos a utilizar los módulos de entorno.
2. Aprendemos a utilizar los módulos de control de acceso y autentificación.
3. Aprendemos a utilizar los módulos de generación de contenido dinámico.
4. Aprendemos a utilizar los módulos de listado de directorios.
5. Aprendemos a utilizar los módulos de tipo de contenido.
6. Aprendemos a utilizar los módulos de generación de contenido dinámico.
7. Aprendemos a utilizar los módulos de cabecera de respuesta.
8. Aprendemos a utilizar los módulos de información y registro de servidores.
9. Aprendemos a utilizar los módulos de integración de URL.
10. Aprendemos a utilizar otros módulos.

Anteriormente, hemos discutido las directivas del módulo general y de los módulos multiproceso (MPM). Apache ofrece muchas más directivas, las cuales

están disponibles en los módulos distribuidos en la fuente estándar. Estos módulos aportan una gran cantidad de funcionalidad mediante la utilización de directivas. Este capítulo discute estos módulos y sus directivas.

Un vistazo a los módulos

En lugar de realizar una lista de todos los módulos por orden alfabético, los he agrupado basándome en su funcionalidad. Estos módulos se dividen en las categorías siguientes:

- **Relacionados con el entorno:** estas directivas le permiten asignar y reajustar las variables de entorno.
- **Control de autentificación y de acceso:** estas directivas le permiten autentificar y autorizar el acceso a usuarios, para restringir partes de su sitio Web.
- **Generación de contenido dinámico:** estas directivas le permiten ejecutar programas externos como los scripts CGI o Server Side Includes para generar contenido dinámico.
- **Configuración del tipo de contenido:** estas directivas le permiten controlar los tipos MIME de los archivos.
- **Listas de directorios:** estas directivas le permiten controlar cómo se formatean estas listas de directorios.
- **Cabecera de la solicitud:** estas directivas le permiten controlar las cabeceras de la solicitud HTTP.
- **Información y registro del servidor:** estas directivas le permiten controlar la información sobre los registros y el estado del servidor.
- **Integración URL:** estas directivas le permiten integrar, reescribir y crear alias para una URL.
- **Módulos diversos:** estas directivas le permiten controlar diversos aspectos de Apache como es el servicio proxy, el módulo WEBDEV, etc.

Módulos relacionados con el entorno

Los módulos de la tabla 5.1, le permiten manipular el entorno que se encuentra disponible para otros módulos o programas externos como los scripts CGI (Common Gateway Interface), SSI (Server-Side Include), scripts `mod_perl`, scripts PHP, servlets Java, y similares.

Tabla 5.1. Módulos relacionados con el entorno

Módulo	Función
`mod_env`	Pasa variables de entorno a programas externos como los scripts CGI y SSI.
`mod_setenvif`	Asigna variables de entorno condicionales utilizando información del lado del cliente.
`mod_unique_id`	Este módulo genera un único ID por solicitud. No tiene directivas. Este módulo está compilado por defecto. Debe configurar la fuente utilizando la opción `--enable-unique-id` con el script de configuración y compilar e instalar Apache.

mod_env

`mod_env` está compilado por defecto. Le permite pasar variables de entorno a programas externos como los scripts CGI, SSI, scripts `mod_perl`, scripts PHP, y similares. `mod-env` tiene las siguientes directivas.

PassEnv

La directiva `PassEnv` le dice al módulo que pase una o más variables de entorno desde el propio entorno del servidor a los scripts CGI y SSI.

Sintaxis: PassEnv *variable* [...]

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva pasa las variables de entorno HOSTTYPE y PATH a programas.

SetEnv

La directiva `SetEnv` asigna un valor determinado a una variable de entorno, que se pasa a scripts CGI/SSI.

Únicamente puede definir un par de variable y el valor por cada directiva `SetEnv`.

Sintaxis: `SetEnv variable valor`

Contexto: Configuración del servidor, host virtual

Por ejemplo, la siguiente directiva `SetEnv` envía la variable CURRENT_CITY a SACRAMENTO:

```
SetEnv CURRENT_CITY SACRAMENTO
```

UnsetEnv

La directiva `UnsetEnv` elimina una o más variables de entorno de las que se han pasado a los scripst CGI/SSI. Se puede utilizar para asegurar que ciertas variables de entorno que están disponibles para el servidor Apache, no lo estén en sus scripts CGI.

Sintaxis: `UnsetEnv variable [...]`

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva `UnsetEnv` elimina la variable CURRENT_STATE de la lista de variables de entorno:

```
UnsetEnv CURRENT_STATE
```

mod_setenvif

El módulo `mod_setenvif` está compilado en Apache por defecto. Le permite crear variables de entorno personalizadas utilizando información de una solicitud HTTP. Puede utilizar esta información en la reescritura de las URL o redirigir a los usuarios a páginas distintas.

BrowserMatch

La directiva `BrowserMatch` asigna y borra la asignación de variables de entorno personalizadas cuando una expresión regular coincide con un patrón encontrado en la cabecera de una solicitud HTTP `User-Agent`. Los clientes Web envían la cabecera `User-Agent` como si fueran navegadores Web, robots Web y similares.

Sintaxis: `BrowserMatch regex variable[=value] [...]`

Contexto: configuración del servidor

Por ejemplo, la siguiente directiva le asigna a una variable llamada `vbscript` el valor `no` si el campo de la cabecera de la solicitud HTTP `User-Agent` contiene la palabra Mozilla, y le asigna el valor 1 a una variable de entorno llamada `javascript` porque no hay ningún valor especificado para esta variable:

```
BrowserMatch ^Mozilla vbscript=no javascript
```

Vamos a ver otro ejemplo:

```
BrowserMatch IE vbscript !javascript
```

En este caso, se elimina la variable `javascript` y se le asigna el valor 1 a la variable `vbscript` si se encuentra la palabra IE en la cabecera de la solicitud HTTP `User-Agent`. El carácter `!` elimina la variable del entorno.

NOTA: Una coincidencia de expresión distingue entre mayúsculas y minúsculas.

BrowserMatchNoCase

La directiva `BrowserMatchNoCase` es la misma que la directiva `BrowserMatch`, excepto en que proporciona coincidencia sin distinguir entre mayúsculas y minúsculas en expresiones regulares.

Sintaxis: `BrowserMatchNoCase regex variable[=value] [...]`

Contexto: configuración del servidor

Por ejemplo, la siguiente directiva busca coincidencias con `MSIE`, `msie`, `Msie` y similares:

```
BrowserMatchNoCase ^MSIE vbscript=yes
```

SetEnvIf

Al igual que las directivas `BrowserMatch` y `BrowserMatchNoCase`, la directiva `SetEnvIf` le permite asignar y borrar variables de entorno personalizadas. En realidad, `BrowserMatch` y `BrowserMatchNoCase` son dos versiones especiales de `SetEnvIf`. Estas dos directivas únicamente pueden funcionar en la expresión regular del campo de la cabecera de la solicitud HTTP `User-Agent`, mientras que `SetEnvIf` se puede utilizar para todos los campos de las cabeceras de la solicitud, al igual que otros tipos de información relacionada con la solicitud, como el nombre del host remoto (`Remote_Host`), la dirección IP remota (`Remote_Addr`), el método de la solicitud (`Request_Method`), el URI solicitado (`Request_URI`), y el referer o la dirección desde la cual un internauta llega a su página (`Referer`).

Sintaxis: `SetEnvIf attribute regex envar[=value] [...]`

Contexto: configuración del servidor

Por ejemplo, la siguiente directiva `SetEnvIf` asigna el valor `true` a la variable `local_user` si la cabecera de la solicitud HTTP `Remote_Host` tiene asignado el valor `yourdomain.com`.

```
SetEnvIf Remote_Host "yourdomain\.com" local_user=true
```

SetEnvIfNoCase

La directiva `SetEnvIfNoCase` es la misma que `SetEnvIf`, excepto en que proporciona coincidencia sin distinguir entre mayúsculas y minúsculas en expresiones regulares.

Sintaxis: `SetEnvIfNoCase attribute regex variable [=value] [...]`

Contexto: configuración del servidor

mod_unique_id

El módulo `mod_unique_id` proporciona un toque mágico para cada solicitud que tenga garantizado ser la única a lo largo de "todas" las solicitudes bajo condiciones muy específicas. El identificador es el único a lo largo de varias máquinas en un grupo de configuración de máquinas adecuado. La variable de entorno `UNIQUE_ID` está asignada al identificador para cada solicitud. No hay directivas para este módulo.

Módulos de control de acceso y autentificación

Apache tiene una serie de módulos para llevar a cabo las tareas de autentificación y autorización.

En la mayor parte de los casos, los módulos de autentificación utilizan autentificación HTTP básica, que utiliza contraseñas de texto simple. El módulo de autorización le permite controlar el acceso a un directorio de su sitio Web mediante un nombre de usuario o una dirección IP. Los módulos que se muestran en la tabla 5.2 le permiten llevar a cabo las tares de control de la autentificación y el acceso.

Tabla 5.2. Módulos de control de acceso y autentificación

Módulo	Función
`mod_auth`	Este es el módulo estándar de autentificación, que implementa la autentificación HTTP `Basic`. Ver los siguientes capítulos para obtener los detalles.
`mod_auth_anon`	Da acceso a los usuarios anónimos para áreas autentificadas.
`mod_auth_dbm`	Ofrece autentificación de usuarios utilizando archivos DBM.
`mod_auth_db`	Ofrece autentificación de usuarios utilizando archivos Berkeley DB.
`mod_auth_digest`	Este módulo implementa la autentificación `Digest` utilizando Message Digest 5 (MD5).

Módulo	Función
mod_access	Este módulo le permite autorizar el acceso utilizando el nombre del host o la dirección IP.

mod_auth_anon

El módulo mod_auth_anon permite el acceso anónimo a áreas autentificadas. Si está familiarizado con los servidores FTP anónimos, se trata de un sistema muy parecido. Todos los usuarios pueden utilizar un ID de usuario denominado "anónimo" y su dirección de correo electrónico como contraseña para entrar. La dirección de correo electrónico introducida se almacena en los archivos de registro y se puede utilizar para seguir la pista de usuarios o para crear listas de correo electrónico de futuros clientes. Tiene que permitir este módulo utilizando la opción `--enable-auth-anon` en el script de configuración de la distribución de la fuente, y compilando e instalando Apache.

Anonymous

Utilizando la directiva `Anonymous` puede especificar uno o más nombres de usuario que se pueden utilizar para acceder al área. Es una buena idea mantener el nombre de usuario "anónimo" en la lista elegida, porque está fuertemente asociado con el acceso anónimo. Si el nombre de usuario que ha elegido tiene algún espacio, asegúrese de que el nombre de usuario está rodeado por comillas.

Sintaxis: `Anonymous user user ...`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Por ejemplo, la siguiente directiva permite a los usuarios introducir `Unregistered User` o `anonymous` como los nombre de usuario que hay que introducir en el área anónima.

```
Anonymous "Unregistered User" anonymous
```

NOTA: La cadena del nombre de usuario no distingue entre mayúsculas y minúsculas.

Anonymous_Authoritative

Cuando se asigna el valor `on`, la autentificación anónima se convierte en el esquema de autentificación completo para un directorio. En otras palabras, si

tiene varios requisitos de autentificación para un directorio y además tiene asignada esta directiva, entonces se ignorarán el resto de los métodos.

Sintaxis: `Anonymous_Authoritative On | Off`

Predefinido: `Anonymous_Authoritative Off`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Anonymous_LogEmail

Cuando asignamos el valor `on` a la directiva `Anonymous_LogEmail`, cualquiera que sea la entrada en el campo de la contraseña de la ventana de autentificación del navegador, se registrará en el archivo de registro de accesos de Apache.

Sintaxis: `Anonymous_LogEmail On | Off`

Predefinido: `Anonymous_LogEmail On`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Anonymous_MustGiveEmail

Cuando asignamos el valor `On`, la directiva `Anonymous_MustGiveEmail` permite que el módulo rechace solicitudes de acceso que no proporcionan las contraseñas en la forma de una dirección de correo electrónico.

Sintaxis: `Anonymous_MustGiveEmail On | Off`

Configuración por defecto: `Anonymous_MustGiveEmail On`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

ADVERTENCIA: No debe confiar en las direcciones de correo electrónico que introduce la gente cuando esta directiva tiene asignado el valor `On`, porque esa no es la forma de verificar quién introduce qué direcciones de correo electrónico.

Anonymous_NoUserID

Si quiere que los usuarios dejen el campo del nombre de usuario de la ventana pop-up vacío, asigne el valor `On` a la directiva `Anonymous_NoUserID`; en

caso contrario, se necesitará un nombre de usuario que coincida con el valor proporcionado en la directiva Anonymous.

Sintaxis: `Anonymous_NoUserID On | Off`

Predefinido: `Anonymous_NoUserID Off`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Anonymous_VerifyEmail

Cuando la directiva `Anonymous_VerifyEmail` está fijada con el valor `on`, necesita que la contraseña sea una dirección válida de correo electrónico. Sin embargo, la verificación de validez es limitada. El módulo únicamente comprueba el símbolo @ y el punto (.) en el campo de la contraseña. Si la contraseña introducida tiene ambos símbolos, es aceptada.

Sintaxis: `Anonymous_VerifyEmail On | Off`

Predefinido: `Anonymous_VerifyEmail Off`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

La siguiente configuración muestra como se pueden utilizar las directivas anteriores para proporcionar acceso anónimo a un directorio.

```
Anonymous_NoUserId off
Anonymous_MustGiveEmail on
Anonymous_VerifyEmail on
Anonymous_LogEmail on
Anonymous anonymous guest "I do not know"
AuthName Use 'anonymous' & Email address for guest entry
AuthType basic
require valid-user
```

mod_auth_dbm

La autentificación basada en texto sencillo (utilizando `mod_auth`) es ineficaz en procesos de alta velocidad y podría afectar negativamente al rendimiento de servidor Web cuando necesitan acceso autenticado una gran cantidad de usuarios (más de 2000) a secciones Web restringidas. El módulo `mod_auth_dbm` es la mejor elección en este tipo de casos. El módulo `mod_auth_dbm` utiliza archivos DBM en lugar de archivos de texto para almacenar datos. Un archivo DBM es un tipo especial de archivo de datos que permite un acceso aleatorio a los datos almacenados más rápido.

NOTA: En realidad si tiene gran cantidad de usuarios, considere la autentificación basada en `mod_auth_mysql` o en `Apache::AuthDBI` que se discute en próximos capítulos. La autentificación basada en DBM se recomienda únicamente si no puede utilizar una base de datos.

Un archivo DBM almacena registros de datos en un par clave=valor y mantiene una tabla de índices computada para las claves en el archivo. Utilizando la tabla de índices en un archivo DBM, es posible recuperar los registros asociados con la clave de en menos tiempo que el necesario para analizar un archivo de texto importante con miles de registros. Hay muchos DBM disponibles, siendo los más comunes GDBM, NDBM, SDBM, y Berkeley DB (BSD-DB). La tabla 5.3 muestra una lista de características para estos DBM.

Tabla 5.3. Características de DBM

Características	NDBM	SDBM	GDBM	BSD-DB
Restricciones de licencia	Desconocido	No	Sí	No
Independiente del orden de los bytes	No	No	No	Sí
Límites de tamaño por defecto	4K	1K	Ninguno	Ninguno
Crea archivos seguros FTP	No	Sí	Sí	Sí
Velocidad	Desconocida	Baja	Media	Rápida
Tamaño de la base de datos	Desconocido	Pequeño	Grande	Medio
Tamaño de código	Desconocido	Pequeño	Grande	Grande
La fuente se encuentra junto con Perl	No	Sí	No	No

Esta tabla está basada en la información encontrada en la documentación de Perl 5. Antes de que pueda utilizar un DBM con Apache, debe asegurarse de que el DBM elegido está instalado en su sistema. Lleve esto a cabo confirmando que la biblioteca de archivos DBM se encuentra localizada en el directorio de bibliotecas por defecto de su sistema. Va a necesitar Perl con soporte DBM. Asegúrese de que tiene la última versión de Perl compilada con el soporte del DBM elegido.

NOTA: Puede bajar Perl de `www.perl.com`. Es muy sencillo configurar Perl para soporte DBM. Simplemente ejecute el script de configuración, y

le indicará el soporte DBM. Por ejemplo, si elige NDBM o GDBM como su DBM, y lo tiene instalado en su sistema, entonces el script de configuración Perl le preguntará si quiere compilar Perl con `-lndbm`, `-lgdbm`, y con los indicadores de bibliotecas.

Una vez que ha instalado las bibliotecas apropiadas de DBM en su sistema, necesita configurar a Apache para que soporte archivos DBM, porque la distribución estándar de Apache no permite el soporte de DBM. Configure el soporte de Apache utilizando la opción `--enable-auth-dbm` en el script de configuración, y compilando e instalando Apache.

NOTA: Si tiene problemas compilando Apache, trate de añadir el `-lyour dbmname` a `EXTRA_LIBS` en el archivo `Configuration`. Por ejemplo, si está utilizando GDBM, puede añadir `-lgdbm` de modo que tenemos `EXTRA_LIBS=-lgdbm`. Asegúrese de que vuelve a ejecutar el script `configure` y que después vuelve a ejecutar `make`. En caso de problemas, puede ser mejor intentarlo con GNU GDBM porque se utiliza en muchos más sistemas y tiene más probabilidad de encontrar ayuda en los grupos de noticias USENET.

Una vez que Apache está compilado adecuadamente para archivos DBM, puede utilizar `dbmmanage` para crear un archivo de usuario DBM. Comience utilizando el script de Perl `dbmmanage` que se encuentra en su directorio de soporte de la distribución estándar de Apache (o la distribución de la fuente) para crear un archivo de usuario basado en DBM. El script de Perl `dbmmanage` puede crear muchos archivos DBM populares como los archivos NDBM, GDBM y Berkley DB.

Este script puede utilizarse para crear un archivo DBM nuevo, para añadir usuarios y contraseñas, para cambiar contraseñas, para borrar usuarios o para ver información sobre usuarios. Antes de utilizar el script, debe modificar la siguiente línea en el script, para que el DBM que quiere utilizar se encuentre el primero de la lista en el array ISA:

```
BEGIN { @AnyDBM_File::ISA = qw(DB_File, NDBM_File, GDBM_file) }
```

Por ejemplo, si está pensando utilizar archivos GDBM, cambie la línea a:

```
BEGIN { @AnyDBM_File::ISA = qw(GDBM_file, DB_File, NDBM_File) }
```

Para determinar qué opciones ofrece el script, ejecútelo del siguiente modo:

```
./dbmmanage
```

Esto le muestra una línea de sintaxis con todas las opciones posibles.

Para crear un nuevo archivo DBM llamado `/www/secrets/myuserdbm` añadiendo un usuario llamado `reader`, introduzca el siguiente comando:

```
./dbmmanage /www/secrets/myuserdbm adduser reader
```

El script le pedirá que introduzca (una reentrada) una contraseña para el usuario `reader`. Una vez que lo ha hecho, añadirá el nombre de usuario y encriptará la contraseña para el archivo DBM `myuserdbm`. No utilice la opción `add` para añadir un usuario, porque no encripta la contraseña. Para ver una lista de usuarios en un archivo DBM, utilice el script siguiente:

```
./dbmmanage /path/to/your/dbmfile view
```

Una vez que tiene recompilado Apache con soporte DBM, puede utilizar el módulo `mod_auth_dbm` para proporcionar autentificación HTTP `Basic` basada en DBM. Observe que para Berkeley DB tiene que utilizar `mod_auth_db` en lugar de `mod_auth_dbm`.

El módulo `mod_auth_dbm` proporciona las directivas `AuthDBMUserFIle`, `AuthDBMGroupFile` y `AuthDBMAuthoritative`. Vamos a ver cada una de estas directivas y algunos ejemplos en los que se utiliza el módulo `mod_auth_dbm`.

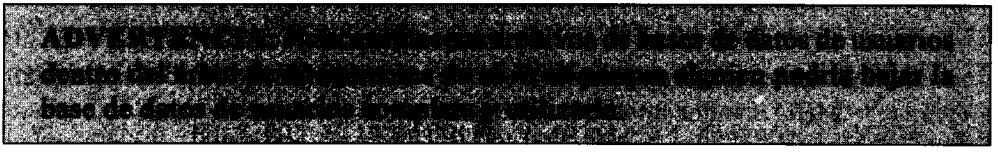
AuthDBMUserFile

La directiva `AuthDBMUserFile` asigna el fully qualified pathname (nombre completo de la ruta de una máquina incluido, su dominio) de un archivo DBM para utilizarlo como el archivo de usuario para la autentificación DBM. El archivo contiene un par clave=valor para cada registro, en el que el nombre de usuario es la clave y la contraseña encriptada `crypt()` es el valor. Observe que cada campo en el registro está separado por dos puntos, y se puede adjuntar un dato arbitrario después del nombre de usuario inicial y los campos de contraseñas.

Sintaxis: `AuthDBMUserFile nombre archivo`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`



AuthDbmGroupFile

La directiva `AuthDbmGroupFile` asigna el fully qualified pathname (nombre completo de la ruta de una máquina incluido su dominio) del grupo de archivos

que contiene la lista de grupos de usuarios. Cada registro en el archivo es un par clave=valor, en el que la clave es el nombre de usuario y el valor es una lista de nombres separados por comas a la que pertenece el usuario.

Sintaxis: `AuthDBMGroupFile nombre archivo`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Si prefiere no utilizar un archivo de grupos separado, puede utilizar un solo archivo DBM que proporcione tanto la información de la contraseña como la del grupo.

El formato del archivo es el siguiente:

```
DBM_Record_Key{username} = encrypted_password:
comma_separated_group_list
```

En este caso, el nombre de usuario es la clave, y la contraseña y la lista de grupos son los dos campos del valor. Puede poner más datos en el archivo DBM tras otros dos puntos, si así lo desea; será ignorado por el módulo de autentificación.

Si utiliza un solo DBM para proporcionar tanto la información del grupo como la de la contraseña, ha de colocar las directivas `AuthDBMGroup` y `AuthDBMUserFile` en el mismo archivo.

AuthDBMAuthoritative

Cuando utilizamos varios esquemas de autentificación como `mod_dbm` y el estándar `mod_auth` en el mismo directorio, puede utilizar la directiva `AuthDbmGroupFile` para determinar si `mod_auth_dbm` es el esquema de autentificación completo.

Sintaxis: `AuthDBMAuthoritative On | Off`

Predefinido: `AuthDBMAuthoritative On`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

El valor por defecto de la directiva, permite que `mod_auth_dbm` se convierta en la autentificación completa para el directorio.

Por eso, si la autentificación basada en DBM falla para un usuario determinado, las credenciales del mismo no se pasan al esquema de autentificación de menor nivel.

Cuando asignamos el valor `off`, las credenciales de una autentificación fallida se pasan al siguiente nivel de autentificación.

TRUCO: Un uso habitual de este módulo es en unión con uno de los módulos básicos `auth`, como el `mod_auth.c`. Mientras que este módulo DBM sustituye el conjunto de verificaciones de credenciales de usuario, unos cuantos (administrador) accesos relacionados caen directamente a un nivel inferior con el archivo `.htpasswd` bien protegido.

A continuación, vamos a ver un ejemplo de cómo puede utilizar un nombre de usuario y una contraseña basados en DBM. Suponiendo que tiene creado el archivo DBM de usuario, está capacitado para restringir el acceso a cualquier directorio Web. En el siguiente ejemplo, supongo que el archivo DBM de usuarios es `/www/secrets/myuserdbm`. Puede añadir el esquema de autentificación al servidor global o virtual utilizando un contenedor `<Directory>`, o puede utilizar el archivo `.htaccess`, no hay diferencia. El ejemplo de configuración es el siguiente:

```
AuthName "Apache Server Bible Readers Only"
AuthType Basic
AuthUserDBMFile  /www/secrets/myuserdbm
require valid-user
```

Ahora Apache utiliza el módulo `mod_auth_dbm` para la autentificación en el directorio en el que se aplica la configuración.

ADVERTENCIA: Asegúrese de que sólo pueden leer el archivo DBM Apache y el dueño. Nadie excepto el dueño puede ser capaz de escribir en él.

mod_auth_db

Si su sistema no es capaz de utilizar DBM, pero está disponible el soporte de Berkeley DB, puede utilizar `mod_auth_db` para utilizar archivos DB en lugar de los módulos DBM de Apache. Este módulo no está compilado en la distribución estándar de Apache.

Antes de configurar Apache con el módulo de autentificación basado en archivos DB, asegúrese de que sabe dónde están almacenados los archivos DB en su sistema. Por ejemplo, en un sistema Linux, los archivos se encuentran en el directorio estándar `/usr/lib`. Si su sistema no tiene las bibliotecas DB, tendrá que obtener el código fuente y compilar primero el soporte DB. Puede encontrar información sobre las bibliotecas DB en `www.sleepycat.com`.

Una vez que se ha asegurado que su sistema tiene bibliotecas DB, puede proceder a la reconfiguración y la recompilación de Apache. Utilice la opción `--enable-db` con el script `configure` para configurar la fuente Apache para

el soporte de Berkeley DB, y entonces compile e instale Apache de la forma habitual.

En este momento, está listo para utilizar el módulo `mod_auth_db`. Este módulo `mod_auth_db` proporciona las directivas `AuthDBUserFile`, `AuthDBGroupFile` y `AuthDBAuthoritative`.

AuthDBUserFile

La directiva `AuthDBUserFile` asigna el fully qualified pathname (nombre completo de la ruta de una máquina incluido su dominio) del usuario del archivo DB que contiene la lista de usuarios y las contraseñas encriptadas.

Sintaxis: `AuthDBUserFile nombre archivo`

Contexto: Directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Al igual que su equivalente DBM, el archivo de usuario DB también tiene como clave el nombre de usuario y el valor es la contraseña encriptada `crypt()`.

ADVERTENCIA: Asegúrese siempre de que sus archivos de usuario están guardados fuera del árbol de documentos Web y sólo lo puede leer Apache. Nadie excepto el dueño (Apache) debería tener acceso a estos archivos.

AuthDBGroupFile

La directiva `AuthDBGroupFile` asigna el fully qualified pathname (nombre completo de la ruta de una máquina incluido su dominio) del archivo de grupo DB, que contiene la lista de grupos de usuarios para la autentificación de usuarios. Al igual que su DBM equivalente, el archivo de grupo utiliza el nombre de usuario como llave y la lista de grupos separados por comas como valor. No debe haber espacios entre el valor, y nunca puede contener los dos puntos.

Sintaxis: `AuthDBGroupFile nombre archivo`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Si prefiere no utilizar un archivo de grupos separados, puede utilizar un solo archivo DB para proporcionar la información de contraseña y de grupo. El formato de este archivo es:

```
DB_File_Key{username} = encrypted_password:
comma_separated_group_list
```

donde el nombre de usuario (username) es la clave, y la contraseña y la lista de grupos son dos campos del valor. Se pueden dejar otros datos en el archivo DB tras dos puntos; el módulo de autentificación los ignorará. Si utiliza un solo DB para proporcionar la información del grupo y de la contraseña, tendrá que colocar las directivas `AuthDBGroup` y `AuthDBUserFile` en el mismo archivo.

AuthDBAuthoritative

Cuando utiliza varios esquemas de autentificación como `mod_db`, `mod_dbm` y el estándar `mod_auth` en el mismo directorio, puede utilizar la directiva `AuthDBAuthoritative` para determinar si `mod_auth_db` es el esquema de autentificación completo.

El valor por defecto de la directiva permite que `mod_auth_db` se convierta en la autentificación completa para el directorio. Por eso, si la autentificación basada en DB falla para un usuario determinado, las credenciales de este usuario no se pasan al esquema de autentificación de nivel inferior. Cuando se asigna el valor `Off`, las credenciales de una autentificación fallida se pasan al siguiente nivel de autentificación.

Sintaxis: `AuthDBAuthoritative On | Off`

Predefinido: `AuthDBAuthoritative On`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Invalidar: `AuthConfig`

Módulos de generación de contenido dinámico

Los módulos que se discuten aquí permiten a Apache ejecutar scripts CGI, SSI, filtros y similares. La tabla 5.4 contiene una lista de estos módulos.

Tabla 5.4. Módulos de generación de contenido dinámico

Módulo	Función
`mod_cgi`	Ejecuta scripts CGI .
`mod_include`	Filtro SSI.
`mod_actions`	Ejecuta scripts CGI basados en los tipos MIME o en el método de la solicitud.
`mod_ext_filter`	Filtra salidas con programas externos.

mod_actions

El módulo `mod_actions` está compilado por defecto. `mod_actions` le permite ejecutar un script de Perl basado en el tipo MIME o en el método de la solicitud HTTP. Ofrece las directivas siguientes.

Action

La directiva `Action` le permite asociar una acción a un tipo MIME determinado. La acción es normalmente un script CGI que procesa el archivo que se ha solicitado. Esto le permite ejecutar un script CGI para un tipo MIME dado.

> **Sintaxis:** `Action MIME_type cgi_script`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `FileInfo`

Por ejemplo, la siguiente directiva hace que Apache ejecute el script específico cada vez que se solicita un archivo HTML:

```
Action text/html /cgi-bin/somescript.pl
```

El script recibe la URL y la ruta del archivo del documento solicitado mediante el `CGI PATH_INFO` estándar y las variables de entorno `PATH_TRANSLATED`. Esto puede ser útil en el desarrollo de los scripts de filtro. Esta sección discute uno de estos scripts de filtro. Cuando se solicita un archivo de texto (`.txt`) mediante la Web, aparece en el navegador Web en un formato muy poco deseable, porque la mayoría de los navegadores no pueden traducir los saltos de línea. Normalmente, la mayor parte de los archivos de texto aparecen como grandes párrafos. Utilizando la directiva `Action`, puede encontrar una solución mucho mejor. Para tener un ejemplo más interesante, vamos a imaginar que quiere desarrollar una solución que no sólo muestre mejor el archivo de texto en el navegador Web sino que, además, inserte un mensaje copyright al final de cada archivo de texto. Para llevar esto a cabo, necesita hacer dos cosas. Primero, añadir la siguiente directiva en el archivo `httpd.conf`:

```
Action plain/text /cgi-bin/textfilter.pl
```

Entonces, desarrollar el script Perl `textfilter` que mostrará el archivo de texto según sus preferencias. El listado 5.1 muestra un script de este tipo. Puede encontrar este script en el CD-ROM.

Listado 5.1. textfilter.pl

```
#!/usr/bin/perl
#
# Script: textfilter.pl
```

```
#
# Función: Este script de filtro convierte archivos de texto
#          en un documento HTML pero mantiene la composición del
#          texto.
#
# Copyright (c) 2001 by Mohammed J. Kabir
#
# Licencia: GPL
#

# El archivo del mensaje copyright se almacena siempre en
# el directorio raíz de documentos del servidor
# y se llama copyright.html.
#
my $copyright_file = $ENV{DOCUMENT_ROOT} . "/copyright.html";
# Obtiene la ruta del documento solicitado
my $path_translated = $ENV{PATH_TRANSLATED};

# Otras variables necesarias para almacenar datos
my $line;
my @text;
my @html;

# Almacena la información de la ruta y el nombre del archivo del
# documento solicitado en un array
@filename = split(/\//,$path_translated);

# Como se utilizan etiquetas HTML para mostrar el archivo de
texto,
# vamos a imprimir la cabecera de contenido text/html.
print "Content-type: text/html\n\n";

# Lee el documento solicitado y almacena los datos
# en la variable array @text
@text = &readFile($path_translated);

# Ahora imprime las siguientes etiquetas de documento HTML.
# Estas etiquetas se enviarán antes que el contenido del
# verdadero documento
#
print <<HEAD;
   <HTML>
   <HEAD> <TITLE>$filename[-1] </TITLE> </HEAD>
   <BODY BGCOLOR="white">
   <BLOCKQUOTE>
   <PRE>
HEAD

# Ahora imprime cada línea almacenada en el array @text
# (es decir, el contenido del documento solicitado)
#
foreach $line (@text) { print $line; }
# Ahora lee el archivo copyright y almacena el contenido
```

```
# en la variable array @html
#
@html = &readFile($copyright_file);
# Imprime cada línea almacenada en el array @html (es decir,
# el contenido del archivo de mensaje copyright)
#
foreach $line (@html){ print $line; }
# Sale del filtro
exit 0;

sub readFile {
    #
    # Subrutina: readFile
    # Función: Lee un archivo si éste existe o sino imprime
    # un mensaje de error y sale del script
    #

    # Obtiene el nombre del nombre del archivo pasado y lo guarda
    # en la variable $file
    my $file = shift;


    # Variable local buffer
     my @buffer;

    # Si existe el archivo, lo abre y lee todas
    # las líneas de la variable array @buffer
     if(-e $file) {

        open(FP,$file) || die "Can not open $file.";
        while(<FP>){
          push(@buffer,$_);
        }

        close(FP);
     } else {
          push(@buffer,"$file is missing.");
     }

    # Devuelve el contenido del buffer.
    return (@buffer);
}
```

El script anterior lee el archivo de texto solicitado e imprime el contenido dentro de unas cuantas etiquetas HTML que permiten que se muestre el contenido tal y como es. Esto se realiza utilizando la etiqueta HTML <PRE>. Una vez que se ha impreso el contenido, el contenido del archivo del mensaje copyright se inserta al final de la salida. Esto permite que se imprima un mensaje copyright con cada archivo de texto solicitado.

La figura 5.1 muestra un ejemplo de salida en la que se muestra un archivo de texto en el navegador Web.

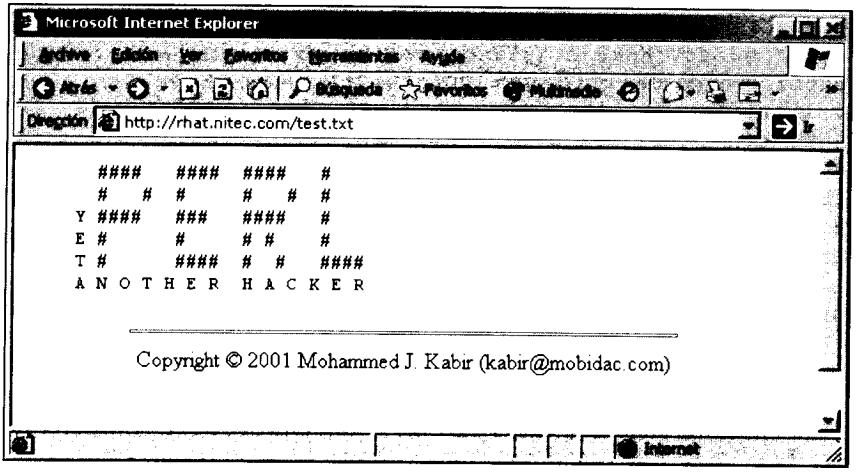


Figura 5.1. Salida del textfilter.pl

Como puede ver, el nombre del archivo solicitado, aparece como el título. El documento se encuentra entre comillas, y se imprime un mensaje de texto personalizado. El archivo del mensaje copyright se almacena en el directorio raíz de documentos. El archivo utilizado en este ejemplo es:

```
</PRE>
<BLOCKQUOTE>
<CENTER>
<HR>
Copyright © 2001 Mohammed J. Kabir (kabir@mobidac.com})
</CENTER>
</BODY>
</HTML>
```

Script

La directiva `Script` es como la directiva `Action`, pero en lugar de asociar una acción con el tipo MIME, lo asocia con una solicitud HTTP como `GET`, `POST`, `PUT` o `DELETE`. El script CGI recibe la URL y la ruta del archivo del documento solicitado utilizando las variables de entorno estándar CGI `PATH_INFO` y `PATH_TRANSLATED`.

Sintaxis: `Script method cgi-script`

Contexto: configuración del servidor, host virtual, directorio

Esta directiva define la acción por defecto. En otras palabras, si tiene definido lo siguiente:

```
Script POST /cgi-bin/deafult_post.pl
```

en un archivo de configuración Apache (como `srm.conf`), entonces, cada vez que se realiza una solicitud mediante el método `POST` HTTP, será procesado

como es habitual, a no ser que tengamos que utilizar la acción por defecto especificada por la directiva.

```
<FORM METHOD="POST">
Enter Name: <INPUT TYPE=TEXT NAME="name" SIZE=25>
<INPUT TYPE=SUBMIT VALUE="Click Here">
</FORM>
```

Si un usuario envía un nombre mediante este formulario, no hay un script CGI especificado para procesar la información, por lo que en este caso, se ejecutará por defecto la acción POST del script `/cgi-bin/default_post.pl`. Sin embargo, si se cambia la etiqueta `<FORM . . .>` a:

```
<FORM ACTION="/cgi-bin/form_processor.pl" METHOD="POST">
```

entonces cada vez que se envía el formulario, se llama al script `/cgi-bin/form_processor.pl` como es habitual. Lo que haga en el script de la acción por defecto es decisión suya. En un sistema de un proveedor de Internet recomiendo hacer que el script por defecto imprima mensajes significativos, para que el usuario del formulario HTML obtenga alguna pista de lo que está haciendo mal.

En caso de una solicitud `GET`, la acción por defecto se utiliza únicamente si la solicitud acompaña datos de consulta. Por ejemplo, `www.yoursite.com/somefile.html` se procesa de la forma habitual, pero si se recibe una solicitud como `http://www.yoursite.com/somefile.html?some=data`, se ejecutará la acción por defecto para `GET`.

mod_ext_filter

El módulo `mod_ext_filter` permite a Apache utilizar un programa externo como filtro de entradas y salidas (input y output). Se puede utilizar cualquier programa que pueda leer una entrada de `STDIN` y cualquiera que pueda escribir una salida en `STDOUT`. Por supuesto, ejecutar un programa externo para procesar una entrada o una salida para cada solicitud es una tarea que consume tiempo y debe evitarse en un entorno de producción. Los filtros se desarrollan mejor utilizando el API (interfaz de aplicación de programas) de Apache y ejecutando dentro del proceso del servidor Apache.

ExtFilterDefine

La directiva `ExtFilterDefine` le permite definir un filtro que el nombre del filtro especificado en esta directiva puede utilizar más tarde.

> **Sintaxis:** `ExtFilterDefine filter_name [mode=input | output] [intype=MIME-type] [outtype=MIME-type] [PreservesContentLength]`
>
> **Contexto:** configuración del servidor

En el siguiente ejemplo, se define un filtro llamado `gzfilter` para que sea el filtro de salida (`mode=output`), que se ejecuta en el programa `/usr/bin/gzip` cuando se le llama:

```
ExtFilterDefine gzfilter mode=output cmd=/usr/bin/gzip
```

Aquí, `mode` sólo puede tener el valor `output`. Las asignaciones `intype` y `outtype` se utilizan para definir el tipo MIME utilizado para la salida y la entrada. Por ejemplo, si un filtro recibe `intype=text/plain` y `outtype=text/html`, entonces el filtro es responsable de traducir el dato del formato texto al formato HTML. El parámetro `PreservesContentLength` se podría utilizar cuando el filtro no cambie el tamaño de los datos en bytes.

Imagine que tiene su propio programa de filtrado llamado `/usr/local/bin/program` y quiere utilizarlo como un filtro output para archivos de texto en un directorio Web llamado `/www/mysite/htdocs/mytxts`. A continuación tenemos una muestra de la configuración que le permite realizarlo:

```
ExtFilterDefine my_test_filter \
                mode=output cmd=/usr/local/bin/program \
                intype=text/plain \
                outtype=text/html

<Directory  "/www/mysite/htdocs/mytxts">

    SetOutputFilter my_test_filter
    AddType text/html

</Directory>
```

En este ejemplo, `ExtFilterDefine` define un filtro llamado `my_filter` que ejecuta `/usr/local/bin/program` cuando es llamado, y toma datos todo texto de `STDIN` y escribe texto/html en `STDOUT`. Ahora el contenedor `<Directory>` asigna este filtro como el filtro output para este directorio y además, le dice a Apache que el tipo output MIME es texto/html utilizando la directiva `AddType`. Si almacena archivos de texto en este directorio y los clientes Web lo solicitan, los archivos se traducirán a HTML utilizando el `/usr/local/bin/program`.

ExtFilterOptions

La directiva `ExtFilterOptions` determina opciones de depuración de errores para el módulo. Cuando se necesita asignar una depuración de errores, utiliza la opción `DebugLevel`.

Fijar esta opción en `0`, que es el valor por defecto, desactiva la depuración de errores. Fijar esta opción en `1` permite las entradas de registro de depuración de errores que muestran las opciones. Fijar esta opción en `9` permite todos y cada uno de los detalles del proceso de filtrado.

Sintaxis: `ExtFilterOptions DebugLevel=n LogStderr | NoLogStderr`

Contexto: servidor

Módulos de configuración de tipo de contenido

Los módulos de esta sección, mostrados en la tabla 5.5, le permiten configurar, detectar y negociar tipos de contenido de la forma apropiada para servir solicitudes.

Tabla 5.5. Módulos Content-Type

Módulo	Función
`mod_mime`	Permite a Apache determinar el tipo MIME utilizando la extensión del archivo.
`mod_mime_magic`	Permite a Apache determinar el tipo MIME utilizando los números mágicos (patrones de bytes).
`mod_negotiation`	Permite a Apache realizar negociación de contenido enviando el mejor tipo de contenido que el cliente puede aceptar.

mod_mime

El módulo `mod_mime` está compilado por defecto en Apache. Proporciona clientes con meta información sobre documentos. También le permite definir un manejador para un documento para determinar el modo en el que Apache procesa el documento.

AddCharset

La directiva `AddCharset` integra una o más extensiones de archivos a un carácter MIME fijo. Esto le permite asociar un carácter fijo a una o más extensiones de archivo.

Sintaxis: `AddCharset charset file_extension [file_extension ...]`

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

Invalidar: `FileInfo`

El siguiente ejemplo da lugar a que un archivo llamado `filename.utf8` se integre como un carácter fijo llamado UTF-8.

```
AddCharset UTF-8        .utf8
```

AddEncoding

La directiva `AddEncoding` integra una o más extensiones a un esquema de codificación MIME.

En otras palabras, esta directiva asocia un esquema de codificación a una o más extensiones de archivo.

> **Sintaxis:** `AddEncoding MIME file_extension [file_extension...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, configuración en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `FileInfo`

Por ejemplo, las siguientes directivas dan lugar a que un archivo llamado `backup.gz` se integre como un archivo x-gzip–encoded, y a que un archivo llamado `tarball.tar` se integre como un archivo `x-tar-encoded`.

```
AddEncoding x-gzip gz
AddEncoding x-tar tar
```

AddHandler

La directiva `AddHandler` define un manejador para una o más extensiones de archivo. Cada vez que Apache encuentra un archivo con un manejador definido, permite que el manejador procese el archivo.

> **Sintaxis:** `AddHandler handler-name file-extension [file-extension ...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

El siguiente ejemplo, la directiva determina que todos los archivos `.cgi` se procesen con un manejador llamado `cgi-script`.

```
AddHandler cgi-script .cgi
```

AddLanguage

La directiva `AddLanguage` integra una lista de extensiones de archivo a un idioma MIME. Cuando Apache encuentra un archivo con esa extensión sabe qué idioma soporta el archivo.

> **Sintaxis:** `AddLanguage MIME_language file_extension [file_extension] [...]`

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

Invalidar: `FileInfo`

El siguiente ejemplo integra todos los archivos con extensiones `.en` o `.english` para ser integrados como archivos English-language. Esto es útil en negociación de contenido, en donde el servidor puede devolver un documento basado en la preferencia de idioma del cliente.

```
AddLanguage en .en .english
```

O, en el ejemplo siguiente, si el cliente prefiere un documento en inglés y están disponibles tanto `document.fr.html` como `document.en.html`, el servidor debería devolver el documento `document.en.html`.

```
AddLanguage en .en
AddLanguage fr .fr
```

AddType

La directiva `AddType` integra una lista de extensiones de archivo al tipo MIME de modo que cuando Apache encuentra archivos con esas extensiones sabe qué tipo MIME utilizar.

Sintaxis: `AddType MIME file_extension [file_extension ...]`

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

Invalidar: `FileInfo`

Por ejemplo, la siguiente línea asocia el tipo MIME llamado `text/html` a las extensiones `htm`, `html`, `HTM` y `HTML`.

```
AddType text/html htm html HTM HTML
```

DefaultLanguage

La directiva `DefaultLanguage` asigna el idioma por defecto.

Sintaxis: `DefaultLanguage MIME_language`

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

Invalidar: `FileInfo`

Por ejemplo, en la siguiente directiva todos los contenidos del directorio `/www/mysite/Japanese` están integrados al idioma por defecto, japonés:

```
<Directory /www/mysite/japanese>
    DefaultLanguage .jp
</Directory>
```

ForceType

La directiva `ForceType` fuerza un tipo determinado de MIME para todos los archivos en un directorio.

El directorio se puede especificar con un contenedor `<Directory>` o un contenedor `<Location>`.

> **Sintaxis:** `ForceType MIME_type`
>
> **Contexto:** directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Por ejemplo, la siguiente directiva fuerza el tipo MIME `text/html` para todos los archivos en el directorio especificado, independientemente de sus extensiones:

```
<Directory /www/nitec/public/htdocs/files/with/no/extensions>
    ForceType text/html
</Directory>
```

SetHandler

La directiva `SetHandler` define un manejador para un directorio o una localización URL. El manejador se utiliza para procesar todos los archivos en el directorio.

> **Sintaxis:** `SetHandler handler_name`
>
> **Contexto:** directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Por ejemplo, la siguiente directiva fuerza a que todos los archivos de la localización `/bin` sean tratados como scripts CGI, que son manejados por el manejador `cgi-bin`:

```
<Location /bin>
    Options ExecCGI
    SetHandler cgi-bin
</Location>
```

RemoveHandler

La directiva `RemoveHandler` deshace un manejador para un directorio o para una localización URL. Es útil para limitar `SetHandler`, que normalmente se aplica a todos los archivos en el directorio.

Utilizando `RemoveHandler` puede eliminar manejadores para algunos archivos o, incluso, un subdirectorio.

Sintaxis: `RemoveHandler handler_name`

Contexto: directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)

Por ejemplo, en la siguiente directiva, el manejador `my-handler` se fija con una extensión `.mjk` fuera del directorio `/www/mysite/htdocs/special`, por lo que, automáticamente, se aplica también a este directorio. Sin embargo, como `RemoveHandler` se aplica a este directorio para deshacer la asociación entre `my-handler` y `.mjk`, los archivos con extensiones `.mjk` en este directorio no se manejan con `my-handler`:

```
SetHandler my-handler .mjk

<Directory /www/mysite/htdocs/special>
   RemoveHandler .mjk
</Location>
```

TypesConfig

La directiva `TypesConfig` determina el archivo de configuración MIME por defecto. El valor por defecto debería ser adecuado para la mayoría de las instalaciones Apache. Si quiere añadir sus propios tipos MIME, utilice la directiva `AddType` en lugar de modificar este archivo.

Sintaxis: `TypesConfig nombre archivo`

Predefinido: `TypesConfig conf/mime.types`

Contexto: configuración del servidor

NOTA: Si necesita soporte adicional para manejar tipos MIME, tendrá que dirigirse al módulo `mod_mime_magic` de la sección siguiente. Para la mayor parte de las instalaciones Apache esto no es necesario, por lo que no se discute en este libro.

mod_mime_magic

El módulo `mod_mime_magic` le permite a Apache determinar el tipo de un archivo MIME comparando unos cuantos bytes del archivo con un valor mágico almacenado en un archivo. Este módulo sólo se necesita cuando `mod_mime` no puede adivinar el tipo MIME de un archivo. En la mayoría de los casos, no necesita este módulo. Este módulo tiene una directiva llamada `MimeMagicFile`.

Esta directiva permite el módulo `mod_mime_magic` y señala el archivo mágico necesario para este módulo. La distribución Apache contiene un archivo mágico en el subdirectorio `conf`, por lo que si quiere utilizar este módulo, fije esta directiva en `conf/magic`.

Sintaxis: `MimeMagicFile magic_file_filename`

Contexto: configuración del servidor, host virtual

mod_negotiation

El módulo `mod_negotiation` está compilado por defecto. Proporciona soporte para negociaciones de contenido. En un escenario típico de negociación de contenido, el cliente proporciona información sobre el tipo de contenido que puede manejar, y el servidor intenta proporcionar el contenido más apropiado. El servidor lleva esto a cabo con la ayuda de integración de tipos y el `MultiViews`.

Un tipo `map` proporciona una descripción de documentos. Cada descripción de documentos contiene una o más cabeceras. Puede contener también líneas de comentarios que comienzan con un carácter almohadilla (#). Las descripciones de documentos están separadas por líneas en blanco. Las cabeceras de descripción de documentos son:

- **Content-Encoding:** especifica el tipo de codificación del archivo. Únicamente están permitidas la codificación `x-compress` y `x-gzip`.
- **Content-Language:** el idioma del documento.
- **Content-Length:** la longitud del archivo en bytes.
- **Content-Type:** el tipo MIME del documento. Están permitidos parámetros opcionales clave-valor. Los parámetros permitidos son `level`, que proporciona el número de versión (como un integer) del tipo MIME, y `qs`, que indica la cualidad (como un número decimal floating) del documento.
- **URI:** la ruta del documento referida al archivo de integración.

El buscador `MultiViews` trata de determinar la coincidencia más cercana para el documento perdido utilizando la información que tiene del cliente, y devuelve esta correspondencia si es posible. Cuando permite la opción `MultiViews` en la directiva `Options`, el servidor es capaz de realizar la búsqueda `MultiViews` cuando no se encuentra el documento solicitado. Este módulo proporciona las dos directivas siguientes.

CacheNegotiatedDocs

La directiva `CacheNegotiatedDocs` permite que se almacenen en el caché los documentos de contenido negociado en servidor proxy. Tenga en cuenta que la nueva especificación HTTP 1.1 proporciona mucho más control para cachear los documentos negociados, y `CacheNegotiatedDocs` no tiene efecto sobre las respuestas a las solicitudes HTTP 1.1. Esta directiva ha desaparecido prácticamente desde la aparición de HTTP 1.1. No se recomienda la utilización de `CacheNegotiatedDocs`.

Sintaxis: `CacheNegotiatedDocs`

Contexto: configuración del servidor

LanguagePriority

La directiva `LanguagePriority` determina el idioma que el servidor va a preferir en un escenario de búsqueda `MultiViews`, cuando el cliente no proporciona ninguna información sobre sus preferencias de idioma.

Sintaxis: `LanguagePriority MIME_language [MIME_language ...]`

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

Invalidar: `FileInfo`

En la siguiente directiva, por ejemplo, si está activada la opción `MultiViews` y el cliente no proporciona ninguna información sobre sus preferencias de idioma para un archivo perdido, el servidor primero sirve la versión en inglés de la coincidencia más cercana, y sino la francesa, y así sucesivamente. Al igual que la directiva `CacheNegotiatedDocs`, esta directiva no es efectiva en el entorno HTTP 1.1.

```
LanguagePriority en fr de
```

Módulos de listas de directorios

Si tiene un directorio dentro de su árbol de documentos Web que no tiene un archivo con el índice de directorios (asignado utilizando la directiva `DirectoryIndex`) entonces Apache generará automáticamente una lista de directorios, en el caso de que no haya desactivado la lista de directorios automática utilizando la directiva `Options -Indexes`. Apache le permite personalizar la lista de directorios generada automáticamente. Los módulos de esta sección, que se muestran en la tabla 5.6, le permiten configurar el modo en el que se muestran las listas de directorios.

Tabla 5.6. Módulos de listas de directorios

Módulo	Función
`mod_dir`	Manejador básico de directorios.
`mod_autoindex`	Lista automática de directorios.

mod_dir

El módulo `mod_dir` está compilado en Apache por defecto. Utilizando este módulo, Apache puede redirigir cualquier solicitud que no incluya ninguna barra final. Por ejemplo, este módulo puede redirigir `www.yoursite.com/somedirectory` a `www.yoursite.com/somedirectory/`. Además proporciona la directiva `DirectoryIndex` para ayudar con la realización de un índice del contenido del directorio.

La directiva `DirectoryIndex` especifica el nombre(s) de los archivos que Apache debería buscar antes de crear un índice dinámico de directorios. Los archivos pueden ser de cualquier tipo, desde un archivo HTML hasta un script CGI. El valor por defecto permite a Apache buscar el archivo `index.html` para cualquier solicitud que termine con el nombre de un directorio.

> **Sintaxis:** `DirectoryIndex local_URL [local_URL ...]`
>
> **Predefinido:** `DirectoryIndex index.html`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `Indexes`

Por ejemplo, `www.yoursite.com/some/directory/` hace que Apache busque un archivo llamado `/some/directory/index.html`. Si el archivo existe; su contenido es enviado al cliente. En ausencia de este archivo, Apache crea una lista dinámica de directorios.

Puede especificar por defecto uno o más archivos, como el archivo de índice de directorios. En el siguiente ejemplo, se le dice a Apache que busque todos los archivos nombrados para cada solicitud de un directorio:

```
DirectoryIndex index.html index.htm welcome.html welcome.htm
```

Tenga en cuenta que Apache buscará los archivos en el mismo orden (de izquierda a derecha) en el que aparecen en la configuración anterior. En otras palabras, si Apache encuentra `index.html`, no buscará `index.htm`, `welcome.html` o `welcome.htm`. Puede especificar un nombre de script CGI script como el índice por defecto. Por ejemplo, la siguiente directiva consigue que Apache ejecute el `script /cgi-bin/show_index.cgi` cada vez que Apache obtiene la solicitud de un directorio:

```
DirectoryIndex /cgi-bin/show_index.cgi
```

mod_autoindex

El módulo `mod_autoindex` está compilado en Apache por defecto. Cuando Apache recibe una solicitud para un directorio, busca uno o más archivos de

índices de directorios especificados por la directiva `DirectoryIndex`. Normalmente este archivo es `index.html` o `index.htm`. En ausencia de ese archivo de índices, sin embargo, Apache puede generar una lista dinámica de directorios. Este módulo le permite controlar el modo en el que Apache crea la lista dinámica de directorios.

Apache genera dos tipos de índices dinámicos de directorios: simple y personalizada. El índice personalizado y otras muchas opciones de índices están disponibles para este módulo. Las directivas para `mod_authoindex` son las siguientes.

AddAlt

Cuando está activada `FancyIndexing`, esta directiva determina el texto especificado como una alternativa al icono que se muestra para uno o más archivos o extensiones de archivos especificados como argumentos. Esto se realiza para navegadores no gráficos como Linx.

> **Sintaxis:** `AddAlt "text" nombrearchivo [nombrearchivo ...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `Indexes`

Por ejemplo, la directiva siguiente le permite a Apache mostrar el texto alternativo "`Pictures`" en lugar del icono para cada tipo de archivo gráfico especificado aquí. Para los navegadores gráficos como Netscape Navigator o Internet Explorer, el texto alternativo se muestra como texto de ayuda bajo las populares plataformas Windows. En estos sistemas, los usuarios pueden obtener un aviso o ayuda sobre el archivo cuando pasan el ratón por encima del icono que representa a alguno de los tipos de archivo:

```
AddAlt "Pictures" gif jpeg jpg bmp
```

AddAltByEncoding

Si no quiere asignar un texto alternativo a los nombres de archivo o a las extensiones de los archivos mediante la directiva `AddAlt`, puede utilizar la directiva `AddAltByEncoding` para asignar ese texto para una o más codificaciones MIME. Al igual que `AddAlt`, esta directiva sólo se puede utilizar cuando está activada `FancyIndexing`.

> **Sintaxis:** `AddAltByEncoding "text" MIME_encoding [MIME_encoding ...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `Indexes`

Por ejemplo, la siguiente directiva hace que Apache muestre el texto alternativo "Compressed File" ("Archivo comprimido") para todos los archivos de tipo MIME x-compress.

```
AddAltByEncoding "Compressed File" x-compress
```

AddAltByType

Al igual que la directiva AddAltByEncoding, la directiva AddAltByType asigna texto alternativo para un archivo, en lugar de un icono para FancyIndexing. Sin embargo, utiliza un tipo MIME en lugar de codificación MIME.

Sintaxis: AddAltByType "text" MIME-type [MIME_type ...]

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

Invalidar: Indexes

Por ejemplo, la siguiente directiva muestra el texto "HTML FILE" en lugar del icono, en los navegadores no gráficos. En el caso de navegadores gráficos, este texto aparecerá como aviso o ayuda:

```
AddAltByType "HTML FILE" text/html
```

AddDescription

La directiva AddDescription asigna un texto descriptivo para un nombre de archivo, a un nombre de archivo parcial o a un nombre de archivo comodín cuando está activada FancyIndexing.

Sintaxis: AddDescription "text" file [file ...]

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

Invalidar: Indexes

Por ejemplo, la siguiente directiva muestra la descripción para todos los archivos GIF, JPEG, JPG y BMP generados en la lista de directorios:

```
AddDescription "Graphics File" *.gif *.jpeg *.jpg *.bmp
```

AddIcon

La directiva AddIcon le permite asignar iconos a nombres de archivos y de directorios que se muestran para FancyIndexing.

Sintaxis: AddIcon icono nombre archivo [nombrearchivo ...]

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

Invalidar: `Indexes`

Por ejemplo, la siguiente directiva le dice a Apache que muestre `/icons/picture.gif` cerca de los archivos que tengan extensiones `.gif`, `.jpg` y `.bmp`:

```
AddIcon /icons/picture.gif .gif .jpg .bmp
```

Si además, quiere proporcionar texto alternativo para la extensión de los archivos de la lista, puede utilizar un formato del siguiente tipo, en donde `IMG` es el texto alternativo que se muestra en los navegadores no gráficos:

```
AddIcon (IMG, /icons/picture.gif) .gif .jpg .bmp
```

Si quiere mostrar un icono para un directorio, puede utilizar la directiva del siguiente modo:

```
AddIcon /path/to/your/directory/icon      ^^DIRECTORY^^
```

De igual manera, si quiere mostrar un icono para cada línea en blanco mostrada por el esquema de índice personalizada, puede utilizar:

```
AddIcon /path/to/your/blank/line/icon     ^^BLANKICON^^
```

AddIconByEncoding

La directiva AddIconByEncoding le deja asignar iconos a las codificaciones MIME. En otras palabras, puede asignar una imagen de un icono a un tipo MIME.

Sintaxis: `AddIconByEncoding icon_file MIME_encoding [MIME_encoding...]`

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

Invalidar: `Indexes`

Por ejemplo, la siguiente directiva `AddIconByEncoding` le dice a Apache que muestre el icono `/icons/zip.gif` en todos los archivos que sean del tipo MIME `x-gzip` (por ejemplo los que tienen la extensión `.gz`).

```
AddIconByEncoding  /icons/zip.gif        x-gzip
```

AddIconByType

La directiva `AddIconByType` también le permite asignar iconos a uno o más tipos MIME.

Sintaxis: `AddIconByType icon_file MIME_type [MIME_type...]`

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

Invalidar: `Indexes`

Por ejemplo, la siguiente directiva `AddIconByType` le dice a Apache que muestre `/icons/html.gif` para todos los archivos texto/html.

```
AddIconByType (HTML,/icons/html.gif) text/html
```

DefaultIcon

Cuando `AddIcon`, `AddIconByEncoding` o `AddIconByType` no encuentran asociación para un determinado archivo, se puede mostrar un icono por defecto. La directiva `DefaultIcon` le permite asignar ese icono.

Sintaxis: `DefaultIcon URL`

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

Invalidar: `Indexes`

Por ejemplo, la siguiente directiva muestra `idontknow.gif` como icono de cualquier archivo cuya asociación sea desconocida:

```
DefaultIcon /icon/idontknow.gif
```

FancyIndexing

La directiva `FancyIndexing` le permite activar y desactivar un índice personalizado de directorios. Puede obtener el mismo efecto con la directiva `IndexOptions`.

Sintaxis: `FancyIndexing On | Off`

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

Invalidar: `Indexes`

HeaderName

Si utiliza `FancyIndexing`, puede insertar contenido de un archivo al principio de la lista de índices. La directiva `HeaderName` le permite especificar el nombre del archivo para tal inserción.

Sintaxis: `HeaderName nombre archivo`

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

Invalidar: `Indexes`

Por ejemplo, la siguiente directiva le dice a Apache que busque un archivo llamado `welcome` o `welcome.html` en el directorio de la lista; si se encuentra ese archivo, se inserta el contenido antes de la lista real:

```
HeaderName welcome
```

IndexIgnore

Si necesita que sean visibles algunos archivos o extensiones de archivos en la lista de directorios, puede utilizar la directiva `IndexIgnore`.

> **Sintaxis:** `IndexIgnore file [file...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `Indexes`

Por ejemplo, la siguiente directiva garantiza que Apache no mete en la lista de directorios los archivos `welcome`, `welcome.html` o los archivos de configuración del ámbito de directorios (`.htaccess`):

```
IndexIgnore welcome welcome.html.htaccess
```

El carácter . (punto) se encuentra automáticamente en la lista `IndexIgnore`; por eso, los archivos que comienzan por este carácter no se encuentran en la lista. Sin embargo, puede preferir añadir configuración del ámbito de directorios (`.htaccess`) en la lista, para sentirse más seguro.

IndexOptions

La directiva `IndexOptions` especifica el comportamiento de la generación automática de un índice de directorios.

> **Sintaxis:** `IndexOptions option [option] [...]`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorios (`.htaccess`)
>
> **Invalidar:** `Indexes`

La tabla 5.7 muestra las opciones que puede utilizar con `IndexOptions`.

Tabla 5.7. Opciones para IndexOptions

Opción	Lo que hace
`FancyIndexing`	Activa los índices de directorios personalizados. Tenga en cuenta que las directivas `FancyIndexing` e `IndexOptions` se invalidan la una a la otra.

Opción	Lo que hace
IconHeight[=pixels]	Permite a Apache incluir el atributo HEIGHT=pixels en la etiqueta IMG del icono, lo que hace que la carga del icono sea más rápida en la mayoría de los navegadores. Si no especifica el tamaño en píxeles, se utiliza un estándar por defecto.
IconsAreLinks	Hace que los iconos formen parte del anchor para el nombre de archivo, en los índices personalizados.
IconWidth[=pixels]	Permite a Apache incluir el atributo WIDTH=pixels en la etiqueta IMG del icono, lo que hace que la carga del icono sea más rápida en la mayoría de los navegadores. Si no especifica el tamaño en píxeles, se utiliza un estándar por defecto.
ScanHTMLTitles	Si quiere que Apache lea el título (indicado por el par de etiquetas <TITLE> y </TITLE>) de un documento HTML para un índice personalizado, utilice esta opción. Si tiene ya especificada una descripción utilizando la directiva AddDescription, sin embargo, esta opción no se utiliza. Tenga en cuenta que leer cada contenido de los archivos y buscar el título es una tarea que gasta mucho tiempo y que hará que se ralentice el envío de la lista de directorios. No recomiendo esta opción.
SuppressColumnSorting	Por defecto, Apache permite pinchar en los encabezados de las columnas de los índices de directorios personalizados, lo que permite a los usuarios ordenar información en esa columna. Esta opción inactiva esta característica.
SuppressDescription	Si no quiere mostrar descriptores de archivos en la lista de directorios personalizada, utilice esta opción.
SuppressHTMLPreamble	Si el directorio realmente contiene un archivo especificado por la directiva HeaderName, el módulo normalmente incluye el contenido del archivo después de un preámbulo HTML estándar (<HTML>, <HEAD>, etc.). La opción SuppressHTMLPreamble desactiva este comportamiento.

Opción	Lo que hace
SuppressLastModified	Suprime el despliegue de la última fecha de modificación en las listas de directorios personalizada.
SuppressSize	Suprime el tamaño del archivo en una lista de índices personalizada.

IndexOrderDefault

La directiva IndexOrderDefault le permite cambiar la vista de la lista de directorios ordenando varios campos como nombre, fecha, tamaño y descripción en los directorios que se han mostrado utilizando la característica FancyIndexing.

Sintaxis: IndexOrderDefault Ascending | Descending Name | Date | Size | Description

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

Invalidar: Indexes

ReadmeName

Si quiere insertar un archivo al final de la lista de directorios personalizada, utilice la directiva ReadmeName.

Sintaxis: ReadmeName nombrearchivo

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

Invalidar: Indexes

Por ejemplo, la siguiente directiva hace que Apache busque un archivo llamado readme.html o readme para insertarlo al final de la lista:

```
ReadmeName readme
```

Response Header Modules

Apache le permite enviar cabeceras de respuestas HTTP al cliente cuando envía datos. Los módulos de esta sección, mostrados en la tabla 5.8, le permiten configurar varias cabeceras de respuesta.

Tabla 5.8. Response-Header Modules

Módulo	Función
mod_asis	Envía archivos que contiene sus propias cabeceras HTTP.
mod_headers	Añade cabeceras HTTP de forma arbitraria a los recursos.
mod_expires	Aplica Expires: cabeceras a los recursos.
mod_cern_meta	Soporte para meta archivos con cabeceras HTTP.

mod_asis

El módulo mod_asis está compilado por defecto. Este módulo le permite enviar un documento tal cual es, en otras palabras, el documento se envía al cliente sin cabeceras HTTP. Esto puede ser útil cuando redirigimos clientes sin ayuda de ningún script. Para enviar un archivo tal cual es, necesita asegurarse de que el archivo httpd.conf contiene una entrada del siguiente tipo:

```
AddType httpd/send-as-is asis
```

Esto asigna el tipo MIME httpd/send-as-is a la extensión de archivo .asis. Si crea un archivo llamado foobar.asis y un cliente lo solicita, el archivo se envía al cliente sin ninguna cabecera HTTP. Es su trabajo incluir las cabeceras apropiadas en el archivo. Por ejemplo, si quiere proporcionar un mecanismo de redirección mediante los archivos .asis, puede crear archivos con cabeceras del tipo:

```
Status: 301 Text Message
Location: new-URL
Content-type: text/html
```

El listado 5.2 muestra un archivo llamado redirect.asis, que redirecciona al cliente a una nueva localización.

Listado 5.2. redirect.asis

```
Status: 301 We have moved.
Location: http://www.our-new-site/
Content-type: text/html
<H1>Notice to Visitors</H1>
Please update your bookmark to point to <A HREF="http://
www.our-new-site/ "> www.our-new-site/ </A><BR>
<BR>
Thanks.
```

Cuando el cliente solicita este archivo, el mensaje de estado 301 le dice que utilice la información de localización para redireccionar la solicitud. No tiene que añadir las cabeceras `Date:` y `Server:`, porque el servidor las añade automáticamente. Sin embargo, el servidor no proporciona una cabecera `Last-Modified`.

mod_headers

Este módulo no está compilado por defecto. `mod_headers` le permite manipular las cabeceras de respuesta HTTP, y proporciona una sola directiva llamada `Header`, que le permite manipular la cabecera de respuesta HTTP.

Sintaxis: `Header action header value`

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

Invalidar: `FileInfo`

Las acciones permitidas son:

Acción	Lo que hace
`Set`	Fija una cabecera. Si existe una cabecera antigua con el mismo nombre, su valor se cambia por uno nuevo.
`Add`	Añade una cabecera. Puede dar lugar a varias cabeceras con el mismo nombre cuando existen una o más cabeceras con el mismo nombre.
`Append`	Adjunta el valor de una cabecera que ya existe.
`Unset`	Elimina la cabecera.

Por ejemplo, la siguiente directiva añade la cabecera `Author` con el valor `"Mohammed J. Kabir"`:

```
Header add Author "Mohammed J. Kabir"
```

Y la siguiente línea elimina la misma cabecera:

```
Header unset Author
```

mod_expires

El módulo `mod_expires` no está compilado en Apache por defecto. Le permite determinar el modo en el que Apache trata con cabeceras Expires HTTP en

la respuesta del servidor a las solicitudes. Las cabeceras Expires HTTP le proporcionan una forma de hablarle al cliente sobre el tiempo que tardan los recursos solicitados en inactivarse. Esto resulta útil cuando los documentos están en la memoria caché del cliente y ha de solicitarlos de nuevo. La mayor parte de los clientes determinan la validez de un documento solicitado investigando el tiempo de expiración de los documentos en el caché proporcionado por las cabeceras Expires HTTP. Este módulo le permite controlar las asignaciones de las cabeceras Expires HTTP.

ExpiresActive

La directiva `ExpiresActive` activa o inactiva la generación de la cabecera Expires. No garantiza que una cabecera Expires se genere. Si no se encuentra el criterio, no se envía la cabecera.

Sintaxis: `ExpiresActive On | Off`

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

Invalidar: `Indexes`

ExpiresByType

La directiva `ExpiresByType` especifica el valor de la cabecera Expires HTTP para documentos de un tipo específico de `MIME-type`. El tiempo de expiración se determina en segundos. Puede definir el tiempo de dos formas. Si elige utilizar el formato `Mseconds` para marcar el tiempo de expiración, se utiliza el momento de la última modificación como base. En otras palabras, M3600 significa que quiere que el archivo expire una hora después de su modificación. Por otro lado, si utiliza el formato `Aseconds`, entonces el momento de acceso de clientes se utiliza como base de tiempo. A continuación tenemos algunos ejemplos.

Sintaxis 1: `ExpiresByType MIME_type Mseconds | Aseconds`

Sintaxis 2: `ExpiresByType MIME-type "base_time [plus] num Years|Months|Weeks|Days|Hours|Minutes|Seconds"`

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

Invalidar: `Indexes`

Las siguientes directivas hacen que todos los archivos de texto expiren una hora después de estar en el caché del cliente:

```
ExpiresByType text/plain A3600
```

Y las siguientes consiguen que todos los archivos GIF expiren tras una semana de su última modificación:

```
ExpiresByType image/gif M604800
```

Si quiere utilizar la segunda sintaxis para especificar el tiempo de expiración, necesita determinar el valor apropiado de tiempo base utilizando las siguientes opciones:

Valor	Lo que significa
`Access`	Momento en el que el cliente accedió al archivo.
`Now`	Momento actual. Es lo mismo que el momento de acceso.
`Modification`	Momento en el que el archivo fue cambiado por última vez.

Por ejemplo, las siguientes directivas le dicen a Apache que envíe cabeceras para decirle al navegador que los documentos HTML expirarán tras siete días desde el momento de su acceso y que las imágenes GIF expirarán después de cualquier cambio en el archivo o tres horas y diez minutos después.

```
ExpiresByType text/html "access plus 7 days"
ExpiresByType image/gif "modification plus 3 hours 10 minutes"
```

ExpiresDefault

La directiva `ExpiresDefault` asigna el momento de expiración por defecto para todos los documentos en el que contexto en que se encuentran especificados. Por ejemplo, si esta directiva está especificada en el host virtual, sólo se aplicará a los documentos accesibles mediante el host virtual. Del mismo modo, puede especificar esta directiva en un contexto en el ámbito de directorios, lo que permitirá que todos los documentos en ese directorio expiren en un intervalo específico. Ver `ExpiresByType` para obtener los detalles de sintaxis.

Sintaxis 1: `ExpiresDefault Mseconds | Aseconds`

Sintaxis 2: `ExpiresDefault "base_time [plus] num Years|Months|Weeks|Days|Hours|Minutes|Seconds"`

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

Invalidar: `Indexes`

A continuación tenemos dos ejemplos:

```
ExpiresDefault M3600
ExpiresDefault "access plus 2 days"
```

El primer ejemplo marca el momento de expiración en una hora después de la última modificación de los documentos. El segundo marca el momento de expiración en dos días después de que el cliente acceda.

mod_cern_meta

El módulo `mod_cern_meta` no está compilado por defecto. Proporciona soporte para meta información. Esta información puede consistir en cabeceras HTTP adicionales como:

```
Expires: Saturday, 19-May-01 12:00:00 GMT
```

o puede ser cualquier otro tipo de información como la que tenemos a continuación, en la que la meta información se almacena en un archivo y aparece junto con cabecera de la respuesta HTTP:

```
Foo=Bar
```

MetaFiles

La directiva `MetaFiles` activa o desactiva el procesamiento de los archivos de meta cabecera.

Sintaxis: `MetaFiles On | Off`

Predefinido: `MetaFiles Off`

Contexto: archivo de control de acceso en el ámbito de directorio (`.htaccess`)

MetaDir

La directiva `MetaDir` especifica el nombre del directorio que almacena los archivos meta cabecera. Por ejemplo, si tiene un directorio llamado `/www/mycompany/public/htdocs` y quiere almacenar archivos meta cabecera para ese directorio, necesitaría crear un subdirectorio llamado `.web` si utiliza el valor por defecto de la directiva `MetaDir`. El directorio `.web` almacena archivos meta cabecera.

Sintaxis: `MetaDir directorio`

Predefinido: `MetaDir .web`

Contexto: archivo de control de acceso en el ámbito de directorio (`.htaccess`)

MetaSuffix

La directiva `MetaSuffix` especifica la extensión del nombre de archivo para los archivos de meta información. Por ejemplo, si tiene un archivo HTML llamado `mypage.html`, entonces necesita crear `mypage.html.meta` (utilizando el valor por defecto de esta directiva) para almacenar sus meta cabeceras. El archivo `mypage.html.meta` debe residir en el directorio especificado por la directiva `MetaDir`.

> **Sintaxis:** `MetaSuffix suffix`
>
> **Predefinido:** `MetaSuffix .meta`
>
> **Contexto:** archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Para permitir a Apache que envíe meta información sobre un directorio llamado `/www/mycompany/public/htdocs`, necesita hacer lo siguiente:

1. Fijar la directiva `MetaFiles` en on en el archivo de configuración en el ámbito de directorio (`.htaccess`) para `/www/mycompany/public/htdocs`. También puede asignar las directivas `MetaDir` y `MetaSuffix` en este archivo.
2. Crear un subdirectorio llamado `.web` (suponiendo que está utilizando el valor por defecto de la directiva `MetaDir`).
3. Crear un archivo de texto con la extensión `.meta` (suponiendo que está utilizando el valor por defecto de la directiva `MetaSuffix`).
4. Poner todas las cabeceras HTTP que quiera suministrar en esta archivo.

Por ejemplo, para proporcionar meta cabeceras para un archivo llamado `/www/mycompany/public/htdocs/mypage.html`, necesita crear un archivo llamado `/www/mycompany/public/htdocs/.web/mypage.html.meta`. Este archivo puede incluir líneas como estas:

```
Expires: Saturday, 19-May-01 12:00:00 GMT
Anything=Whatever
```

Módulos de información de servidores y de registro

Lo módulos de esta sección, mostrados en la tabla 5.9, le permiten registrar accesos, informar sobre el estado del servidor y dar información de configuración, e incluso, seguir la pista de usuarios que están utilizando cookies.

Tabla 5.9. Módulos de información de servidores y de registro

Módulo	Función
`mod_log_config`	Proporciona registros de acceso personalizados.
`mod_status`	Muestra información sobre el estado.
`mod_info`	Muestra información sobre la configuración del servidor.
`mod_usertrack`	Proporciona un seguimiento de clientes utilizando Cookies http.

mod_log_config

Este módulo se discute en detalle en el capítulo 8. Ver la sección "Crear archivos de registro" en ese capítulo para aprender más cosas sobre este módulo y sus directivas.

mod_status

Este módulo se discute en detalle en el capítulo 8. Ver la sección "Permitir páginas de estado con `mod_status`" en ese capítulo para aprender más cosas sobre este módulo y sus directivas.

mod_info

Este módulo se discute en detalle en el capítulo 8. Ver la sección "Acceder a la configuración de acceso con `mod_info`" en ese capítulo para aprender más cosas sobre este módulo y sus directivas.

mod_usertrack

Este módulo se discute en detalle en el capítulo 8. Ver la sección "Registrar cookies" en ese capítulo para aprender más cosas sobre este módulo y sus directivas.

Módulos de integración URL

Los módulos en esta sección, mostrados en la tabla 5.10, le permiten integrar distintas URL a directorios físicos determinados, crear reglas complejas de reescritura, crear alias y automatizar las URL de host virtuales a integraciones de directorios físicos.

Tabla 5.10. Módulos de integración de URL

Módulo	Función
`mod_userdir`	Le permite acceder a sitios Web personales almacenados en directorios locales del usuario.
`mod_rewrite`	La reglas de reescritura de URL se crean utilizando este módulo. Ver el capítulo 9 para obtener los detalles.
`mod_alias`	Para integrar distintas partes del sistema de archivos del host en el árbol de documentos y para redireccionar URL.
`mod_speling`	Corrección automática de pequeños errores en las URL.
`mod_vhost_alias`	Soporte de configuración dinámica en masa de alojamientos virtuales.

mod_userdir

El módulo `mod_userdir` le permite a Apache hacer accesibles directorios Web específicos de usuario mediante `http://your_server_name/~username`. Si no tiene pensado el soporte de ese tipo de sitios Web, no necesita este módulo. La directiva `UserDir` (la única directiva de este módulo) le permite asignar el directorio que Apache debería considerar como raíz de documentos para el sitio Web del usuario.

Sintaxis: `UserDir directorio_nombredirectorio`

Predefinido: `UserDir public_html`

Contexto: configuración del servidor, host virtual

Por ejemplo, si mantiene el valor por defecto, cada vez que Apache reconozca una ruta `~username` después del nombre del servidor en la URL solicitada, traducirá el `~username` a `user_home_directory/public_html`. Si los directorios locales del usuario están almacenados en `/home`, la ruta traducida es `/home/username/publich_html`.

ADVERTENCIA: Debería añadir un `Userdir disabled root` para invalidar la capacidad de asignar esta directiva para señalar el directorio raíz.

El nombre del directorio que asigna con esta directiva debe ser accesible para el servidor Web. En otras palabras, si `/home/username` es el directorio local

deje `Userdir` asignado con el valor `public_html`, entonces `/home/username/public_html` debe ser accesible para el servidor Web. De hecho, Apache necesitará también leer y ejecutar acceso a ambos directorios, `/home` y `/home/username`.

Algunos administradores preocupados por la seguridad no son partidarios de esta idea de crear un directorio Web en un directorio local del usuario, puede asignar el `UserDir` a una ruta distinta como:

```
UserDir /www/users
```

Ahora, cuando se solicite `http://your_server_name/~username`, Apache traducirá esta solicitud a `/www/users/username`. De este modo puede mantener los archivos Web del usuario fuera del directorio local (`/home/username`) creando un nuevo directorio de nivel superior en el que tiene que crear un directorio para cada usuario.

Recuerde asegurarse de que Apache tiene acceso de lectura y escritura a cada uno de estos directorios.

mod_alias

El módulo `mod_alias` está compilado por defecto en Apache. Proporciona varias directivas que se pueden utilizar para asociar una parte del sistema de archivos del servidor a otra o, incluso, realizar servicios de redirección de URL.

Alias

La directiva Alias le permite asociar una ruta a cualquier sitio de su sistema de archivos del sistema.

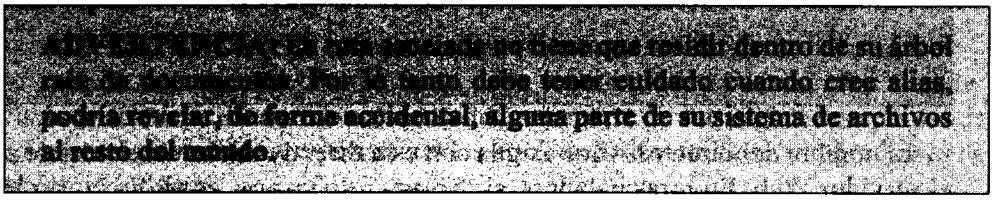
Sintaxis: `Alias URL-ruta ruta`

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva asocia `/data/` con `/web/data`; por lo tanto, cuando http://www.yoursite.com/data/:

```
Alias /data/ "/web/data/"
```

recibe una solicitud del tipo `http://data/datafile.cvs`, se devuelve el archivo llamado `/web/data/datafile.cvs`.



NOTA: Si utiliza un una barra final en la [illegible] tudes capaces de acceder al directorio de [illegible] un una barra final /.

AliasMatch

La directiva `AliasMatch` es parecida a la directiva `Alias`, excepto en que puede utilizar expresiones regulares.

Sintaxis: `AliasMatch regex ruta`

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva hace corresponder `www.yoursite.com/data/index.html` con el archivo `/web/data/index.html`:

```
AliasMatch ^/data(.*)   /web/data$1
```

Redirect

La directiva `Redirect` redirige una solicitud URL a otra. Si ha movido una sección de su sitio Web a un nuevo directorio o, incluso, a un nuevo sitio Web, puede utilizar esta directiva para asegurarse de que la gente que tiene el antiguo sitio Web en la carpeta de favoritos seguirá siendo capaz de encontrarlo.

Sintaxis: `Redirect [status_code] URL-antigua URL-nueva`

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

Por ejemplo, la siguiente directiva redirige todas las URL solicitadas que contienen la ruta `/data` a una nueva URL. Por lo tanto, las solicitudes a `www.yoursite.com/data/somefile.txt` se redireccionarán a `www.your-new-site.com/data/somefile.txt`:

```
Redirect /data www.your-new-site.com/data
```

La directiva `Redirect` tiene preferencia sobre las directivas `Alias` y `ScriptAlias`. Por defecto, el código de estado enviado al cliente es Temp (código de estado HTTP 302). Si quiere especificar un código de estado distinto, utilice lo siguiente:

Código de estado	Lo que hace
`Permanent`	Le dice al cliente que el redireccionamiento es permanente. Se devuelve el código de estado HTTP 301.

Código de estado	Lo que hace
`Temp`	Devuelve un estado temporal de redireccionamiento (302). Este es el valor por defecto.
`See other`	Devuelve un estado See Other (303), indicando que ese recurso se ha reemplazado.
`Gone`	Devuelve un estado Gone (410) indicando que el recurso se ha eliminado de forma permanente. Cuando se utiliza este estado, se suele omitir el argumento de la URL.

NOTA: Puede proporcionar códigos de estado HTTP válidos en formato numérico. Si el estado que proporciona se encuentra entre 300 y 399, debe estar presente la nueva URL; en caso contrario, debe omitirse. Podría preguntarse sobre la utilización de los distintos códigos de estado. En el futuro, los sistemas cliente pueden ser lo suficientemente inteligentes como para reconocer los códigos de estado de una forma más significativa. Por ejemplo, si un servidor proxy recibe un código de estado con una redirección permanente, puede almacenar esta información en el caché de modo que sea capaz de acceder directamente al nuevo recurso en una solicitud posterior.

RedirectMatch

La directiva `RedirectMatch` es parecida a la directiva `Redirect`, pero acepta expresiones regulares en lugar de una simple URL.

Sintaxis: `RedirectMatch [código-estado] regex URL`

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva redirige todas las solicitudes que terminan en `.htm` a una versión `.html` de la misma solicitud:

```
RedirectMatch (.*)\.htm$ www.yourserver.com$1.html
```

Como ejemplo del modo en el que esto podría funcionar, tenemos que la siguiente solicitud:

```
http://www.yoursite.com/some/old/dos/files/index.htm
```

es redirigida a:

```
http://www.yoursite.com/some/old/dos/files/index.html
```

Ver la directiva `Redirect` (última sección) para obtener información sobre `status_code`.

RedirectTemp

La directiva `RedirectTemp` es similar a la directiva `Redirect`. Permite al cliente saber que el rediccionamiento es temporal. Tenga en cuenta que la directiva `Redirect` también produce un estado temporal por defecto.

Sintaxis: `RedirectTemp URL-antigua URL-nueva`

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

RedirectPermanent

La directiva `RedirectPermanent` es parecida a la directiva `Redirect`. Permite que el cliente sepa que la redirección es permanente. Tenga en cuenta que la directiva `Redirect` produce un estado temporal por defecto, pero puede utilizar el código de estado 301 o la palabra clave `permanent`.

Sintaxis: `RedirectPermanent URL-antigua URL-nueva`

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

ScriptAlias

La directiva `ScriptAlias` crea un alias para la ruta física del directorio. Además, cualquier nombre de archivo suministrado en la solicitud es tratado como un script CGI, y el servidor intenta ejecutar el script.

Sintaxis: `ScriptAlias alias "ruta-fisica-directorio"`

Contexto: configuración del servidor, host virtual

Por ejemplo, la siguiente directiva puede utilizarse para procesar una solicitud del tipo `www.nitec.com/cgi-bin/somescript.pl`. El servidor intenta ejecutar `somescript.pl` si se verifica el permiso adecuado. Tenga en cuenta que el directorio `ScriptAlias` no es navegable:

```
ScriptAlias /cgi-bin/ "/www/nitec/public/cgi-bin/"
```

ScriptAliasMatch

La directiva `ScriptAliasMatch` es equivalente a la directiva `ScriptAlias` excepto en que utiliza una expresión regular, que le permite definir una regla dinámica para alias, en lugar de un alias fijo.

Sintaxis: `ScriptAliasMatch regex directorio`

Contexto: Configuración del servidor, host virtual

Por ejemplo, las dos directivas siguientes hacen exactamente lo mismo:

```
ScriptAliasMatch ^/cgi-bin(.*)    "/www/nitec/public/cgi-bin$1"
ScriptAlias   /cgi-bin/       "/www/nitec/public/cgi-bin/"
```

mod_speling

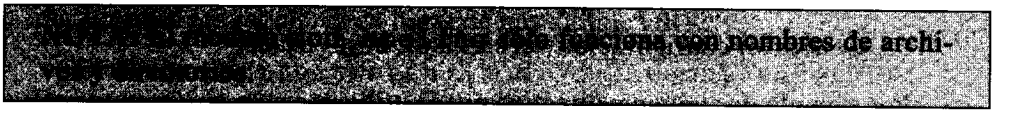
El módulo `mod_speling` no está compilado en Apache por defecto. Le permite manejar las solicitudes URL mal escritas o con problemas de mayúsculas y minúsculas. Compara el nombre del documento solicitado con todos los nombres de documentos en el directorio solicitado que tengan una o más coincidencias.

En el caso de la solicitud de un documento mal escrito, el módulo permite un solo error, como por ejemplo, la inserción de un carácter extra, la omisión de un carácter o una transposición. En el caso de errores en las mayúsculas y minúsculas, realiza una comparación de archivos independiente de mayúsculas y minúsculas. En cualquiera de los dos casos, si el módulo localiza un solo documento que se parece realmente al solicitado, lo envía al cliente. Si hay más de una coincidencia, le manda al cliente una lista. La única directiva que ofrece este módulo es `CheckSpelling`. La directiva `CheckSpelling` activa o desactiva el módulo `mod_speling`. Tenga en cuenta que cuando esta activada la corrección ortográfica, el servidor podría experimentar una pérdida de rendimiento debida a la realización de búsquedas extra que son necesarias para servir una solicitud de un documento mal escrito.

Sintaxis: `CheckSpelling On | Off`

Predefinido: `CheckSpelling Off`

Contexto: configuración del servidor, host virtual



mod_vhost_alias

El módulo `mod_vhost_alias` le permite crear dinámicamente host virtuales configurados. Este módulo sólo es adecuado para instalaciones Apache que necesiten muchos host virtuales. Por ejemplo, un Internet Service Provider (ISP) utilizando Apache, puede utilizar este módulo para reducir el trabajo de configuración que, de otro modo, sería necesario para cada nuevo cliente del host virtual.

Este módulo le permite crear configuraciones dinámicas de host virtuales utilizando la dirección IP o el nombre del host de un sitio Web virtual en la creación de las rutas físicas de los directorios necesarias para servir el sitio.

VirtualDocumentRoot

La directiva `VirtualDocumentRoot` le permite asignar la raíz de documentos para el host virtual utilizando una ruta de directorios interpolados.

Sintaxis: `VirtualDocumentRoot directorio-interpolado`

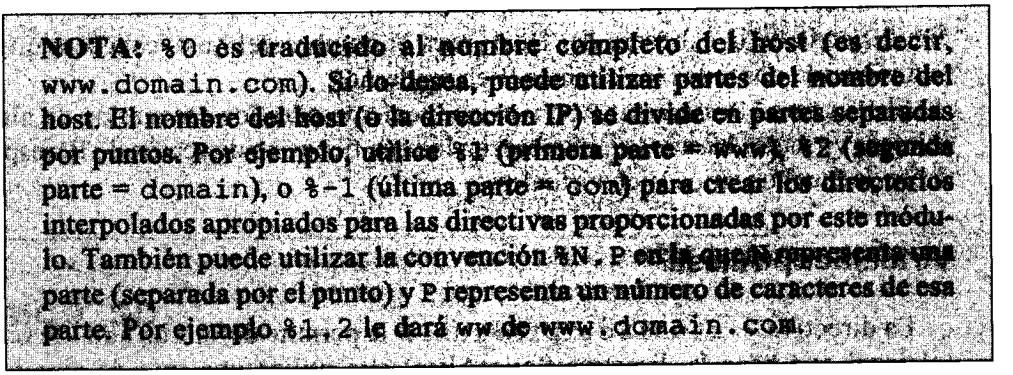
Contexto: configuración del servidor, host virtual

En la siguiente directiva, por ejemplo, cuando Apache recibe una solicitud para `http://www.domain.com/somepage.html`, la traduce a `/www/www.domain.com/htdocs/somepage.html`:

```
UseCanonicalName        Off
VirtualDocumentRoot     /www/%0/htdocs
```

`UseCanonicalName` está fijada en `off` por lo que Apache depende de la cabecera del Host para el nombre del host, que es suministrado por todos los clientes Web modernos.

La directiva `VirtualDocumentRoot` está indicada para los escenarios de alojamientos virtuales basados en el nombre, en los que tiene una dirección IP responsable de varios sitios Web.


NOTA: `%0` es traducido al nombre completo del host (es decir, `www.domain.com`). Si lo desea, puede utilizar partes del nombre del host. El nombre del host (o la dirección IP) se divide en partes separadas por puntos. Por ejemplo, utilice `%1` (primera parte = `www`), `%2` (segunda parte = `domain`), o `%-1` (última parte = `com`) para crear los directorios interpolados apropiados para las directivas proporcionadas por este módulo. También puede utilizar la convención `%N.P` en la que `N` representa una parte (separada por el punto) y `P` representa un número de caracteres de esa parte. Por ejemplo `%1.2` le dará `ww` de `www.domain.com`.

VirtualDocumentRootIP

La directiva `VirtualDocumentRootIP` le permite asignar la raíz de documentos para el host virtual utilizando una ruta de directorios interpolados, que se construye utilizando la dirección IP del sitio Web. Este método es conveniente si utiliza un alojamiento virtual basado en IP, porque tiene una sola dirección IP para cada sitio Web virtual.

Sintaxis: `VirtualDocumentRootIP directorio-interpolado`

Contexto: configuración del servidor, host virtual

En la siguiente directiva, por ejemplo, cuando Apache recibe una solicitud para `http://www.domain.com/somepage.html`, traduce la solicitud a `/www/IP_address_of_www.domain.com/htdocs/somepage.html`:

```
VirtualDocumentRootIP    /www/%0/htdocs
```

VirtualScriptAlias

La directiva `VirtualScriptAlias` le permite definir un alias de script (al igual que la directiva `ScriptAlias`) que utiliza una ruta de directorio interpolado.

> **Sintaxis:** `VirtualScriptAlias alias interpolated_directory`
>
> **Contexto:** configuración del servidor, host virtual

En la siguiente directiva. Por ejemplo, cuando Apache recibe una solicitud para `http://www.domain.com/cgi-bin/script_name`, traduce la solicitud a `/www/www.domain.com/cgi-bin/script_name`:

```
UseCanonicalName          Off
VirtualScriptAlias   /cgi-bin/      /www/%0/cgi-bin
```

La directiva `UseCanonicalName` está fijada en `off`, y necesita que Apache dependa de la cabecera del Host para el nombre del host, cabecera que es suministrada por todos los clientes Web modernos.

El `VirtualDocumentRoot` es conveniente para los escenarios de alojamientos virtuales basados en nombre, en los que tiene una dirección de IP responsable de muchos sitios Web virtuales.

VirtualScriptAliasIP

La directiva `VirtualScriptAliasIP` le permite definir un alias de script (al igual que hace la directiva ScriptAlias) que utiliza una ruta de directorio interpolada.

> **Sintaxis:** `VirtualScriptAliasIP alias directorio-interpolado`
>
> **Contexto:** Configuración del servidor, host virtual

En la siguiente directiva, por ejemplo, cuando se recibe en Apache una solicitud para `http://www.domain.com/cgi-bin/script_name`, Apache traduce la solicitud a `/www/IP_address/cgi-bin/script_nameVirtualScriptAliasIP /cgi-bin/ /www/%0/cgi-bin`.

Otros módulos

Los módulos de esta sección, mostrados en la tabla 5.11, no se encuentran dentro de ninguna categoría en particular.

Tabla 5.11. Otros módulos

Módulo	Función
mod_so	Soporte para cargar módulos en tiempo de ejecución.
mod_imap	El manejador de archivos de integración de imágenes.
mod_proxy	Convierte a Apache en un servidor proxy con capacidad de caching. Ver el capítulo 10 para obtener los detalles.
mod_isapi	Soporte de la extensión ISAPI de Windows. Ver el capítulo 21 para obtener los detalles.
mod_file_cache	Archivos cacheados en la memoria para dar un servicio más rápido.
mod_dav	Proporciona funcionalidad Web-based Distributed Authoring and Versioning (WebDAV) de clase 1 y de clase 2.
mod_example	Un ejemplo de desarrollo de un módulo para aprender a escribir un módulo de Apache. Sólo es útil para los programadores de C.

mod_so

El módulo mod_so permite a Apache cargar el código ejecutable que necesitan otros módulos o cargar otros módulos durante la puesta en marcha del servidor. Puede compilar todos los módulos como módulos DSO (Dynamic Shared Object) excepto éste.

LoadFile

La directiva LoadFile carga el archivo nombrado durante el arranque. Normalmente, se carga un archivo DLL (dynamically linked library) que necesita otro módulo utilizando esta directiva (sólo en Windows).

Sintaxis: LoadFile nombrearchivo [nombrearchivo...]

Contexto: configuración del servidor

LoadModule

La directiva LoadModule carga un módulo que se ha compilado como un DSO.

Sintaxis: `LoadModule module_filename`

Contexto: configuración del servidor

mod_imap

El módulo `mod_imap` se compila en Apache por defecto. Proporciona soporte de integración de imágenes, que ha sido proporcionada por el programa CGI `imagemap`. Puede utilizar la directiva `AddHandler` para especificar el manejador de `imap-file` (construido en este módulo) para cualquier extensión de archivo.

Por ejemplo, la siguiente directiva hace que Apache trate a todos los archivos que tengan la extensión `.map` como integración de imágenes, y los Apache procesa los archivos utilizando el módulo `mod_imap`:

```
AddHandler imap-file map
```

NOTA: El módulo mod-imap sigue soportando el antiguo formato:

`AddType application/x-httpd-imap map`

Sin embargo, no se recomienda el formato antiguo porque su soporte posiblemente se rechazará en el futuro.

Los contenidos de un archivo de integración de imágenes pueden tener cualquiera de las siguientes sintaxis:

```
directive value [x,y ...]
directive value "Menu text" [x,y ...]
directive value x,y ... "Menu text"
```

Las directivas permitidas en un archivo de integración de imágenes son:

- `base`: las URL relativas que se utilizan en los archivos de integración se consideran relativas al valor de esta directiva. Tenga en cuenta que la asignación de la directiva `Imapbase` se invalida por esta directiva cuando se encuentra en un archivo de integración. Su valor por defecto es `http://server_name/. base_uri`, que es sinónimo de base.
- `default`: especifica la acción a tomar cuando las coordenadas no se corresponden con las de `poly`, `circle` o `rect`, y no se dan directivas `point`. El valor por defecto para esta directiva es `nocontent`, que le dice al cliente que mantenga la misma página desplegada.
- `poly`: define un polígono utilizando al menos 3 puntos hasta un máximo de 100 puntos. Si el usuario que proporciona las coordenadas cae dentro del polígono, se activa esta directiva.

- `circle`: define un círculo utilizando el centro de coordenadas y un punto del círculo. Si el usuario que proporciona las coordenadas cae dentro del círculo, se activa esta directiva.
- `rect`: define un rectángulo utilizando las coordenadas de dos esquinas opuestas. Si el usuario que proporciona las coordenadas cae dentro del rectángulo, se activa esta directiva.
- `point`: define un sola coordenada puntual. La directiva `point` más cercana a la coordenada suministrada por el usuario, se utiliza cuando no se satisface ninguna otra directiva.

El valor es una URL absoluta o relativa, o uno de los valores especiales en la lista siguiente. Las coordenadas (*x,y*) están separadas por caracteres en blanco. El texto que va entre comillas dobles (mostrado en el segundo tipo de sintaxis) se utiliza como texto del enlace, si se genera un menú de integración de imágenes. Cualquier línea con un carácter almohadilla # se considera como un comentario y Apache lo ignora. Las coordenadas están escritas en el formato x,y, en el que cada coordenada está separada por un carácter en blanco. La cadena de texto entre comillas se utiliza como enlace cuando se genera un menú. En ausencia de dicha cadena, la URL es el enlace, tal y como se muestra en el siguiente archivo de integración de imágenes:

```
# Los comentarios van aqui
#  Versión 1.0.0
base http://www.yoursite.com/some/dir
rect thisfile.html "Customer info" 0,0 100,200
circle http://download.yoursite.com/index.html 295,0 100,22
```

Si este archivo de integración de imágenes se llama `imagemap.map`, puede venir referido como se muestra a continuación desde otro archivo HTML:

```
<A HREF="/path/to/imagemap.map"><IMG ISMAP SRC="/path/to/
imagemap.gif"></A>
```

ImapMenu

La directiva `ImapMenu` determina la acción para una solicitud de un archivo de integración de imágenes sin coordenadas válidas.

> **Sintaxis:** `ImapMenu {None, Formatted, Semi-formatted, Unformatted}`
>
> **Contexto:** configuración del servidor, host virtual, directorio, archivo de control de acceso en el ámbito de directorio (`.htaccess`)
>
> **Invalidar:** `Indexes`

`ImapMenu` permite las siguientes acciones:

Acción	Lo que hace
None	No se genera ningún menú, y se lleva a cabo la acción por defecto.
Formatted	Se genera el menú más sencillo. Se ignoran los comentarios. Se imprime una cabecera de nivel 1, y después una regla horizontal y los enlaces, en líneas separadas.
Semi-formatted	En el menú semi-formateado, se imprimen los comentarios, las líneas en blanco se convierten en saltos HTML y no se imprime ni una cabecera ni una regla horizontal.
Unformatted	En el menú sin formato, se imprimen los comentarios y se ignoran las líneas en blanco.

ImapDefault

La directiva `ImapDefault` define la acción por defecto para la integración de imágenes. Este valor por defecto se puede invalidar en el archivo de integración de imágenes utilizando la directiva por defecto.

Sintaxis: `ImapDefault {Error, Nocontent, Map, Referer, URL}`

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

Invalidar: `Indexes`

La siguiente tabla muestra el significado de cada uno de los posibles valores para la directiva `ImapDefault`.

Valor	Lo que significa
URL	Una URL relativa o absoluta. Las URL relativas resuelven en relación con la base.
Map	Lo mismo que URL del archivo de integración de imágenes en sí. A no ser que `ImapMenu` tenga un valor asignado de `none`, se creará un menú.
Menu	Lo mismo que Map.
Referer	Lo mismo que URL del documento al que se refiere. El valor por defecto es http://servername/ si no está presente `Referer`: header.

Valor	Lo que significa
`Nocontent`	Se envía un código de estado de 204 para decirle al cliente que mantenga la misma página desplegada. No es válido para base.
`Error`	Se envía un código de estado de 500 para informar al cliente sobre un error del servidor.

ImapBase

La directiva `ImapBase` asigna la base por defecto que se utiliza en los archivos de integración de imágenes. Esta base asignada se puede invalidar utilizando la directiva `base` dentro del archivo de integración de imágenes. Si esta directiva no está presente, la base por defecto es `http://servername/`.

Sintaxis: `ImapBase {Map, Referer, URL}`

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

Invalidar: `Indexes`

mod_file_cache

El módulo `mod_file_cache` permite a Apache guardar archivos estáticos frecuentemente utilizados en el caché que no se cambian a menudo. Utilizando este módulo, puede cargar un archivo en la memoria o preabrir un archivo y reducir la cantidad de entradas y salidas en el disco para cada solicitud. Esto significa que Apache no tiene que leer archivos desde el disco para cada solicitud. Este módulo no funciona en todas las plataformas y debería utilizarse únicamente si está seguro de que ciertos archivos solicitados con frecuencia no se van a cambiar. Si un archivo cacheado o un archivo preabierto cambia, el servidor Apache no puede admitir los cambios hasta que no sea reiniciado.

MMapFile

La directiva `MMapFile` le permite cargar un archivo en la memoria mediante llamadas sistemáticas `mmap()`. Una vez que se ha cargado un archivo en la memoria, el servidor no detectará ningún cambio en el archivo físico. Por lo tanto, es necesario tener cuidado cuando se utiliza esta directiva. Si cambia un archivo que se ha cargado previamente en la memoria, asegúrese de reiniciar el servidor Apache.

Sintaxis: `MmapFile nombrearchivo [nombrearchivo] [...]`

Contexto: configuración del servidor

CacheFile

La directiva `CacheFile` preabre un archivo y lo guarda en el caché, de modo que cuando llega la solicitud, Apache no tiene que realizar llamadas sistemáticas para abrir y leer el archivo. Una vez que se ha guardado un archivo en el caché, cualquier cambio que se quiera realizar en el archivo requerirá reiniciar el servidor Apache.

Sintaxis: `CacheFile nombrearchivo[nombrearchivo] [...]`

Contexto: configuración del servidor

mod_dav

El módulo `mod_dav` le permite utilizar las extensiones WebDAV del protocolo HTTP 1.1. Para aprender más sobre estas extensiones visite el sitio `www.webdav.org`.

Dav

La directiva `Dav` valida o invalida el módulo `mod_dav`. Debe asignarse el valor `On` si quiere utilizar la característica WebDAV dentro de un contenedor de directivas.

Sintaxis: `Dav On | Off`

Contexto: directorio

DavLockDB

La directiva `DavLockDB` asigna el fully qualified pathname (nombre completo de la ruta de la máquina, incluido el dominio) del archivo de bloqueo de bases de datos.

Sintaxis: `DavLockDB nombrearchivo`

Contexto: configuración del servidor, host virtual

DavMinTimeout

La directiva `DavMinTimeout` fija el intervalo de bloqueo de recursos mínimo en segundos. El valor por defecto asegura que un cliente WebDAV no es bloqueado (independientemente del tiempo) automáticamente por el servidor.

Sintaxis: `DavMinTimeout seconds`

Predefinido: `DavMinTimeout 0`

Contexto: directorio

DavDepthInfinity

La directiva `DavDepthInfinity` permite que una solicitud `PROPFIND` con la cabecera `Depth` se fije en infinito. Se recomienda el valor por defecto.

Sintaxis: `DavDepthInfinity On | Off`

Predefinido: `DavDepthInfinity Off`

Contexto: directorio

Parte II
Administrar sitios Web

6 Alojar sitios Web virtuales

En este capítulo

1. Conocemos los dos tipos de host virtuales.
2. Creamos host virtuales basados en IP.
3. Creamos host virtuales basados en nombre.
4. Configuramos DNS para host virtuales.
5. Ejecutamos un sitio Web con sus propios privilegios de usuario y de grupo.
6. Manejamos una gran cantidad de host virtuales en un solo servidor Apache utilizando `mod_perl`.
7. Simplificamos el proceso de creación de nuevos host virtuales utilizando el script `makesite`.
8. Manejamos host virtuales utilizando el módulo `mod_v2h` y la base de datos MySQL.

Apache puede atender varios sitios Web desde un único servidor. Por ejemplo, una compañía Web alojada puede tener un solo servidor Web ejecutando

Apache, que está sirviendo a cientos de sitios Web clientes. Este tipo de sistema tiene un nombre de host principal y muchos alias IP o nombres de host virtuales. Un sitio Web atendido mediante este tipo de host virtuales se denomina sitio Web virtual. El soporte de Apache para sitios virtuales (llamados hosts virtuales) es impresionante. Este capítulo discute el modo de crear varios tipos de host virtuales y cómo manejarlos utilizando distintas técnicas.

Entender las capacidades del hospedaje virtual en Apache

Cuando establece Apache en un host de Internet, puede responder a una solicitud HTTP a ese host. Por ejemplo, si establece Apache en un host llamado `server1.doman.com`, Apache servirá solicitudes HTTP a ese host. Sin embargo, si establece sus registros DNS con dos nombres de host (como por ejemplo, `www.mycompany-domain.com` y `www.friendscompany-domain.com`) en la misma máquina, Apache puede servir a estos dos dominios como sitios Web virtuales. En ese caso, `www.mycompany-domain.com` se considera el nombre del host Web principal (servidor principal), y el resto se considerarán como sitios Web virtuales o host virtuales.

Apache permite que los host virtuales hereden configuración del servidor principal, lo que da lugar a una configuración del host virtual mucho más manejable en grandes instalaciones, en las que se puede compartir parte del contenido. Por ejemplo, si decide tener únicamente un depósito central de CGI y permitir a los host virtuales utilizar los script que están almacenados en ese depósito, no necesita crear una directiva `ScriptAlias` en cada contenedor de host virtual. Simplemente utilice una directiva en la configuración del servidor principal y tendrá todo el trabajo hecho. Cada host virtual puede utilizar el alias como si le perteneciese.

El archivo de configuración de Apache, el `httpd.conf`, separa la configuración del host virtual de la configuración del servidor principal utilizando el contenedor `<VirtualHost>`. Por ejemplo, observe el archivo `httpd.conf` del listado 6.1.

Listado 6.1. httpd.conf

```
# archivo httpd.conf

ServerName main.server.com
Port 80
ServerAdmin mainguy@server.com
DocumentRoot "/www/main/htdocs"

ScriptAlias /cgi-bin/ "/www/main/cgi-bin/"
```

```
Alias /images/  "/www/main/htdocs/images/"

<VirtualHost  192.168.1.100>
     ServerName vhost1.server.com
     ServerAdmin vhost1_guy@vhost1.server.com
     DocumentRoot "/www/vhost1/htdocs"
     ScriptAlias /cgi-bin/ "/www/vhost1/cgi-bin/"
</VirtualHost>

<VirtualHost  192.168.1.110>
     ServerName vhost2.server.com
     ServerAdmin vhost2_guy@vhost2.server.com
     DocumentRoot "/www/vhost2/htdocs"
     ScriptAlias /cgi-bin/ "/www/vhost2/cgi-bin/"
     Alias /images/ "/www/vhost2/htdocs/images/"
</VirtualHost>
```

El listado 6.1 muestra dos sitios Web virtuales llamados `vhost1.server.com` y `vhost2.server.com`, que están definidos en sus propios contenedores `<VirtualHost>`. Todas las directivas incluidas en cada uno de los contenedores `<VirtualHost>` se aplican, únicamente, al host virtual al que sirven. Por tanto, cuando un navegador Web solicita `http://vhost1.server.com/index.html`, el servidor Web Apache busca la página `index.html` en el directorio `/www/vhost1/htdocs`. De igual modo, cuando un navegador Web solicita `http://vhost2.server.com/cgi-bin/hello.pl`, el script se ejecuta desde el directorio `/www/vhost2/cgi-bin`.

Sin embargo, muchas directivas de la configuración del servidor principal (es decir, cualquier directiva fuera del contenedor `<VirtualHost>`) se siguen aplicando al host virtual que no las invalida. Por ejemplo, el servidor `vhost1.server.com` del listado 6.1 no tiene una directiva `Alias` para el alias del directorio `/images/`. Por lo tanto, cuando el navegador Web solicita el archivo `http://vhost1.server.com/images/pretty_pic.gif`, la imagen se sirve desde el directorio `/www/main/htdocs/images`. Como `vhost2.server.com` invalida el alias `/images/` por su cuenta, se sirve una solicitud similar desde el directorio `/www/vhost2/htdocs/images`.

Establecer un host virtual

Hay tres modos de crear sitios Web virtuales cuando utilizamos Apache:

- **Basado en el nombre:** los sitios Web virtuales basados en el nombre son muy comunes. Este tipo de configuración exige que tenga varios nombres de host en un solo sistema. Puede crear varios CNAME o registros A en DNS para que se encuentren en un solo host. Como este método no utiliza direcciones IP en la configuración de Apache, es fácil de portar si cambia sus direcciones IP a su servidor Web.

- **Basado en IP:** este método necesita direcciones IP en la configuración de Apache y por eso resulta sencillo de portar cuando hay que cambiar las direcciones IP.
- **Varios servidores principales:** este método implica la utilización de varias configuraciones de servidores Web principales. Este método sólo se recomienda en el caso de que deba mantener archivos de configuración separados para host virtuales. Este es el método menos recomendado y es difícil de usar.

Host virtuales basados en nombre

Este es el método más recomendado. Necesita una sola dirección IP para alojar cientos de sitios Web virtuales.

NOTA: Lo único que ha de recordar cuando utilice este método, es que no funciona con los navegadores Web que no soportan el protocolo HTTP 1.1. Únicamente los primeros navegadores, como Microsoft IE 1.x o Netscape Navigator 1.x, no soportan HTTP 1.1. Por lo que, realmente, no se trata de un problema muy grave. La mayoría de la gente utiliza 3.x o versiones posteriores de navegadores Web, que son compatibles con la técnica de hospedaje virtual basada en nombre.

Por ejemplo, imagine que tiene una dirección IP `192.168.1.100` y quiere alojar `vhost1.domain.com` y `vhost2.domain.com` en el mismo servidor. Puede hacer lo siguiente:

1. Primero, tiene que crear el registro DNS apropiado en su servidor DNS para enlazar `vhost1.domain.com` y `vhost2.domain.com` a la dirección IP `192.168.1.100`. Diríjase a la sección "Configurar DNS para un host virtual" para obtener los detalles.
2. Tiene que crear un segmento de configuración, parecido al que le presentamos a continuación, en el archivo `httpd.conf`.

```
NameVirtualHost  192.168.1.100

<VirtualHost  192.168.1.100>
     ServerName vhost1.domain.com
     ServerAdmin someone@vhost1.domain.com
     DocumentRoot "/www/vhost1/htdocs"

     #
     # Aquí se coloca cualquier directiva extra que necesite
     #
```

```
</VirtualHost>

<VirtualHost 192.168.1.100>
    ServerName vhost2.domain.com
    ServerAdmin someone@vhost2.domain.com
    DocumentRoot "/www/vhost2/htdocs"

    #
    # Aquí se coloca cualquier directiva extra que necesite
    #

</VirtualHost>
```

No olvide crear el directorio raíz de documentos si aún no lo ha hecho. Además, puede añadir más directivas en cada una de las configuraciones de los host virtuales en caso necesario.

3. Reinicie Apache utilizando el comando de reinicio `/usr/local/apache/apachectl` y acceda a cada uno de los host virtuales utilizando `http://vhost1.domain.com` y `http://vhost2.domain.com`.

Como acabamos de ver en el ejemplo, ambos contenedores de host utilizan la misma dirección IP (`192.168.1.100`). Por lo tanto, debemos preguntarnos cómo sabe Apache cuál es el sitio Web que se está solicitando cuando llega una solicitud por la dirección IP `192.168.1.100`.

HTTP 1.1 necesita que una cabecera llamada `Host` esté presente en cada solicitud que el navegador le presenta al sitio Web. Por ejemplo, a continuación tenemos una cabecera de una solicitud HTTP de un navegador Web a un servidor ejecutándose en `rhat.domain.com`.

```
GET / HTTP/1.1
Host: rhat.domain.com
Accept: text/html, text/plain
Accept: postscript-file, default, text/sgml, */*;q=0.01
Accept-Encoding: gzip, compress
Accept-Language: en
User-Agent: Lynx/3.0.0dev.9 libwww-FM/2.14
```

Cuando Apache ve la cabecera `Host: rhat.domain.com`, puede servir de forma inmediata, la solicitud utilizando el host virtual apropiado, es decir aquel que tiene un `ServerName` coincidente.

Host virtuales basados en IP

Este método exige el uso de direcciones IP en la creación de los host. La dirección IP ha de estar codificada por harware en el archivo de configuración en cada etiqueta del contenedor `<VirtualHost>`. Esto puede crear grandes quebraderos de cabeza si cambia las direcciones IP con cierta frecuencia. Este

método no tiene ninguna ventaja respecto al método anterior. El siguiente ejemplo muestra tres host virtuales basados en IP.

```
<VirtualHost 192.168.1.1>
    ServerName vhost1.server.com
    # Aquí se colocan otras directivas
</VirtualHost>

<VirtualHost 192.168.1.2>
    ServerName vhost2.server.com
    # Aquí se colocan otras directivas
</VirtualHost>

<VirtualHost 192.168.1.3>
    ServerName vhost3.server.com
    # Aquí se colocan otras directivas
</VirtualHost>
```

Cada una de estas direcciones IP, deben ir unidas a la interfaz Ethernet apropiada en el servidor. Por ejemplo, la configuración anterior requiere un sistema que aloje los sitios Web para tener los siguientes registros DNS en su archivo de configuración del servidor DNS.

```
; Address Records
vhost1.server.com.    IN    A 192.168.1.1
vhost2.server.com.    IN    A 192.168.1.2
vhost3.server.com.    IN    A 192.168.1.3

; Reverse DNS records
1                     IN PTR  vhost1.server.com.
2                     IN PTR  vhost2.server.com.
3                     IN PTR  vhost3.server.com.
```

Cada una de estas direcciones deben estar unidas a una o más interfaces Ethernet en el servidor. En un sistema Linux, se pueden unir varias direcciones IP utilizando la técnica de denominación IP (aliasing IP). Por ejemplo:

```
/sbin/ifconfig eth0    192.168.1.1 up
/sbin/ifconfig eth0:0  192.168.1.2 up
/sbin/ifconfig eth0:1  192.168.1.3 up
```

Las tres direcciones IP están unidas a la interfaz Ethernet `eth0` y a sus dos alias, `eth0:0` y `eth0:0`. Como consecuencia, el sistema responderá a cada dirección IP.

Varios servidores principales como host virtuales

Sólo se recomienda utilizar varios servidores principales como host virtuales cuando está obligado (normalmente, por razones que no son de índole técnico) a

mantener distintos archivos de configuración `httpd.conf`. Por ejemplo, imagine que tiene 16 direcciones IP y que quiere suministrar a 16 clientes (o departamentos) su propio archivo de configuración `httpd.conf` para, que de ese modo, cada entidad pueda gestionar absolutamente todo por su cuenta e, incluso, crear un host virtual utilizando los contenedores `<VirtualHost>` dentro de su propio `httpd.conf`.

NOTA: Antes de adentrarse en la aventura de utilizar este método, considere con cuidado, la posibilidad de evitar crear varias instancias de servidores principales utilizando los contenedores `<virtualhost>`.

El listado 6.2 muestra una versión simplificada de un `httpd.conf` (llamado `httpd-100.conf`) que utiliza la directiva `Listen` para decirle a Apache que únicamente sirva la dirección IP `192.168.1.100`, asociada con el sistema que está ejecutándose. Esto da lugar a la implementación de un solo host virtual.

Listado 6.2. httpd-100.conf

```
ServerType standalone
ServerRoot "/usr/local/apache"
PidFile /usr/local/apache/logs/httpd-192.168.1.100.pid
ScoreBoardFile /usr/local/apache/logs/httpd-
192.168.1.100.scoreboard
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 60
MaxSpareServers 100
StartServers 50
MaxClients 200
MaxRequestsPerChild 0
Port 80
Listen 192.168.1.100:80
User prod
Group prod
ServerName prod.domain.com
ServerAdmin kabir@prod.domain.com
DocumentRoot "/www/prod/htdocs"
```

Tenga en cuenta que la directiva `Listen` también toma un número de puerto como parámetro. En el listado 6.2, se le dice a Apache que escuche la dirección IP dada en el puerto 80. La directiva `Port` sigue siendo necesaria porque su valor se utiliza en las URL de referencia propia generadas.

El listado 6.3 muestra otro archivo `httpd.conf` simplificado (llamado `httpd-101.conf`) que le dice a Apache que escuche únicamente la dirección `192.168.1.10`. Esto da lugar a la implementación de otro host virtual

utilizando el método de varios servidores principales en la creación de host virtuales.

Listado 6.3. httpd-101.conf

```
ServerType standalone
ServerRoot "/usr/local/apache"
PidFile /usr/local/apache/logs/httpd-192.168.1.101.pid
ScoreBoardFile /usr/local/apache/logs/httpd-
192.168.1.101.scoreboard
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 5
MaxSpareServers 10
StartServers 5
MaxClients 10
MaxRequestsPerChild 0
Port 80
Listen 192.168.1.101:80
User stage
Group stage
ServerAdmin lance@stage.domain.com
DocumentRoot "/www/stage/htdocs"
```

Un sistema que quiera ejecutar dos servidores principales Apache, utilizando los archivos de configuración de los listados 6.2 y 6.3, debe tener:

- Una o más interfaces Ethernet respondiendo a las direcciones IP nombradas. Por ejemplo, en un sistema Lynux, puede unir la interfaz Ethernet `eth0` tanto a `192.168.1.100` como a `192.168.1.101` utilizando el comando denominación (aliasing) IP, del siguiente modo:

  ```
  /sbin/ifconfig eth0 192.168.1.100 up
  /sbin/ifconfig eth0:0 192.168.1.101 up
  ```

  Por supuesto, si un sistema tiene varias interfaces Ethernet y quiere utilizar un servidor principal para cada interfaz, no necesita utilizar alias IP.

- Las direcciones IP deben tener nombres de host asociados con ellas. Para los archivos de configuración anteriores, la dirección `192.168.1.100` está asociada con `prod.domain.com` y la dirección `192.168.101` está asociada a `stage.domain.com`.

Un sistema que tenga esas direcciones IP unidas a su interfaz o interfaces, puede ejecutar dos demonios principales de Apache utilizando los siguientes comandos:

```
/usr/local/apache/bin/httpd -f   conf/httpd-100.conf
/usr/local/apache/bin/httpd -f   conf/httpd-101.conf
```

Si está ejecutando el comando `ps auxww | grep httpd | grep root | grep conf`, verá dos servidores principales Apache que se ejecutan como usuario raíz.

Configurar DNS para un host virtual

En la mayor parte de los casos, su ISP es responsable de proporcionar servicio DNS para dominios. En este caso, puede saltarse esta sección. Sin embargo, si no ejecuta BIND, el servidor DNS más utilizado, en Lynux o en otro sistema del tipo Lynux, le mostrará cómo configurar DNS para sus host virtuales.

A lo largo de este libro, los usuarios de Windows: pueden obtener los detalles de la configuración en su servidor DNS de Windows.

Entender los archivos de zona

Es necesario un *archivo de zona* para establecer DNS para un host virtual. Un archivo de zona es una descripción textual de sus registros DNS. El servidor DNS carga estos archivos para implementar su servicio DNS para una sola zona, que normalmente denominamos dominio Internet.

A continuación tenemos un ejemplo de un archivo de zona. En este ejemplo, supongo que el nuevo host virtual que quiere establecer se llama `www.newdomain.com`, que los nombres de host del servidor son `ns1.domain.com` (principal) y `ns2.domain.com` (secundario), y que el nombre del host en su servidor Web es `www.domain.com` (`192.168.1.100`). Asegúrese de que cambia los nombres de host y las direcciones IP para tener el mismo establecimiento.

El archivo de zona para www.newdomain.com se llama `/var/named/newdomain.zone` y contiene las líneas siguientes:

```
@ IN SOA newdomain.com.    hostmaster.newdomain.com.(

              20011201001  ; Serial YYYYMMDDXXX
              7200         ; refresh
              3600         ; (1 hour) retry
              1728000      ; (20 days) expire
              3600)        ; (1 hr) minimum ttl

; Nombres de los servidores
  IN NS      ns1.domain.com.
  IN NS      ns2.domain.com.

; Registros A para alojamiento virtual basado en IP
; www.newdomain.com.      IN   A    192.168.1.100

; CNAME para alojamiento virtual basado en nombre
  www.newdomain.com.      IN   CNAME  www.domain.com.
```

NOTA: La configuración DNS anterior, supone que no va a utilizar el alojamiento virtual basado en IP y, por lo tanto, no crea un registro A para `www.newdomain.com`. La línea de registro A se comenta aparte. Puede borrar la señal de comentario si utiliza el método de alojamiento virtual basado en IP. Como el alojamiento virtual basado den nombre no necesita una dirección IP única, se crea un registro CNAME para relacionar `www.newdomain.com` con `www.domain.com` (que es el servidor Web principal).

A continuación tenemos lo que está ocurriendo en la configuración anterior:

- La primera línea comienza con un registro DNS llamado Start of Address (`SOA`), que especifica el número de serie, la razón de actualización, la razón de reintento, el tiempo de expiración y el tiempo de vida (TTL) de los valores.
- Los servidores DNS utilizan el número de serie para determinar si necesitan o no sus registros cacheados. Para entenderlo, en el ejemplo tenemos el número de serie `20011201001`, que indica que la última actualización tuvo lugar el 12/01/2001 y que su primer (`001`) cambio tuvo lugar ese día. Si el administrador de DNS cambia la configuración DNS el día 12/02/2001, el número de serie debería cambiar para reflejar el cambio, de modo que cualquier servidor DNS remoto que tenga cacheados los registros, puede comparar números de serie entre la versión cacheada y la nueva versión, y decidir si descarga los datos DNS nuevos.
- La razón de actualización nos dice la frecuencia con que deberían actualizarse los registros en el servidor.
- La razón de reintento establece que en caso de un fallo en el proceso de pedida de solicitudes por parte de un servidor DNS remoto, este servidor DNS reintenta enviar la solicitud cada un cierto intervalo.
- El tiempo de expiración, le dice al servidor DNS remoto que elimine, a la fuerza, cualquier dato del caché que tenga en el dominio dado una vez que pasan una cantidad de días determinados.
- La entrada final en `SOA`, establece que los registros tengan un valor mínimo especificado de tiempo de vida. Tenga en cuenta que todo lo que se encuentre detrás del punto y coma es tratado como una línea de comentario y, por lo tanto, ignorado.
- Las siguientes dos líneas no comentadas establecen que los servidores DNS responsables de `newdomain.com` son `ns1.domain.com` y `ns2.domain.com`. Si no tiene un segundo servidor DNS, debería considerar el uso de un servicio DNS secundario de terceras partes como es el `http://www.secondary.com`.

Establecer las DNS para host virtuales nuevos

Puede establecer un DNS para cada nuevo host virtual en Linux del siguiente modo:

1. Cree un archivo de zona para el nuevo dominio. (Ver la sección anterior para obtener los detalles sobre los archivos de zona.)
2. Añada las líneas que se encuentran a continuación, en el archivo `/etc/named.conf`, para permitir la zona `newdomain`.

```
zone "newdomain.com" IN {
        type master;
        file "newdomain.zone";
        allow-update { none; };
};
```

   Esto le dice al demonio DNS que el archivo de zona de `newdomain.com` es `/var/named/newdomain.zone`, y que se trata de su servidor DNS principal (maestro), para esta zona.
3. Ejecute el comando `killall -HUP named` para forzar al servidor a recargar el `/etc/named.conf` y el nuevo archivo de zona.
4. Intente hacer un ping `www.newdomain.com`. Si no obtiene una respuesta, compruebe el archivo `/var/log/messages` para encontrar los errores que pueda tener el servidor (`named_`) registrados en los archivos de zona `/etc/named.conf` o `/var/named/newdomain`. En ese caso, corrija los errores y reinicie el servidor tal y como se ha discutido en el paso previo.
5. Una vez que puede realizar un ping a `www.newdomain.com`, está preparado para crear configuración Apache para este host virtual, del modo descrito en secciones anteriores de este capítulo.

Ofrecer servicios de correo virtual

Para proporcionar servicios de correo virtual en los host virtuales nuevos, tiene que modificar el archivo newdomain.zone y añadir uno o más registros MX apropiados. Por ejemplo, imagine que su servidor de correo es `mail.domain.com` y que está configurado para aceptar correos de `newdomain.com`. En este caso, puede modificar `/var/named/newdomain.zone` para que sea del siguiente modo:

```
@ IN SOA newdomain.com.    hostmaster.newdomain.com.(
              20011201002  ; Serial YYYYMMDDXXX
              7200         ; refresh
              3600         ; (1 hour) retry
              1728000      ; (20 days) expire
              3600)        ; (1 hr) minimum ttl
```

```
; Nombres de servidores
  IN NS      ns1.domain.com.
  IN NS      ns2.domain.com.
  IN MX  10 mail.domain.com.

; CNAME para el alojamiento virtual basado en nombre
  www.newdomain.com.      IN    CNAME   www.domain.com.
```

Tenga en cuenta que si tiene varios servidores de correo, puede añadirlos utilizando un esquema sencillo de prioridades. Por ejemplo, imagine que quiere que todos los correos de `newdomain.com` vayan a `mail.domain.com`, y que en caso de que este servidor esté incapacitado, quiere utilizar `bkupmail.domain.com`.

Puede añadir simplemente un segundo registro MX, tal y como se muestra a continuación:

```
IN MX   10 mail.domain.com.
IN MX   20 bkupmail.domain.com.
```

Al ser más pequeño el número asignado a `mail.domain.com`, le convierte en un servidor Web de mayor prioridad que el servidor `bkupmail.domain.com`. Asegúrese de que tiene configurado el software de su servidor de correo para que soporte dominios de correo virtuales.

Asignar usuario y grupo a cada host virtual

Si ha configurado Apache utilizando el módulo MPM Perchild, puede asignar usuario y grupo para cada host virtual. El principal beneficio de este método es su simpleza y no el sistema de múltiples servidores de correo basados en `httpd.conf`.

ADVERTENCIA: Esta aproximación es más lenta que la ejecución de varios servidores principales Apache con distintas direcciones IP y distintas directivas `User` y `Group`. La pérdida de velocidad se debe al incremento de complejidad en el procesamiento de solicitudes internas. El concepto de este método es el siguiente: instruye a Apache para que ejecute un conjunto de procesos hijo con un ID de usuario y un ID de grupos determinados. Cuando llega una solicitud al host virtual y la toma un proceso hijo, éste determina primero si puede manejarla. Si el hijo no es responsable del host virtual solicitado deberá pasar la solicitud al hijo apropiado mediante una llamada socket, que disminuirá la velocidad del proceso de solicitud. Como este es un concepto nuevo, en futuras versiones Apache se solucionará el tema de velocidades.

El material que se presenta a continuación supone que quiere establecer dos host virtuales llamados `vhost1.domain.com` y `vhost2.domain.com`. Asegúrese de reemplazarlos con los nombres que utilice. La tabla 6.1 muestra los ID de usuario y de grupo que tiene que se utilizan para estos dominios virtuales.

Tabla 6.1. Las ID de usuario y de grupo en los host virtuales

Host virtual	ID de usuario	ID de grupo
vhost1.domain.com	vh1user	vh1group
vhost2.domain.com	vh2user	vh2group

Se puede configurar Apache para que soporte distintos ID de usuario y de grupo para cada host virtual del siguiente modo:

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

```
ChildPerUserID 10 vh1user vh1group
ChildPerUserID 10 vh2user vh2group
```

   La directiva `ChildPerUser` le dice a Apache que asocie diez procesos hijo (cada uno con varios hilos que sirven las solicitudes) al ID de usuario y de grupo del sitio `vhost1.domain.com`. Del mismo modo, la segunda línea asocia otros diez procesos hijo al ID de usuario y de grupo del sitio `vhost2.domain.com`.

2. Cree un contenedor `<VirtualHost>` para cada sitio:

```
NameVirtualHost 192.168.1.100

<VirtualHost 192.168.1.100>
  ServerName vhost1.domain.com
  AssignUserID vh1user vh1group
  #
  # Aquí van otras directivas
  #
   </VirtualHost>

   <VirtualHost 192.168.1.100>
  ServerName vhost2.domain.com
  AssignUserID vh2user vh2group

  #
  # Aquí van otras directivas
  #
</VirtualHost>
```

   La directiva `AssignUserID` utilizada en cada configuración del host virtual, le dice a Apache que asocie cada uno de los host virtuales al ID de

usuario y de grupo apropiados. No olvide cambiar la dirección IP en caso necesario.

3. Reinicie el servidor Apache utilizando el comando de reinicio `/usr/local/apache/bin/apachectl`.

Gestionar un gran número de host virtuales

Si tiene muchos host virtuales, el archivo `httpd.conf` puede convertirse en un archivo muy grande y difícil de manejar. Una solución sencilla es poner cada host virtual en un archivo e incluir cada archivo utilizando la directiva `Include`. Por ejemplo, el listado 6.4 muestra un archivo `httpd.conf` con dos configuraciones de host virtual que son externas al archivo de configuración principal.

Listado 6.4. httpd.conf con dos configuraciones externas de host virtuales

```
ServerType standalone
ServerRoot "/usr/local/apache"
PidFile /usr/local/apache/logs/httpd.pid
ScoreBoardFile /usr/local/apache/logs/httpd.scoreboard
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 5
MaxSpareServers 10
StartServers 5
MaxClients 10
MaxRequestsPerChild 0
Port 80
User httpd
Group httpd
ServerName www.domain.com
ServerAdmin webmaster@domain.com
DocumentRoot "/www/mysite/htdocs"

# Nombre de los Hosts virtuales
NameVirtualHost 192.168.1.100
Include vhost1.domain.com.conf
Include vhost2.domain.com.conf
```

Tenga en cuenta que la directiva `Include` sigue a la directiva `NameVirtualHost`. La directiva `Include` le dice a Apache que lea los archivos `vhosts1.domain.com.conf` y `vhost2.domain.com.conf` para cargar la configuración del host virtual. Estos archivos pueden tener los contenedores `<VirtualHost>` que normalmente pone en `httpd.conf`. El beneficio de esta aproximación es que puede crear una configuración muy elaborada para cada host virtual sin complicar el archivo `httpd.conf`.

Configuración automática de host virtuales utilizando mod_perl

Aunque el uso de la directiva `Include` (discutida en la sección anterior) ayuda a simplificar `httpd.conf`, sigue sin ser una solución lo suficientemente buena para sitios con muchos servidores Web, que ejecutan muchos sitios Web virtuales. Lo ideal es que escriba la configuración de Apache con un programa o un script, si es posible, de modo que se creen todas esas opciones de configuración habituales de forma automática.

Si utiliza el módulo `mod_perl` con Apache (capítulo 16), puede escribir código Perl para generar su configuración Apache. Como la compilación y la instalación de `mod_perl` se realiza en el capítulo 16, no lo vamos a repetir aquí.

Podrán acudir a capítulos siguientes, en caso de que necesite tener compilado e instalado soporte `mod_perl` en Apache. Recuerde que cuando compila `mod_perl` siguiendo las instrucciones del capítulo 16, tiene que utilizar la opción EVERYTHING=1 o la opción PERL_SECTIONS=1 con el script de configuración. Para utilizar código Perl para generar la configuración Apache, necesita utilizar el contenedor `<Perl>` en `httpd.conf`. Por ejemplo:

```
<Perl>

  #
  # Aquí se coloca su código Perl
  #

  1;
</Perl>
```

Este contenedor Perl es el mínimo absoluto que debe tener en `httpd.conf` para ser capaz de utilizar la configuración basada en Perl. La última línea dentro del contenedor devuelve 1 (valor verdadero), que es el valor necesario para satisfacer `mod_perl`. Su código puede ser cualquier script de Perl. El código en un contenedor `<Perl>` se compila en un paquete especial y `mod_perl` comunica la información de configuración al módulo de configuración general de Apache. La sintaxis que describe la configuración, se discute más tarde.

Las directivas que toman un solo valor se representan con variables escalares (concepto de Perl). Por ejemplo:

```
User httpd
```

Esta directiva `User` toma una sola cadena como valor y, por lo tanto, se puede escribir del siguiente modo:

```
<Perl>
  $User = "httpd";
```

```
  1;
</Perl>
```

A continuación tenemos un ejemplo de configuración:

```
<Perl>
    $User = "httpd";
    $Group = "httpd";
    $ServerAdmin      = 'kabir@mobidac.com';
    $MinSpareServers = 5;
    $MaxSpareServers = 5;
    $MaxClients = 40;
    1;
</Perl>
```

Las directivas que necesitan varios valores, se pueden representar como listas. Por ejemplo, `PerlModule Apache::TestOne Apache::TestTwo`, se puede representar del siguiente modo:

```
@PerlModule = qw(Apache::TestOne Apache::TestTwo );
```

Los contenedores se representan utilizando un hash (es una construcción de programación de ordenadores que se utiliza mucho en Perl y en otros lenguajes modernos como C). Por ejemplo:

```
<VirtualHost 206.171.50.50>
    ServerName www.nitec.com
    ServerAdmin kabir@nitec.com
</VirtualHost>
```

se puede representar del siguiente modo:

```
$VirtualHost{"206.171.60.60"} = {
     ServerName => 'www.nitec.com',
     ServerAdmin => 'kabir@nitec.com'
 }
```

Un ejemplo ligeramente más complicado sería el siguiente:

```
$Location{"/some_dir_alias/"} = {
     AuthUserFile => '/www/nitec/secret/htpasswd',
     AuthType => 'Basic',
     AuthName => 'Subscribers Only Access',
     DirectoryIndex => [qw(welcome.html welcome.htm)],
     Limit => {
        METHODS => POST GET',
        require => 'user reader'
     }
 };
```

En este ejemplo, el segmento de configuración se utiliza para crear un área restringida, utilizando el método de autentificación HTTP básica, para un alias

llamado `/some dir alias/`. Además, se define un conjunto de nombres de archivo como índices de directorio para este alias.

Puede definir otros contenedores como `<Directory>`, `<Files>`, de esta misma forma. A continuación se presentan algunos ejemplos de Web, `host_a`, `host_b` y `host_c`, y se determina que el `host_a` es más poderoso que el `host_b`, y el `host_b` más que el `host_c`. Siga este ejemplo del archivo `httpd.conf`, para definir un contenedor `<Perl>` que le permita crear una sola configuración para los tres host:

```
<Perl>
# Obtiene el nombre del host utilizando la herramienta Unix
hostname
# y lo almacena en la variable $thisHost.
my $thisHost = '/bin/hostname';
if ($thisHost =~ /host_a/) {
    # la configuración de host_a se coloca aqui
    $MinSpareServers = 10;
    $MaxSpareServers = 20;
    $StartServers  = 30;
    $MaxClients = 256;
    }
elsif ($thisHost =~ /host_b/) {
    # la configuración de host_b se coloca aquí
    $MinSpareServers = 5;
    $MaxSpareServers = 10;
    $StartServers  = 10;
    $MaxClients = 50;
    }
else {
    # la configuración de host_c se coloca aquí
    $MinSpareServers = 3;
    $MaxSpareServers = 5;
    $StartServers  = 5;
    $MaxClients = 30;
        }
1;
</Perl>
```

Para hacer este escenario más interesante, supongamos que tiene distintos host virtuales para cada uno de los tres host y que le gustaría configurarlos de un modo elegante. Por ejemplo:

```
<Perl>
# Obtiene el nombre del host utilizando la herramienta Unix
hostname
# y lo almacena en la variable $thisHost.

    my $thisHost = '/bin/hostname';
    my $thisDomain = 'mydomain.com';
    my @vHosts = ();
```

```
    my $anyHost;

    if ($thisHost =~ /(host_a)/) {

        # la configuración de host_a se coloca aquí
        @vHosts = qw(gaia, athena, romeo, juliet, shazam);

    } elsif ($thisHost =~ /host_b/) {

        # la configuración de host_b se coloca aquí
        @vHosts = qw(catbart, ratbart, dilbert);

    } else {

        # la configuración de host_c se coloca aquí
        @vHosts = qw(lonelyhost);

    }

    for $anyHost (@vHosts) {
        %{$VirtualHost{"$anyHost.$domainName"}} = {
              "ServerName"  => "$anyHost.$domainName",
              "ServerAdmin" => "webmaster\@$anyHost.$domainName"
         }
    }

    1;

</Perl>
```

Una vez que ha creado una configuración basada en Perl adecuada para sus servidores Apache, puede comprobar su sintaxis en el código para asegurarse de que el código es correcto sintácticamente, ejecutando el comando `/usr/local/apache/bin/apachectl configtest`. Si hay un error de sintaxis, verá un mensaje de error en la pantalla. Corrija el error y vuelva a utilizar este comando para asegurarse de que Apache acepta el código.

Generar la configuración de host virtuales utilizando el script makesite

Si añadir host virtuales se ha convertido para usted en un problema diario o semanal, debido a que trabaja para un gran ISP u organización, cuyo departamento de marketing decide crear nuevos sitios Web con frecuencia, necesita el `makesite`. Se trata de un sencillo script de Perl que escribí hace años y que sigo utilizando para crear host virtuales. Este script se encuentra en el CD-ROM y, además, se puede bajar de `http://sourceforge.net/projects/mkweb/`.

Por ejemplo, para crear un nuevo host virtual llamado `newsite.com`, puede ejecutar `makesite newsite.com` y el script crea y adjunta la configuración `httpd.conf` necesaria. Además, creará los archivos de configuración DNS necesarios, utilizando dos plantillas. Para utilizar `makesite`, siga estos pasos.

1. Copie los script `makesite` y los archivos `named.template` y `httpd.template`, en una localización apropiada en su servidor. Normalmente yo guardo el `makesite` en el directorio `/usr/bin`, por lo que se encuentra en mi ruta normal. Recomiendo que cree el directorio `/var/makesite` y que sitúe los archivos `named.template` y `httpd.template` en ese directorio. En los siguientes pasos, supongo que lo ha hecho.
2. Utilice su editor de texto preferido y modifique el script `makesite`. Tiene que modificar una o más líneas de las siguientes:

```
my $MAKESITE_DIR        = '/var/makesite';
my $USER                = 'httpd';
my $GROUP               = 'httpd';
my $PERMISSION          = '2770';
my $BASE_DIR            = '/www';
my $HTDOCS              = 'htdocs';
my $CGIBIN              = 'cgi-bin';
my $NAMED_PATH          = '/var/named';
my $NAMED_FILE_EXT      = '.zone';
my $NAMED_TEMPLATE      = "$MAKESITE_DIR/named.template";
my $NAMED_CONF          = '/etc/named.conf';
my $HTTPD_CONF          = '/usr/local/apache/conf/httpd.conf';
my $VHOST_TEMPLATE      = "$MAKESITE_DIR/httpd.template";
my $LOG_FILE            = "$BASE_DIR/makesite.log";
```

Puede necesitar realizar los cambios siguientes:

- Si siguió mi recomendación y creó `/var/makesite` para guardar los archivos `named.template` y `httpd.template`, no necesita realizar ningún cambio en `$MAKESITE_DIR`.
- Si ejecuta Apache utilizando un usario/grupo distinto de `httpd`, cambie los valores de `$USER` y `$GROUP`. Si quiere que los permisos por defecto del directorio Web sean distintos de `2770`, entonces cambie el valor `$PERMISSION`.
- uta Apache utilizando un usario/grupo distinto de `httpd`, cambie los valores de `$USER` y `$GROUP`. Si quiere que los permisos por defecto del directorio Web sean distintos de `2770`, entonces cambie el valor `$PERMISSION`.
- Si quiere que los directorios de los documentos raíz sean distintos de `htdocs`, cambie `$HTDOCS`.

- Del mismo modo, si no quiere utilizar el alias tradicional `/cgi-bin/` para `ScriptAlias`, cambie `$CGIBIN`.
- En los sistemas Linux, el directorio de registros DNS por defecto es `/var/named`; si lo ha cambiado de ruta, asegúrese que cambia `$NAMED_PATH`.
- Si mantiene `httpd.conf` en una ruta distinta a `/usr/local/apache`, cambie el valor de `$HTTPD_CONF`.

3. Modifique `/var/makesite/named.template` para que refleje el nombre del servidor y los nombres de los servidores Web para su sistema. El archivo por defecto `named.template` supone que los nombres de sus servidores principal y secundario son `ns1.domain.com` y `ns2.domain.com`; que el nombre de su servidor de correo es `mail.domain.com`; y que su servidor Web es `www.domain.com`. Asegúrese de cambiar estos nombres de acuerdo con su configuración.
4. Asegúrese de que `/var/makesite/httpd.template` tiene todas las directivas que quiere añadir a cada nuevo host virtual que ha creado utilizando este script.

Ahora está preparado para ejecutar el script con el fin de crear un nuevo host virtual. Como precaución, haga una copia de seguridad de los archivos `/etc/named.conf` y `/usr/local/apache/httpd.conf`. Para crear un nuevo host virtual llamado `vhost1.com` ejecute el comando `makesite vhost1.com`.

Examine el archivo `/usr/local/httpd.conf`; debería ver:

```
#
# Configuración de www.vhost1.com
#
<VirtualHost www.vhost1.com>
  ServerName www.vhost1.com
  ServerAdmin webmaster@vhost1.com

  DocumentRoot /tmp/vhost1/htdocs
  ScriptAlias /cgi-bin/ /tmp/vhost1/cgi-bin/

  ErrorLog logs/www.vhost1.com.error.log
  TransferLog logs/www.vhost1.com.access.log

</VirtualHost>

#
# Final de la configuración de www.vhost1.com
#
```

Esta es la configuración `<VirtualHost>` que crea el script `makesite`.

TRUCO: Si está creando nombre de host virtuales por primera vez, asegúrese de que añade `NameVirtualHost IP_Address` como última línea en el archivo `httpd.conf` antes de ejecutar el script `makesite` por primera vez.

Además, compruebe el archivo `/etc/named.conf`; debería ver lo siguiente:

```
// vhost1.com was created on 2001-04-25-17-25
zone "vhost1.com" {
        type master;
        file "vhost1.zone";
};
```

Como puede ver, el script crea la información de configuración de la zona apropiada para el host virtual.

Puede encontrar la información sobre la zona DNS en el archivo `/var/named/vhost1.zone`, que muestra:

```
@    IN SOA vhost1.com.  hostmaster.vhost1.com. (
                        20010425000    ; serial YYYYMMDDXXX
                        7200        ; refresh
                        3600        ; (1 hour) retry
                        1728000     ; (20 days) expire
                        3600)           ; (1 hour) minimal TTL

; Nombre de los servidores
    IN  NS  ns1.domain.com.
    IN  NS  ns2.domain.com.
    IN  MX  10 mail.domain.com.

; registros CNAME
www   IN   CNAME   www.domain.com.
```

Mire en /www (o en cualquier $`BASE_DIR`); debería ver el directorio `vhost1` con los archivos y los subdirectorios siguientes:

```
./vhost1
./vhost1/htdocs
./vhost1/htdocs/index.html
./vhost1/cgi-bin
```

Reinicie el servidor utilizando el comando `killall -HUP named`. Además, reinicie el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl restart`. Debería ser capaz de acceder al nuevo sitio Web virtual utilizando `www.vhost1.com/`.

La página por defecto `index.html` se encuentra allí para ayudarle a identificar el dominio en la Web.

Gestionar host virtuales utilizando MySQL con el módulo mod_v2h

El módulo `mod_v2h` es un módulo de hospedaje en masa de host con soporte para realizar traducción de rutas URL desde la base de datos MySQL. Este módulo, puede cachear traducciones de rutas URL en la memoria para aumentar la velocidad. Necesitará tener instalado MySQL en un servidor. Para aprender algo sobre MySQL visite `www.mysql.com`.

Con MySQL instalado, siga los pasos siguientes para compilar e instalar `mod_v2h`:

1. Baje la última versión de `mod_v2h` de `www.fractal.net/mod_v2h.tm`.
2. Como raíz, extraiga la distribución de la fuente utilizando el comando `tar xvzf mod_v2h.tar.gz` en el subdirectorio de módulos del árbol fuente de Apache. Por ejemplo, si mantiene la fuente Apache en el directorio `/usr/local/src/httpd_2_0_16`, entonces extraiga el archivo `mod_v2h.tar.gz` al directorio `/usr/local/src/httpd_2_0_16/modules`. Se crea un subdirectorio nuevo llamado `mod_v2h`.
3. Cambie su directorio a `/usr/local/src/httpd_2_0_16/modules/v2h` y edite el archivo config.m4.

NOTA: Únicamente tiene que editar este archivo si nota que las rutas de los directorios `include` y `lib` para los archivos MySQL son incorrectas. Por ejemplo, en mi sistema Linux, los archivos `include` MySQL están instalados en el directorio `/usr/include/mysql` y los archivos `library` están en el directorio `/usr/lib/mysql`. El valor por defecto `config.m4` señala a `/usr/local/include/mysql` y a `/usr/local/lib/mysql` para los archivos `include` y `library`, respectivamente; por lo que, tuve que corregir la ruta. Edite las rutas si es necesario. Yo tuve que añadir también `-lz` en la línea `LDFLAGS` por lo que quedó del siguiente modo `LDFLAGS="$LDFLAGS -L/usr/lib/mysql -lz -Wl,-R,/usr/lib/mysql"`.

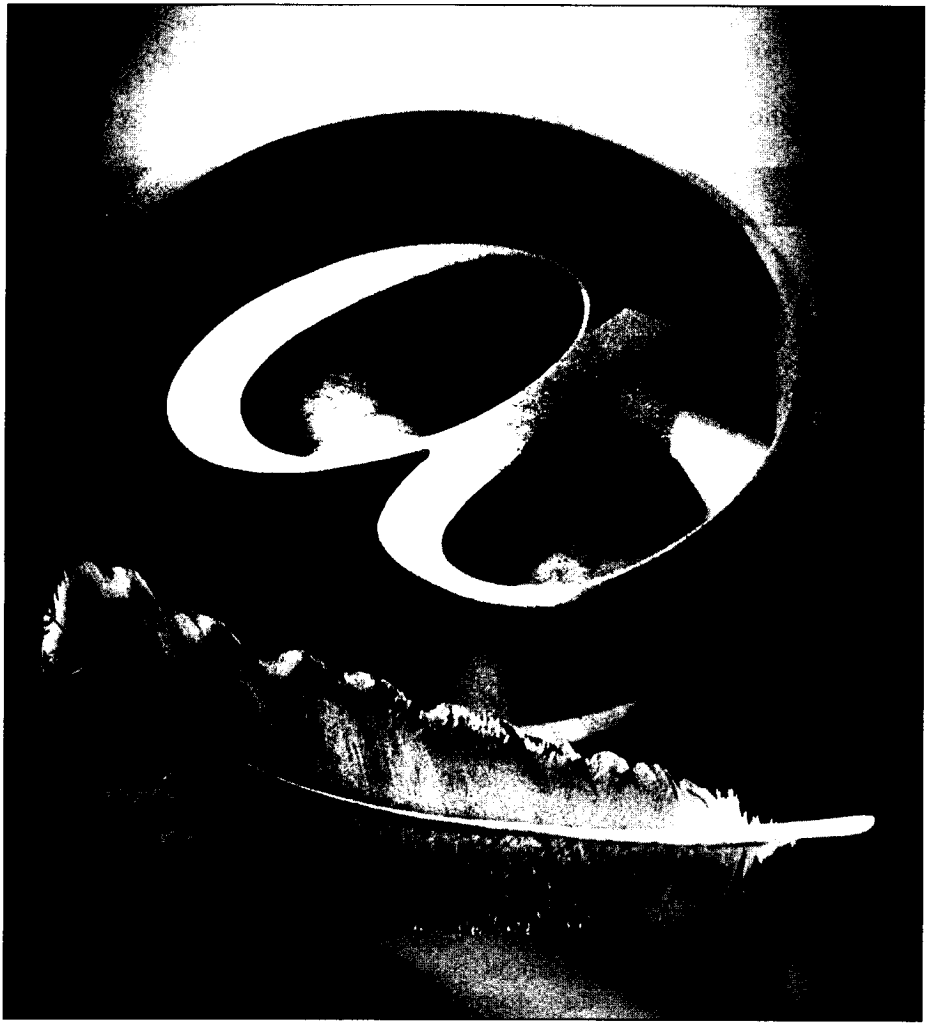
4. Cambie el directorio a `/usr/local/src/httpd_2_0_16` y ejecute el comando `autoconf` para crear el archivo de configuración necesario.
5. Ejecute `./configure` con cualquiera de las opciones que necesite, tal y como se muestra en el capítulo 2, y entonces ejecute los comandos de instalación `make && make` para compilar e instalar Apache con soporte `mod_v2h`. Por ejemplo, yo ejecuté `./configure --prefix=/usr/`

`local/httpd --disable-module=cgi` para configurar la fuente Apache y entonces ejecuté los comandos de instalación `make && make` para instalar Apache en `/usr/local/httpd` con soporte `mod_v2h` y sin soporte `mod_cgi`. Puede utilizar cualquiera de las opciones que ha utilizado antes, ejecutando `./config.status` en lugar de `./configure`.

Una vez que ha compilado Apache con `mod_v2h`, tiene que utilizar las directivas `mod_v2h`, que se muestran en la tabla 6.2, en `httpd.conf`.

Tabla 6.2. Las directivas de mod_v2h

Directiva	Función
`v2h`	Asigna el valor On para activar este módulo. En caso contrario asigna el valor Off.
`v2h_Mysql_Db`	Asigna el nombre de la base de datos MySQL.
`v2h_Mysql_Tbl`	Asigna el nombre de la tabla de la base de datos MySQL.
`v2h_Mysql_Serv_Fld`	Asigna el nombre del campo de la tabla en la que está almacenado el nombre del servidor (por ejemplo, `www.domain.com`).
`v2h_Mysql_Path_Fld`	Especifica la ruta física del URI al que ha de ser traducido (es decir, `/htdocs/www.fractal.net/`).
`v2h_Mysql_Host`	Asigna el nombre del host del servidor MySQL.
`v2h_Mysql_Port`	Asigna el número de puerto de MySQL que se está utilizando para conectarse.
`v2h_Mysql_Pass`	Asigna la contraseña (si hay) para el acceso a la base de datos.
`v2h_Mysql_User`	Asigna el nombre de usuario (si hay) para el acceso a la base de datos.
`v2h_Mysql_Env_Fld`	Asigna el valor de un campo adicional de la base de datos que se utiliza para asignar una variable de entorno llamada `VHE_EXTRA`.
`v2h_PathHead`	Asigna la ruta extra que puede ser prefijada por la ruta que señala a `v2h_Mysql_Path_Fld`.
`v2h_UseImage`	Fija el valor `On` o el valor `Off` para permitir o desactivar el caching en la memoria.
`v2h_ImagePath`	Fija la ruta para almacenar la imagen de memoria.
`v2h_DeclineURI`	Fija el URI que será rechazado.



7 Autentificación y autorización de visitantes al sitio Web

En este capítulo

1. Autentificamos usuarios utilizando nombres de usuario y contraseñas.
2. Autorizamos el acceso utilizando el nombre del host o la dirección IP.
3. Autentificamos con RDBM.
4. Autentificamos utilizando la base de datos MySQL.
5. Autentificamos utilizando el archivo de contraseñas `/etc/passwd`.
6. Autentificamos utilizando cookies.

Se ha estado dando vueltas mucho tiempo al soporte para la autentificación HTTP básica en Apache. Se han escrito muchos módulos para proporcionar autentificación básica HTTP Apache. En este capítulo, vamos a ver los distintos tipos de autentificación y la forma de autorizar el acceso al servidor Web, cómo autentificar usuarios utilizando archivos de contraseña, servidores de bases de datos, etc., y cómo controlar el acceso restringiendo el acceso mediante direcciones IP o nombres de usuario.

Autentificación vs. autorización

Hay mucha gente que confunde autentificación y autorización, y algunos, incluso, piensan que se trata del mismo concepto y no es cierto. Para entender la diferencia, considere el siguiente ejemplo. Cuando quiere visitar otro país necesita un pasaporte y una Visa. El pasaporte es un documento que le autentifica en otro país. Confirma que somos quienes decimos ser. Por lo tanto, cuando presenta su pasaporte, se está identificando ante, por ejemplo, un policía. A continuación, debe probar que está capacitado (es decir, tiene permiso) para entrar en ese país. El documento que se lo permite es la Visa. Ahora, en términos de computación, la autentificación normalmente implica la introducción de un nombre y una contraseña. Una entrada y una aceptación con éxito, exigen que seamos quien decimos ser. En otras palabras, tenemos que autentificarnos.

Para acceder a un recurso determinado va a necesitar autorización y autentificación. Por ejemplo, si está accediendo a un ordenador a las 4 de la mañana, el ordenador puede no permitirle el acceso a esa hora porque el administrador del sistema lo ha decidido así. De igual modo, podría estar autorizado para ver un sitio Web restringido desde la oficina pero no desde casa porque la política de la compañía dicta al administrador de red que todos los accesos restringidos sean realizados en el propio local.

Entender cómo funciona la autentificación

La autentificación HTTP básica es realmente muy sencilla. Se utiliza un mecanismo de intento y respuesta para autentificar usuarios. Los pasos se muestran en la figura 7.1 y se discuten a continuación:

1. La autentificación comienza cuando un navegador Web solicita una URL que está protegida por un esquema de autentificación HTTP. Esto se muestra en la figura con el número (1).
2. El servidor Web devuelve entonces una cabecera de estado 401 junto con una cabecera WWW-Authenticate, que implica que se requiere autentificación para acceder a la URL. La cabecera contiene el esquema de autentificación que se está utilizando (actualmente sólo se soporta la autentificación HTTP básica) y el nombre real. Esto se muestra en la figura con el número (2).
3. En este momento, aparece una caja de diálogo en el navegador Web, pidiéndole al usuario que introduzca un nombre de usuario y una contraseña. Esto se muestra en la figura con el número (3).
4. El usuario introduce el nombre de usuario y la contraseña y hace clic en OK. El navegador envía entonces el nombre de usuario y la contraseña

junto con la URL solicitada al servidor. El servidor comprueba si son válidos el nombre de usuario y la contraseña. Esto se muestra en la figura con el número (4).

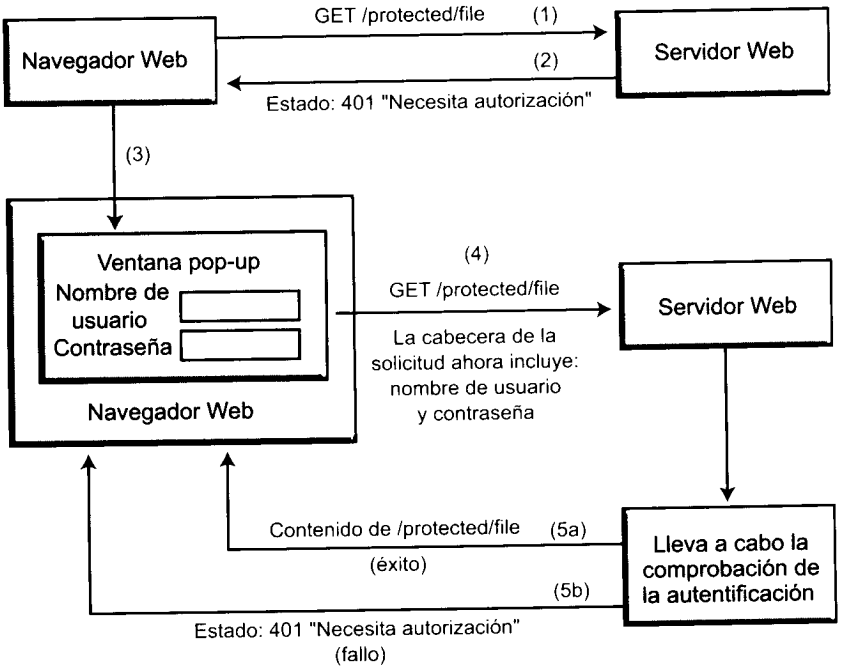


Figura 7.1. El proceso básico de autentificación HTTP

ADVERTENCIA: Cuando se envía la contraseña desde el sistema cliente (el host que se ejecuta en el navegador Web), no se envía como texto sino encriptado. Por el contrario, se envía como un archivo uuencoded y se transmite por Internet. Esta es la desventaja de este método de autentificación, porque cualquier persona que tenga un monitorizador de paquetes de datos circulantes o sniffer ya a ser capaz de recuperar el paquete IP que transporta la contraseña codificada. Como el esquema de codificación de datos uuencode es muy utilizado, también está disponible el codificador uuencode, de modo que prácticamente cualquiera puede decodificar una contraseña uuencode y utilizarla. Es cierto que el sniffer de paquetes ha de ser capaz de encontrar el paquete capaz de realizar la decodificación, pero lo cierto es que técnicamente posible. Este es el motivo por el cual, nunca debe utilizar la autentificación HTTP `Basic` para una aplicación crítica. Por ejemplo, no debería utilizar este tipo de autentificación para proteger secretos nacionales. Sin embargo, si está preparado para permitir el acceso Telnet o a ftp en su sistema, entonces está preparado para utilizar métodos

de autentificación (en estos servicios), que son muy parecidos a la autentificación HTTP `Basic`. Si confía su servidor a una conexión Internet, abierto a intentos de entrada a Telnet por cualquiera que quiera intentarlo, entonces no tiene motivo para no confiar en este método.

5. Si el nombre de usuario y la contraseña son válidos (es decir, auténticos), el servidor devuelve la página solicitada. Esto se muestra en la figura con el número (5a). Si el nombre de usuario y la contraseña no son válidos, el servidor responde con un estado 401 y envía la misma cabecera WWW-Authenticate al navegador. Esto se muestra con el número (5b) en la figura.
6. En cada solicitud posterior al mismo servidor durante la sesión del navegador, el navegador enviará el par nombre de usuario/ contraseña, de modo que el servidor no tiene que generar una cabecera de estado 401 para llamadas en la misma área del sitio. Por ejemplo, si la URL `http://apache.nitec.com/protected/` necesita autentificación HTTP `basic`, las siguientes llamadas a `http://apache.nitec.com/protected/a_page.html` y a `http://apache.nitec.com/protected/b_page` también necesitarán el nombre de usuario y la contraseña. Este es el motivo por el cual el navegador envía ambas cabeceras, la cabecera de estado 401 y la respuesta de cabecera WWW-Authenticate, antes de cualquier otro intento de autentificación, es decir, son emitidas por el servidor. Esto es más rápido y práctico que generar un intento para cada solicitud y hacer que el usuario introduzca el nombre de usuario y la contraseña varias veces.

Autentificar usuarios mediante el módulo mod_auth

El módulo `mod_auth` es el módulo de autentificación por defecto de Apache. Este módulo le permite autentificar usuarios cuyas credenciales están almacenadas en archivos de texto. Normalmente, se utiliza un archivo de texto que contiene un nombre de usuario y una contraseña. También puede utilizar un archivo de texto para crear grupos de usuarios, que podemos utilizar en la creación de reglas de autorización (que se discuten más tarde en este capítulo). Para un número pequeño de usuarios se recomienda que utilice la autentificación basada en `mod_auth`. A menudo, cuando un archivo de texto alcanza tan sólo unos cuantos miles de nombres de usuarios, la realización de la búsqueda cae dramáticamente. Por lo tanto, si tiene una base de usuarios muy grande, no se recomienda este módulo. Sin embargo, este método es perfecto para alrededor de unos cien usuarios.

Puede utilizar `/usr/local/apache/bin/httpd -l` para comprobar si este módulo está compilado en su binario de Apache. En caso contrario, tiene que utilizar la opción `--enable-module=auth` con el script `configure`, y recompilar e instalar su distribución Apache.

Entender las directivas mod_auth

El módulo `mod_auth` aporta las directivas de Apache `AuthUserFile`, `AuthGroupFile` y `AuthAuthoritative`. Vamos a ver estas directivas y algunos ejemplos en los que se utiliza este módulo.

Directiva AuthUserFile

Esta directiva asigna el nombre del archivo de texto que contiene los nombres de usuario y las contraseñas utilizadas en la autentificación HTTP básica. Esta directiva requiere que proporcione un *fully qualified path* (nombre completo de la máquina incluido el dominio) al archivo, para poder utilizarlo.

> **Sintaxis:** `AuthUserFile` *`nombrearchivo`*
>
> **Contexto:** directorio, archivo de control de acceso en el ámbito de directorio (`.htaccess`)
>
> **Invalidar:** `AuthConfig`

Por ejemplo, la siguiente directiva asigna `/www/mobidac/secrets/.htpasswd` como archivo de nombre de usuario y contraseña:

```
AuthUserFile /www/mobidac/secrets/.htpasswd
```

El archivo con el nombre de usuario y la contraseña se crea normalmente utilizando una herramienta llamada `htpasswd`, que está disponible como un programa de soporte en la distribución estándar de Apache. El formato de este archivo es muy sencillo. Cada línea contiene un solo nombre de usuario y una contraseña encriptada. La contraseña está utilizando la función estándar `crypt()`.

ADVERTENCIA: Es importante que el archivo especificado por `AuthUserFile` resida fuera del árbol de documentos del sitio Web. Ponerlo dentro de un directorio Web accesible, podría permitirle bajarlo a alguien.

Directiva AuthGroupFile

Esta directiva especifica el archivo de texto que hay que utilizar como la lista de grupos de usuarios para autentificación HTTP Basic. El nombre de archivo es

la ruta absoluta del archivo de grupo. Puede crear este archivo utilizando cualquier editor de texto.

Sintaxis: `AuthGroupFile` *`nombrearchivo`*

Contexto: directorio, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `AuthConfig`

El formato de este archivo es el siguiente:

```
nombregrupo: nombreusuario nombreusuario nombreusuario [...]
```

Por ejemplo:

```
startrek: kirk spock picard data
```

Esta línea crea un grupo llamado `startrek`, que tiene cuatro usuarios: `kirk`, `spock`, `picard` y `data`. El icono de Advertencia de la sección anterior se aplica también a esta directiva.

Directiva AuthAuthoritative

Si esta utilizando más de un esquema de autentificación para el mismo directorio, puede fijar esta directiva con el valor `off` para que cuando falle el par nombre de usuario/ contraseña en el primer esquema, lo pase al siguiente nivel (inferior).

Sintaxis: `AuthAuthoritative On | Off`

Predefinido: `AuthAuthoritative On`

Contexto: directorio, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `AuthConfig`

Por ejemplo, si está utilizando el módulo `mod_auth_mysql` (que se discutirá más tarde en este capítulo) y el módulo estándar `mod_auth`, para proporcionar servicios de autentificación, y falla el par nombre de usuario/ contraseña para uno de ellos, entonces se utiliza, si es posible, el siguiente módulo para autentificar al usuario. Cuando falla el par nombre de usuario/ contraseña en todos los módulos, el servicio vuelve a editar una cabecera de estado 401 y envía una respuesta de cabecera WWW-Authenticate para volver a realizar una autentificación. Sin embargo, si un módulo determinado autentifica con éxito un par nombre de usuario/ contraseña, los módulos de nivel inferior nunca reciben el par nombre de usuario/ contraseña.

Se recomienda que deje el valor por defecto porque no debería diseñar un esquema de autentificación gradual en el que un usuario pasa al siguiente.

Crear una sección sólo de miembros en su sitio Web

Utilizando las directivas `mod_auth` puede crear una sección sólo de miembros en su sitio Web que necesite autentificación basada en nombre de usuario/contraseña.

Por ejemplo, vamos a suponer que quiera crear una sección sólo de miembros llamada `http://your_server_name/memberonly`. A continuación se presentan lo pasos que hay que seguir.

1. Tiene que determinar el directorio físico al que quiere restringir el acceso: la mayoría de la gente utiliza un directorio dentro del directorio especificado por `DocumentRoo`, pero puede utilizar el directorio que quiera siempre que el usuario Apache (fijado por la directiva `User`) tenga permiso para leer el contenido del directorio. Voy a suponer que su `DocumentRoot` está asignado en `/www/mysite/htdocs` y que quiere restringir el acceso al directorio llamado `/www/mysite/htdocs/memberonly`.

2. Modifique el archivo `httpd.conf` para crear un alias llamado `/memberonly/`, como se muestra a continuación:

```
Alias /memberonly/ "/www/mysite/htdocs/memberonly/"
```

3. A continuación, añada las directivas siguientes al archivo `httpd.conf` para establecer `/memberonly/` como una sección restringida que requiere autentificación de usuario.

```
<Location /memberonly/>
  AuthName "Member-Only Access"
  AuthType  Basic
  AuthUserFile  /www/secrets/.members
  require valid-user
</Location>
```

   Aquí, la directiva `AuthName` simplemente crea una etiqueta que el navegador Web despliega a los usuarios. Esta etiqueta tiene que ser significativa para que los usuarios sepan qué es lo que se está solicitando. Asegúrese de que utiliza las dobles comillas como se muestra arriba. `AuthType` siempre se fija en `Basic` porque HTTP sólo soporta autentificación `Basic` por defecto. El `AuthUserFile` señala a un archivo de contraseñas llamado.`members`. La directiva `Require` exige que sólo tengan acceso los usuarios válidos.

4. Ahora, utilice la herramienta `htpasswd` para crear un archivo de contraseñas. Suponiendo que tiene instalado Apache en `/usr/local/apache`, debe ejecutar el comando `htpasswd` del modo siguiente en el caso de que sea la primera vez que lo ejecuta:

```
/usr/local/apache/bin/htpasswd -c path_to_password_file
username
```

La opción `-c` sólo se necesita para crear el archivo y sólo se debe utilizar una vez. Por ejemplo, para crear el primer usuario llamado `mrbert` para la configuración `/memberonly/`, ejecute:

```
/usr/local/apache/bin/htpasswd -c /www/secrets/.members
mrbert
```

5. Para comprobar que se ha creado el usuario en el archivo de contraseñas, vea su contenido utilizando el editor de texto. Además ha de estar seguro de que sólo el usuario Apache (asígnelo utilizando la directiva `User`) puede acceder a este archivo. Por ejemplo, si ejecuta Apache como usuario `httpd`, puede ejecutar los comandos `chown httpd:httpd /www/secrets/.members && chmod 750 /www/secrets/.members` para permitir la lectura de este archivo unicamente al usuario `httpd` (y al grupo).
6. Reinicie el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl` restart.
7. Ahora, utilice `http://your_server_name/memberonly/` para acceder a la sección sólo de miembros; le pedirá el nombre de usuario y la contraseña. Debería ver el valor de `AuthName` ("Member-Only Access") en la caja de diálogo desplegada.
8. Introduzca un nombre de usuario y una contraseña que no sean válidos y verá un mensaje de rechazo.
9. Para terminar, intente acceder al sitio de nuevo e introduzca un nombre de usuario y una contraseñas válidos haya creado por la herramienta `htpasswd`. Debería tener acceso a la sección restringida.

NOTA: Si está utilizando el formato de registro habitual para el acceso de registro, puede ver los nombres de usuario registrados en sus archivos de registro.

Crear una sección sólo de miembros utilizando un archivo .htaccess

Cuando tenemos organizaciones del tipo Internet Service Providers (ISP) y grandes compañías con muchos departamentos ejecutando sitios Web virtuales en el mismo servidor Web, añadir configuración sólo de usuarios en `httpd.conf` (que se discutió en la última sección), es posible que no sea una buena solución,

porque tendrá que añadir o eliminar configuraciones tan rápidamente como los usuarios (en caso de un sistema ISP) soliciten esos cambios. Utilizando una autentificación basada en `.htaccess`, sin embargo, puede permitir a un usuario o a un departamento crear tantas secciones sólo de miembros como ellos quieran sin su intervención, una inestimable ayuda para un administrador de sistemas muy ocupado.

Para utilizar la autentificación basada en `.htaccess` para autentificación sólo de miembros, siga los siguientes pasos.

1. Añada la siguiente directiva en su archivo `httpd.conf`:

```
AccessFileName .htaccess
```

NOTA: Si desea permitir la autentificación basada en `.htaccess` sólo en un host virtual, añada esta directiva dentro del contenedor `<VirtualHost>` apropiado.

2. Cambie la siguiente configuración por defecto:

```
<Directory />
  Options FollowSymLinks
  AllowOverride None
</Directory>
```

a:

```
<Directory />
  Options FollowSymLinks
  AllowOverride AuthConfig
</Directory>
```

Esto permite utilizar las directivas de autorización (`AuthDBMGroupFile`, `AuthDBMUserFile`, `AuthGroupFile`, `AuthName`, `AuthType`, `AuthUserFile`, `Require`, y similares) en un archivo `.htaccess`.

3. Reinicie el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl`.

4. Ahora puede crear un archivo `.htaccess` en cualquier directorio Web accesible y controlar el acceso. Necesita tener estas directivas en el archivo `.htaccess`:

```
AuthName "Enter Appropriate Label Here"
AuthType  Basic
AuthUserFile  path_to_user_password_file
Require valid-user
```

Por ejemplo, imagine que tiene un directorio, llamado `/www/mysite/htdocs/asb` y que quiere restringir el acceso a este directorio a los

usuarios que se encuentran en la lista `/www/mysite/secrets/users.pwd`. Para hacerlo, debería utilizar la siguiente configuración:

```
AuthName "ASB Member Only Access"
AuthType  Basic
AuthUserFile   /www/mysite/secrets/users.pwd
Require valid-user
```

NOTA: Asegúrese de que únicamente el usuario Apache puede leer el archivo `.htaccess` (hágalo utilizando la directiva `User`). Por ejemplo, si ejecuta Apache como `httpd`, entonces debería ejecutar los comandos `chown httpd:httpd .htaccess && chmod 750 .htaccess` del directorio en el que ha guardado el archivo. Tenga en cuenta, además, que la creación o modificación de un archivo `.htaccess` no necesita reiniciar el servidor Apache, por lo tanto, puede poner a prueba la sección restringida de su sitio Web para determinar si el proceso de autentificación está funcionando adecuadamente.

Agrupar usuarios para accesos restringidos a distintas secciones Web

Si sus usuarios necesitan tener acceso a distintas partes de su sitio Web, tiene varias opciones. En lugar de una configuración de usuario válida, que abra la sección restringida para todos los usuarios válidos, puede utilizar nombres de usuarios específicos. Por ejemplo:

```
<Location /financial>
AuthName "Members Only"
AuthType  Basic
AuthUserFile /www/mysite/secrets/.users.pwd
require cgodsave jolson
 </Location>

<Location /sales>
AuthName "Members Only"
AuthType  Basic
AuthUserFile /www/mysite/secrets/.users.pwd
require  esmith jkirk
 </Location>
```

En este caso, sólo tienen acceso a la sección `/financial`, los usuarios `cgodsave` y `jolson`, y los usuarios `esmith` y `jkirk` tienen acceso a la sección `/sales`. Sin embargo, nombrar a todos los usuarios en la configuración es muy engorroso y a menudo una tarea imposible. Una aproximación es crear cualquiera de los archivos separados de contraseñas, que harán que los segmentos de configuración anterior se conviertan en:

```
<Location /financial>
AuthName "Members Only"
AuthType  Basic
AuthUserFile  /www/mysite/secrets/.financial-team.pwd
require valid-user
 </Location>

<Location /sales>
AuthName "Members Only"
AuthType  Basic
AuthUserFile  /www/mysite/secrets/.sales-team.pwd
require valid-user
 </Location>
```

Ahora, añada únicamente los usuarios que han de añadirse a `/www/mysite/secrets/.financial-team.pwd`, en este caso, `cgodsave` y `jolson`, y añada únicamente los usuarios que deberían añadirse a `/www/mysite/secrets/.sales-team.pwd`, en este caso, `esmith` y `jkirk`.

Sin embargo, si no quiere mantener varios archivos de contraseñas, existe otra aproximación. Por ejemplo, observe los siguientes segmentos de configuración:

```
<Location /financial>
AuthName "Members Only"
AuthType  Basic
AuthUserFile  /www/mysite/secrets/.members.pwd
AuthGroupFile  /www/mysite/secrets/.groups
require group financial
 </Location>

<Location /sales>
AuthName "Members Only"
AuthType  Basic
AuthUserFile  /www/mysite/secrets/.members.pwd
AuthGroupFile  /www/mysite/secrets/.groups
require group sales
 </Location>
```

Aquí se utiliza el mismo archivo de contraseña `.members.pwd` para las dos localizaciones pero cada localización utiliza un grupo distinto. El archivo de grupo es común porque un archivo de grupo puede contener varios grupos. El archivo de grupo `/www/mysite/secrets/.groups` es un archivo de texto sencillo, que para el ejemplo anterior sería:

```
financial: cgodsave jolson
sales: esmith jkirk
```

Ahora, para añadir un nuevo usuario a un grupo no necesita cambiar el archivo `httpd.conf` (o si está utilizando los archivos `.htaccess`, los contenedores `<Location>`). Puede añadir simplemente un usuario al grupo apropiado en el archivo de grupo, una vez que ha creado la cuenta de usuario utilizando el comando `htpasswd`.

Autorizar el acceso mediante el nombre del host o las direcciones IP

En este esquema de autorización, el control de acceso está controlado por el nombre del host o por la dirección IP del host. Cuando se solicita un recurso determinado, el servidor Web comprueba si el host solicitado tiene el acceso permitido al recurso y toma una acción basada en este hecho.

La distribución estándar de Apache incluye un módulo llamado `mod_access`, que permite el control de acceso basado en el nombre del host de Internet de un cliente Web. El nombre del host puede ser un *fully qualified domain name* (FQDN), como `blackhole.mobidac.com`, o una dirección de IP, como `192.168.1.100`. El módulo aporta el soporte de control de acceso utilizando estas directivas Apache: `allow`, `deny`, `order`, `allow from env=variable` y `deny from env=variable`.

Directiva allow

Esta directiva le permite definir una lista de host (que contenga uno o más host o direcciones IP) que tienen el acceso permitido a un directorio determinado. Cuando se ha especificado más de un host o dirección IP, hay que separarlos con espacios. La tabla 7.1 muestra los valores posibles para la directiva.

Sintaxis: `allow from host1 host2 host3 ...`

Contexto: directorio, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `Limit`

Tabla 7.1. Posibles valores para la directiva Allow

Valor	Ejemplo	Descripción
All	allow from all	Esta palabra reservada permite el acceso a todos los host. El ejemplo muestra cómo utilizar esta opción.
Un FQDN de un host	allow from wormhole.mobidac.com	Sólo se permite el acceso al host que tiene el nombre de dominio (FQDN) especificado. La directiva `allow` del ejemplo, solo sólo permite el acceso a `wormhole.mobidac.com`. Tenga en cuenta que se comparan todos los componentes; `toys.com` no coincidirá con `etoys.com`.

Valor	Ejemplo	Descripción
Un nombre parcial de dominio de un host	allow from .mainoffice.mobidac.com	Únicamente los host que tienen el mismo nombre de host, tienen le el acceso permitido. El ejemplo permite que todos los host de la red `.mainoffice.mobidac.com` accedan al sitio. Por ejemplo, `developer1.mainoffice.mobidac.com` y `developer2.mainoffice.mobidac.com` tienen acceso al sitio. Sin embargo, `developer3.baoffice.mobidac.com` no tiene permitido el acceso.
Una dirección IP completa de un host	allow from 192.168.1.100	Sólo las direcciones IP especificadas tienen el acceso permitido. El ejemplo muestra la dirección IP completa (están presentes los cuatro octetos de IP), `192.168.1.100`, que tiene permiso de acceso.
Una dirección IP parcial	1: allow from 192.168.1 2: allow from 130.86	Cuando están presentes menos de cuatro octetos de una dirección IP en la directiva `allow`, la dirección IP parcial se estudia de izquierda a derecha, y los host que tienen el mismo patrón de dirección IP (es decir, pertenecen a la misma máscara de red), tienen el acceso permitido. En el primer ejemplo, todos los host con direcciones IP en el rango de 192.168.1.1 a 192.168.1.255 tienen acceso. En el segundo ejemplo, todos los host de la red tienen el acceso permitido.
Un par red/máscara de red	allow from 192.168.1.0/255.255.255.0	Esto le permite especificar un rango de direcciones IP utilizando la dirección de red y de la máscara de red. El ejemplo permite el acceso solo a los host con la dirección IP en el rango de 192.168.1.1 a 192.168.1.255.
Una especificación red/nnn CIDR	allow 206.171.50.0/24	Es similar a la entrada anterior, excepto en que la máscara de red consiste en nnn de alto nivel. El ejemplo es equivalente a permitir el acceso a los host con las direcciones IP desde 206.171.50.0/255.255. 255.0.

Directiva deny

Esta directiva es exactamente la contraria de la directiva `allow`. Le permite definir una lista de host que tienen el acceso denegado a un directorio especí-

fico. Al igual que la directiva `allow`, puede aceptar todos los valores de la tabla 7.1.

Sintaxis: `deny from host1 host2 host3 [...]`

Contexto: directorio, localización, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `Limit`

Directiva order

Esta directiva controla el modo en que Apache evalúa tanto la directiva `allow` como la directiva `deny`.

Sintaxis: `order deny, allow | allow, deny | mutual-failure`

Predefinido: `order deny, allow`

Contexto: directorio, localización, archivo de control de acceso en el ámbito de directorio (`.htaccess`)

Invalidar: `Limit`

Por ejemplo, la siguiente directiva rechaza el acceso al host `myboss.mycompany.com`, mientras permite el acceso al directorio del resto de los host. El valor de la directiva `order` es una lista separada por comas, que indica qué directiva tiene prioridad:

```
<Directory /mysite/myboss/rants>
    order deny, allow
    deny from myboss.mycompany.com
    allow from all
</Directory>
```

Normalmente, la que afecta a todos los host tiene la menor prioridad. En el ejemplo anterior, como la directiva `allow` afecta a todos los host, tiene la menor prioridad.

Aunque `allow, deny` y `deny, allow` son los valores más utilizados para la directiva order, puede utilizar otro valor, mutual-failure, para indicar que sólo los host que aparecen en la lista `allow` tienen concedido el acceso, pero no lo tienen los que se encuentran en la lista `deny`. En todos los casos, se evalúa cada directiva `allow` y cada directiva `deny`.

Directiva allow from env=variable

Esta directiva, una variación de la directiva `allow`, permite el acceso cuando se asigna la variable de entorno.

Sintaxis: `allow from env=variable`

Contexto: `directorio, archivo de control de acceso en el ámbito de directorio (.htaccess)`

Invalidar: `Limit`

Esto sólo es útil si está utilizando otras directivas como `BrowserMatch` para asignar una variable de entorno. Por ejemplo, imagine que quiere permitirle a Microsoft Internet Explorer 6, la última versión de Internet Explorer, acceder al directorio en el que almacenó algunos archivos HTML con VBScript embebido. Al igual que el resto de navegadores Web principales, como el Netscape Navigator, no soportan directamente VBScript, es preferible que no tenga usuarios de Navigator en el directorio. En este caso, puede utilizar la directiva `BrowserMatch` para asignar una variable de entorno cuando se detecta Internet Explorer 5.5. La directiva sería:

```
BrowserMatch "MSIE 5.5" ms_browser
```

Puede utilizar el contenedor `<Directory>` para especificar la directiva `allow`:

```
<Directory /path/to/Vbscript_directory >
    order deny,allow
    deny from all
    allow from env=ms_browser
</Directory>
```

En este caso, el servidor Apache asignará la variable de entorno `ms_browser` para todos los navegadores que proporcionen la cadena "MSIE 6" como parte del identificador del agente de usuario. La directiva `allow` permitirá el acceso sólo a los navegadores para los que está asignada la variable `ms_browser`.

deny from env=variable

Esta directiva, una variación de la directiva deny, deniega el acceso a todos los host para los que está asignado el entorno especificado.

Sintaxis: `deny from env=variable`

Contexto: directorio, localización, archivo de control de acceso en el ámbito de directorio `(.htaccess)`

Invalidar: `Limit`

Por ejemplo, si quiere denegar el acceso a todos los host utilizando Microsoft Internet Explorer, puede utilizar la directiva `BrowserMatch` para asignar una variable llamada `ms_browser` cada vez que un navegador se identifique al servidor con la cadena "`MSIE`".

```
BrowserMatch "MSIE" ms_browser
```

Ahora puede utilizar un contenedor <Directory> para especificar la directiva deny, del siguiente modo:

```
<Directory /path/to/vbscript_directory >
    order deny,allow
    allow from all
    deny from env=ms_browser
</Directory>
```

Si está interesado en bloquear el acceso a un método determinado de solicitudes HTTP, como GET, POST o PUT, puede utilizar el contenedor <Limit> para hacerlo. Por ejemplo:

```
<Location /cgi-bin>
    <Limit POST>
        order deny,allow
        deny from all
        allow from yourdomain.com
    </Limit>
</Location>
```

Este ejemplo permite solicitudes POST en el directorio cgi-bin únicamente si realizan las solicitudes los host del dominio yourdomain.com. En otras palabras, si este sitio tiene algún formulario HTML que envía datos introducidos por el usuario mediante el método POST HTTP, únicamente los usuarios de yourdomain.com estarían capacitados para utilizar estos formularios de forma efectiva. Normalmente, las aplicaciones CGI están almacenadas en el directorio cgi-bin, y muchos sitios presentan formularios HTML que utilizan el método POST para descargar datos a las aplicaciones CGI. Utilizando la configuración de control de acceso basada en host, un sitio puede permitir a cualquiera ejecutar un script CGI, pero sólo permitirle a un sitio determinado (en este caso, yourdomain.com) enviar realmente datos de uno o más scripts CGI. Eso le aporta al acceso CGI en este tipo de sitios un ligero carácter de sólo lectura. Todo el mundo puede ejecutar aplicaciones que generen salidas sin tomar ninguna entrada de usuario, pero únicamente los usuarios de un dominio determinado pueden proporcionar una entrada.

Combinar autentificación y autorización

La autentificación de usuario HTTP básica, soportada en mod_auth y la autorización soportada en mod_access se puede combinar para implementar los problemas prácticos en el control de acceso. Por ejemplo, imagine que quiere permitir el acceso a un grupo de usuarios a las secciones /aolbuddies/ de su sitio Web en el caso de que estén navegando por el sitio Web mediante una conexión AOL. A continuación tenemos la configuración que puede añadir a

`httpd.conf` una vez que reemplace la ruta y los nombres de usuario por los apropiados:

```
Alias /aolbuddies/    "/path/to/web/directory/for/aolbuddies/"

<Location /aolbuddies/>
   Deny from all
   Allow from .aol.com

   AuthName "AOL Buddies Only"
   AuthType  Basic
   AuthUserFile /path/to/.myusers.pwd
   AuthGroupFile /path/to/.mygroups
   require group aolbuddies

   Satisfy all
</Location>
```

La directiva `Satisfy all` le dice a Apache que sólo permita el acceso a aquéllos que pasan tanto las pruebas de autentificación como las de autorización. Cuando un usuario AOL conecta con `http://your_server/aolbuddies/` mediante AOL, se le pide al usuario que introduzca un nombre de usuario y una contraseña. Si el usuario introduce un nombre de usuario que pertenece al grupo `aolbuddies` y la contraseña del usuario es correcta, el usuario tendrá el acceso permitido.

NOTA: Debe añadir todos sus compañeros AOL como usuarios en `/path/to/.myusers.pwd` y además crear un grupo llamado `aolbuddies` en `/path/to/.mygroupsm` que tenga una lista de todos los compañeros AOL (usuarios en `/path/to/.aol` que añadió antes) en él.

Autentificación con bases de datos relacionales

Si ejecuta un servidor de bases de datos relacionales en su red (o incluso un servidor Web) y tiene muchos usuarios (es decir, más de 1000 usuarios) para autentificar mediante la Web, puede utilizar el servidor de bases de datos para reemplazar la autentificación basada en archivos de texto (`mod_auth`), que se discutió antes. Hay varias ventajas en la utilización de un servidor de bases de datos para muchos usuarios; las ventajas principales son:

- La autentificación `mod_auth` se convierte en lenta cuando hay muchos usuarios almacenados en un archivo de texto.

- Si permite a los usuarios cambiar sus contraseñas vía Web, utilizando una aplicación personalizada, los archivos de texto no son seguros porque debe asegurar que ese acceso de escritura en el archivo está bloqueado y desbloqueado adecuadamente. Almacenar los datos en los servidores de bases de datos elimina esta carga adicional en sus scripts y aporta mucho mayor grado de integridad de datos global.
- En una base de datos relacional, puede almacenar una gran cantidad de información sobre un usuario que se puede utilizar en sus aplicaciones Web. Por lo tanto, es una buena idea, en estos casos, centralizar su base de datos de usuarios utilizando un servidor de bases de datos como MySQL.

NOTA: Puede utilizar servidores de bases de datos modernos como MySQL, Postgres, DB2, Oracle y Microsoft SQL, como base de datos de usuarios. La instalación de cualquiera de estos servidores de bases de datos se aparta del objetivo de este libro. Voy a suponer que tiene instalado uno de estos servidores de bases de datos en su red o en el servidor Web.

TRUCO: Si tiene un sitio con muchos usuarios, debería crear un servidor dedicado para la base de datos que sea accesible en su red Web. De forma ideal, el servidor de bases de datos debería ser accesible mediante una red especializada en lugar de Internet. La mayor parte del tiempo, es mejor tener una segunda interfaz Ethernet en cada sistema del servidor y crear una LAN que sólo sea accesible a los servidores de la LAN. (Ver el capítulo 23 para obtener los detalles de cómo crear tal tipo de red.)

Utilizar un servidor con una base de datos MySQL para la autentificación

MySQL es el servidor de bases de datos gratuito más utilizado en la comunidad de código fuente abierto; está disponible bajo la licencia pública GNU. Es sencillo instalarlo y establecer su servidor como un servidor de bases de datos de autentificación de usuarios. Con la plataforma Linux, puede simplemente bajar e instalar el servidor, el cliente y los paquetes RPM de desarrollo y puede estar listo para usar en cuestión de minutos. El servidor MySQL está disponible en `www.mysql.com`.

Crear la base de datos de autentificación de usuarios en el servidor MySQL

Para utilizar el servidor MySQL como base de datos de autentificación, necesita tener información al menos sobre el nombre de usuario y la contraseña en

una tabla en una base de datos. Si ya tiene una tabla en la base de datos con esa información no necesita seguir los pasos siguientes.

1. Entrar en el servidor MySQL utilizando el comando `mysql -u root -p`. Le pedirá que introduzca la contraseña raíz para la base de datos.

> **ADVERTENCIA: La contraseña raíz para el servidor MySQL debería ser siempre distinta de la contraseña raíz de su sistema Unix. Sus usos son distintos y por lo tanto deben tratarse por separado.**

2. Una vez que se registra en MySQL, ejecute el comando `create database auth;` que crea una base de datos llamada `auth`.
3. Cambie su base de datos actual por la nueva base de datos `auth` utilizando el siguiente comando:
   ```
   use auth;
   ```
4. Ahora tiene que crear una tabla llamada `wwwusers` introduciendo las siguientes líneas en la entrada de comandos MySQL.

```
create table wwwusers (
  username varchar(40)    not null primary key,
  passwd   varchar(20)    not null
);
```

Cada fila de esta tabla consiste en tres campos: `username`, `passwd` y `groups`. El campo `username` es la clave primaria, que significa que MySQL utiliza este campo para indexar la tabla, realizando búsquedas que utilizan el nombre de usuario. Este campo está limitado en 40 caracteres. La utilización de varchar (carácter variable) en lugar de un carácter fijo ahorra espacio si los nombres de usuario no son siempre de 40 caracteres. El campo `username` no puede ser nulo (es decir, está vacío) porque es además la clave principal. El campo de contraseña se llama `passwd`, que tiene un máximo de 20 caracteres y es de tipo varchar. No puede ser nulo.

5. Ahora introduzca el comando `describe wwwusers`, que podría mostrar la salida siguiente:

```
mysql> describe wwwusers;
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| username | varchar(40) |      | PRI |         |       |
| passwd   | varchar(20) |      |     |         |       |
+----------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Esto confirma que la tabla `wwwusers` fue creada como queríamos.

6. Ahora necesita añadir usuarios a la tabla. Para añadir usuarios de forma manual, necesita ejecutar la siguiente sentencia SQL:

```
insert into wwwusers (username, passwd)
              values ('user_name',
                      'user_password'
                      );
```

Por ejemplo:

```
insert into wwwusers (username, passwd)
                  values ('esmith','sale007');
```

Se añade un usuario llamado `esmith` con la contraseña `sale007` al grupo `sales`. Añada tantos usuarios como quiera. Ver "Gestionar usuarios y grupos en una RDBM" para obtener los detalles sobre la gestión de usuarios utilizando scripts.

7. Si piensa utilizar grupos de usuarios para la autentificación, entonces ha de crear la tabla siguiente.

```
create table wwwgroups (
  username      varchar(40),
  groupname     varchar(40)
);
```

8. Ahora introduzca el comando `describe wwwgroups`, que debería mostrar la siguiente salida:

```
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| username  | varchar(40) | YES  |     | NULL    |       |
| groupname | varchar(40) | YES  |     | NULL    |       |
+-----------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Esto confirma que la tabla `wwwgroups` fue creada como queríamos.

9. Puede añadir usuarios que ya existen, a nuevos grupos utilizando la siguiente sentencia SQL:

```
insert into wwwgroups (username, groupname)
              values ('user_name',
                      'name_of_the_group'
                      );
```

Por ejemplo:

```
insert into wwwgroups (username, groupname)
               values ('kabir',
                       'www_wheel'
                       );
```

añade el usuario `Kabir` a un nuevo grupo llamado `www_wheel`. Tenga en cuenta que puede añadir el mismo usuario a varios grupos.

Conceder acceso al servidor Apache a la base de datos de autentificación de usuarios en MySQL

Al igual que los servidores RDBM modernos, MySQL utiliza autentificación basada en nombre de usuario y contraseña para permitir el acceso a la base de datos que almacena. Por lo tanto, antes de que pueda utilizar Apache con MySQL, debería crear un usuario MySQL llamado `httpd`, que tuviese acceso a cualquier base de datos que tenga pensado utilizar con Apache. A continuación tenemos la forma de crear un usuario MySQL para Apache:

1. Entre en el servidor MySQL utilizando el comando `mysql -u root -p`. Le pedirá que introduzca la contraseña raíz para la base de datos.
2. Una vez que está registrado en MySQL, puede editar una sentencia de permiso como la siguiente:

```
grant all privileges on name_of_database
  to username@hostname identified by 'user_password'
  with GRANT option;
```

Por ejemplo, para concederle al usuario `httpd` (es decir, el nombre de usuario especificado en la directiva `User` en el archivo `httpd.conf`) cuando utiliza la contraseña `2manysecrets`, todos los privilegios a la base de datos llamada `auth` para ejecutar el host local se utiliza:

```
grant all privileges on auth
  to httpd@localhost identified by '2manysecrets'
  with GRANT option;
```

Esto permite al usuario `httpd` acceder a la base de datos `auth` del host local. Lo cual supone que el servidor MySQL y el servidor Apache se ejecuten en la misma máquina. Si la base de datos está en una máquina distinta, debería utilizar el nombre de host adecuado como sustituto del host local.

ADVERTENCIA: Si no tiene pensado añadir aplicaciones Web que necesiten escribir en la base de datos de autentificación, no conceda `all privileges` al usuario del servidor Web. Por ejemplo, si simplemente quiere autentificar al usuario pero nunca actualiza o elimina el usuario mediante el servidor Web (es decir, utilizando un script CGI u otra aplicación Web), entonces reemplace todos los privilegios con `select`. Esto asegura que el usuario del servidor Web sólo tiene permitido realizar consultas `select` en la base de datos, que es equivalente al acceso de sólo lectura.

3. Introduzca el comando `flush privileges` para indicarle al servidor MySQL que recargue la tabla de permisos.
4. Salga del monitor del programa MySQL introduciendo `exit` en el prompt de `mysql>`.

   Introdúzcase en MySQL con el comando `mysql -u httpd -p` y proporcionando la contraseña apropiada (`2manysecrets` en este caso). Debería ser capaz de acceder a la base de datos introduciendo `use auth;` una vez en el prompt `mysql>`. Si no puede acceder a la base de datos, asegúrese de que se encuentra en el host que ha especificado en la sentencia emitida anteriormente.

Compilar e instalar el módulo mod_auth_mysql

La situación de ventaja del servidor de bases de datos MySQL en la comunidad de código fuente abierto ha descansado sobre el desarrollo de un módulo de Apache llamado `mod_auth_mysql`.

Este módulo se puede utilizar como interfaz en un servidor MySQL para la autentificación. Puede bajar la última versión de este módulo en `www.mysql.com/Downloads/Contrib/`. A continuación podemos ver cómo puede compilar e instalar `mod_auth_mysql`.

1. Extraiga la distribución de la fuente de `mod_auth_mysql` en el directorio `/usr/local/src` como raíz. Cambie el nuevo directorio mod_auth_mysql-*version* y ejecute:

   ```
   ./configure --with-apache=/usr/local/src/apache_version --
   with-mysql=/usr
   ```

   Asegúrese de cambiar el `/usr/local/src/apache_version` por la ruta de la distribución fuente de Apache y `/usr` por la ruta en la que están instaladas las cabeceras MySQL. Si tiene instalado MySQL con la configuración por defecto, los archivos de cabecera están instalados en `/usr/local/mysql` y que el `--with-mysql=/usr` suministrado es el valor correcto para este tipo de sistemas, porque el script `mod_auth_mysql` genera `/usr/local/mysql` adjuntando `/local/mysql` a `/usr`.
2. Ejecute `make`, cambie el directorio a la distribución fuente de Apache y ejecute `./config.status --activate-module=src/modules/auth_mysql/libauth_mysql.a` (si tiene compilado ya Apache) o `./configure --activate-module=src/modules/auth_mysql/libauth_mysql.a --prefix=/usr/local/apache` (si está compilando Apache por primera vez).
3. Para terminar, ejecute `make && make install` para compilar e instalar Apache con soporte `mod_auth_mysql`.

4. Reinicie el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl restart`.

Autentificar usuarios utilizando el módulo mod_auth_mysql

Una vez que tiene compilado e instalado `mod_auth_mysql`, creada la base de datos que contiene las tablas de usuarios y grupos, y creado un usuario en el nivel de bases de datos para acceder a estas tablas, puede configurar Apache del siguiente modo:

1. Añada las siguientes líneas en `httpd.conf`, fuera de cualquier `<VirtualHost>` o de cualquier otro tipo de contenedor como `<Directory>`, `<Location>` y similares.

```
Auth_MySQL_Info db_hostname db_username db_password
Auth_MySQL_General_DB database_name
```

   La primera directiva le dice a Apache cuál es el servidor de bases de datos con el que tiene que conectar y cuál es el nombre de usuario y la contraseña que tiene que utilizar. El nombre de usuario y la contraseña de la base de datos se crean en MySQL. Este par nombre de usuario- contraseña no debe coincidir con ninguna cuenta de usuario de su sistema. La segunda directiva establece la base de datos con la que se conecta. Si utiliza una sola base de datos para todas sus necesidades de autentificación, puede asignar el nombre de la base de datos aquí. De este modo evitará teclear el nombre de la base de datos en cada segmento de configuración de autentificación.

2. Para exigir autentificación para un subdirectorio de la raíz de documentos `protected_dir` puede crear un contenedor `<Directory>` o un contenedor `<Location>` en `httpd.conf`, o puede utilizar el archivo `.htaccess` (suponiendo que tiene `AllowOverride` AuthConf asignado en `httpd.conf` en el servidor principal o en un host virtual adecuado) para tener el siguiente segmento de configuración:

```
AuthName "Members Only"
AuthType Basic
require valid-user

Auth_MYSQL on
Auth_MySQL_DB database_name
Auth_MySQL_Password_Table password_table_name
Auth_MySQL_Username_Field username_field_name
Auth_MySQL_Password_Field password_field_name

Auth_MySQL_Group_Table group_table_name
Auth_MySQL_Group_Field group_field_name

Auth_MySQL_Empty_Passwords off
Auth_MySQL_Encrypted_Passwords on
```

```
Auth_MySQL_Encryption_Types Crypt_DES
Auth_MySQL_Scrambled_Passwords  off
Auth_MySQL_Authoritative  on
Auth_MySQL_Non_Persistent  off
```

NOTA: No olvide reemplazar `database_name`, `password_table_name`, `username_field_name`, `password_field_name`, `group_table_name` y `group_field_name` con la información apropiada.

- La directiva Auth_MYSQL activa o desactiva mod_auth_mysql.
- La directiva `Auth_MySQL_DB` determina el nombre de la base de datos, el cual sostiene la tabla de contraseñas determinada por `Auth_MySQL_Password_Table` y la tabla de grupos determinada por `Auth_MySQL_Group_Table`.
- Las directivas `Auth_MySQL_Username_Field` y `Auth_MySQL_Password_Field` determinan los nombres de los campos utilizados para almacenar el nombre de usuario y la contraseña en la tabla de contraseñas.
- La directiva `Auth_MySQL_Group_Field` determina el campo del nombre del grupo.
- `Auth_MySQL_Empty_Passwords` está fijada en `off` porque las contraseñas vacías no son apropiadas para la mayor parte de las necesidades de autentificación.
- El soporte de contraseñas encriptadas se activa utilizando `Auth_MySQL_Encrypted_Passwords`, y se fija el tipo de encriptación en el estilo tradicional de `Unix Crypt_DES` utilizando `Auth_MySQL_Encryption_Types`.

NOTA: Aunque pueda elegir entre los tipos de encriptación Plaintext, Crypt_DES y MySQL, nunca recomiendo la contraseña todo texto (Plaintext). El soporte de `Auth_MySQL_Scrambled_Passwords` se desactiva porque no es apropiado para la mayor parte de los escenarios.

- Como no es una buena idea permitir un esquema de autentificación gradual (es decir, si un esquema falla se utiliza otro para el mismo cliente), se activa la directiva `Auth_MySQL_Authoritative`. Esto le indica a Apache que ignore el resto de los esquemas de autentificación utilizados para el mismo directorio. Si `mod_auth_mysql` no

puede permitir a un usuario acceder a un directorio restringido, Apache volverá a editar la solicitud de autentificación.

- La directiva `Auth_MySQL_Non_Persistent`, le indica a Apache que no se desconecte del servidor de la base de datos para cada solicitud de autentificación. Desconectarse para cada solicitud significaría que Apache tendría que conectarse para cada nueva solicitud de autentificación, lo que disminuiría el rendimiento. Por lo tanto, se recomienda el valor por defecto (`off`).

3. Si añade lo que hemos visto al contenedor `<Directory>` o al contenedor `<Location>` en httpd.conf necesita reiniciar el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl restart`. Por otro lado, si utiliza esta configuración en el archivo `.htacces`, puede utilizarlo sin reiniciar el servidor.

Utilizar otras bases de datos para autentificación de usuarios

Puede utilizar Postgres, IBM DB2, Oracle u otro servidor como su base de datos, en lugar de MySQL para la autentificación de usuarios con Apache. Aunque no va a encontrar un módulo Apache como `mod_auth_mysql` para su nuevo RDBM, puede utilizar el módulo `Apache::AuthDBI` para `mod_perl` (ver el capítulo 16, en el que se discute la utilización de `mod_perl`, para obtener los detalles de cómo instalar `mod_perl`) para comunicarse con su servidor de bases de datos y para llevar a cabo la autentificación de usuario. Siga los siguientes pasos:

1. Asegúrese de que están instalados todos los archivos `library` e `include`, incluidos en el paquete del servidor de bases de datos. Normalmente, esto significa que necesita instalar el Software Development Kit (SDK) de su RDBM.
2. Instale la última versión del módulo DBI utilizando el comando `perl -MCPAN -e 'install DBI'`.
3. Instale la última versión del driver apropiado de la base de datos para Perl (DBD) utilizando el comando `perl -MCPAN -e 'install DBD::database'`. Por ejemplo, para instalar el driver de la base de datos para IBM DB2 ejecutará `perl -MCPAN -e 'install DBD::db2'`.
4. Instale la última versión de `Apache::AuthDBI` utilizando el comando `perl -MCPAN -e 'install Apache::AuthDBI'`.
5. Cree una cuenta de usuario en la base de datos de Apache para conectarse con el servidor de la base de datos. Esta no es una cuenta de acceso a su

sistema. Es una cuenta en la ingeniería de la base de datos que concede permiso de acceso a la base de datos de usuario y a sus tablas.

6. Tiene que crear la base de datos `auth` y la tabla `wwwuser` discutida en la sección "Crear una base de datos de autentificación de usuarios en el servidor mysql". Además, tiene que crear uno o más usuarios de prueba, utilizando el script `manage_users.pl` discutido en la sección "Gestionar usuarios y grupos en una RDBM".

7. Añada la siguiente línea en `httpd.conf`:

```
PerlModule  Apache::AuthenDBI
```

   Esta línea le indica a Apache que quiere utilizar el módulo `Apache::AuthenDBI`.

8. Hay que crear un alias llamado `/memberonly/` para indicar el directorio al que quiere restringir el acceso utilizando la siguiente directiva Alias:

```
Alias /memberonly/   "path_to_restricted_access_directory"
```

   Por ejemplo:

```
Alias /memberonly/ "/usr/local/apache/htdocs/protected/"
```

   Aquí, el alias `/memberonly/` está localizado en el directorio `/usr/local/apache/htdocs/protected/`.

9. A continuación, tiene que crear el siguiente segmento de configuración en `httpd.conf`:

```
<Location /memberonly/>
  AuthName "Home"
  AuthType Basic
  PerlAuthenHandler Apache::AuthenDBI
  PerlSetVar Auth_DBI_data_source dbi:mysql:database=auth
  PerlSetVar Auth_DBI_username          httpd
  PerlSetVar Auth_DBI_password          2manysecrets
  PerlSetVar Auth_DBI_pwd_table         wwwusers
  PerlSetVar Auth_DBI_uid_field         user
  PerlSetVar Auth_DBI_pwd_field         passwd
  PerlSetVar Auth_DBI_encrypted         on
  require valid-user
</Location>\
```

   La siguiente lista organiza todo lo que acabamos de ver:

   - La configuración anterior le indica a Apache que utilice `Apache::AuthenDBI` como manejador del alias `/memberonly/`.
   - Las directiva `PerlSetVar` se utiliza para determinar pares clave=valor necesarios para este módulo.

La clave `Auth_DBI_data_source` determina el DSN de la base de datos que le indica al módulo la base de datos con la que tiene que conectar, y cuál es la Perl DBD que tiene que utilizar. En este caso el valor asignado hace que se conecte con una base de datos MySQL llamada `auth`. Debería asignar el driver a la RDBM que está utilizando. Por ejemplo, si está utilizando IBM DB2, su DSN debería ser `dbi:db2:database=auth`.

- Las claves `Auth_DBI_username` y `Auth_DBI_password` determinan el nombre de usuario y la contraseña de la base de datos que hay que utilizar para conectar con la base de datos nombrada (es decir, `auth`).

  El nombre de la tabla de contraseñas está determinado por la clave `Auth_DBI_pwd_table`; de igual modo, las claves `Auth_DBI_uid_field` y `Auth_DBI_pwd_field` especifican los campos del nombre de usuario y la contraseña.

  La clave `Auth_DBI_encrypted` tiene el valor `on`, de modo que se supone que las contraseñas almacenadas en la base de datos se encriptan utilizando el estilo tradicional de Unix en el que hay una función de encriptación por digestión llamada `crypt`.

- Para terminar, la directiva `require valid-user` le indica a Apache que sólo permita el acceso a aquellos usuarios que pasen la prueba de autentificación.

10. Puede reiniciar el servidor Apache utilizando el comando `/usr/local/apache/bin/apachectl restart` e intentar acceder al directorio `http://your_server_name/memberonly`, para ver si puede acceder al directorio con nombres de usuarios y contraseñas que no sean válidos. Si introduce un nombre de usuario y una contraseña válidos, debería ser autentificado.

Gestionar usuarios y grupos en una RDBM

Gestionar usuarios y grupos en una base de datos a mano es muy engorroso. Por suerte, no tiene que enfrentarse a este tipo de tareas. Puede utilizar una asignación de scripts de Perl para manejar estas tareas rutinarias de forma eficaz. A continuación vemos cómo hacerlo.

1. Necesitará instalar el paquete DBI y los módulos `DBD::database` apropiados del Comprehensive Perl Archive Network (CPAN). Por ejemplo, si instala la base de datos MySQL, como usuario raíz puede instalar los módulos DBI y DBD de la línea de comando del siguiente modo:

```
perl -MCPAN -e 'install DBI'
perl -MCPAN -e 'install DBD::mysql'
```

2. Compruebe si tiene los dos módulos CPAN llamados `HTTPD::UserAdmin` y `HTTPD::GroupAdmin` en su distribución instalada de Perl. Puede ejecutar los comandos locales `UserAdmin.pm` y `GroupAdmin.pm` para determinar si los tiene. Normalmente, estos dos módulos están instalados como parte de la distribución estándar. Por ejemplo, en mi sistema, estos módulos aparecen del siguiente modo:

```
/usr/lib/perl5/site_perl/5.6.0/HTTPD/UserAdmin.pm
/usr/lib/perl5/site_perl/5.6.0/HTTPD/GroupAdmin.pm
```

Si no tiene uno de los módulos o no tiene ninguno, instálelos del siguiente modo: como raíz baje los módulos `HTTPD::UserAdmin` y `HTTPD::GroupAdmin` CPAN de CPAN. En un sistema Linux puede simplemente ejecutar los comandos siguientes para instalarlos:

```
perl -MCPAN -e 'install HTTPD::UserAdmin'
perl -MCPAN -e 'install HTTPD::GroupAdmin'
```

TRUCO: Debería instalar el paquete `HTTPD::Tools` porque incluye los dos módulos así como otros módulos que son muy útiles para los servidores Web en general.

3. Copie el script `manage_users.pl` del CD-ROM en el directorio `/usr/bin` (ver el apéndice CD para obtener información sobre donde encontrar `/usr/bin`). Cambie el permiso para permitirle ejecutarlo. Asigne el permiso utilizando el comando `chmod 750 /usr/bin/manage_users.pl`.

4. Utilice su editor de texto favorito para modificar las siguientes líneas del script

```
my $DB_HOST             = 'localhost';
my $DB_PORT             = '';
my $DATABASE            = 'auth';
my $DB_DRIVER           = 'mysql';
my $DB_USER             = 'kabir';
my $DB_PASSWORD         = $dbpwd;
my $ENCRYPTION          = 'crypt';
my $USER_TABLE          = 'wwwusers';
my $USERNAME_FIELD      = 'username',
my $PASSWORD_FIELD      = 'passwd',
my $GROUP_TABLE         = 'wwwgroups';
my $GROUP_FIELD         = 'groupname';
my $MAXSZ_USER          = 40;
```

```
my $MAXSZ_PWD              = 20;
my $MAXSZ_GRP              = 40;
```

Necesita determinar las variables que están a continuación, para el código anterior:

- La variable $DB_HOST debe asignarse al servidor de la base de datos a la que quiere conectarse. Si el servidor de la base de datos se encuentra en la misma máquina que el servidor Web, se puede dejar el valor por defecto `'localhost'`.
- La variable $DB_PORT debe estar asignada al puerto del servidor. Por defecto, el puerto se selecciona automáticamente a no ser que tenga que utilizar un puerto no tradicional en el servidor de la base de datos para la conexión del cliente.
- La variable $DATABASE debe asociarse al nombre de la base de datos. El nombre por defecto de la base de datos es `'auth'` y sólo va a funcionar si ha seguido las instrucciones de secciones anteriores.
- La variable $DB_DRIVER debe asociarse al driver de la base de datos que necesita para conectar con el servidor de la base de datos. Para el servidor de la base de datos MySQL este driver se llama `mysql` y por lo tanto el valor por defecto sólo funcionará si está utilizando una base de datos MySQL.
- La variable $DB_USER debe asignarse al usuario al que se le ha concedido el acceso para crear, modificar o eliminar registros en las tablas especificadas en $DATABASE. Ver "Conceder acceso al servidor Apache a la base de datos de autentificación de usuarios en MySQL", para saber cómo puede conceder acceso a los usuarios a la base de datos MySQL.
- $DB_PASSWORD no se almacena en el script para aumentar la seguridad. Debe proporcionar la contraseña necesaria para acceder a la base de datos utilizando la opción de la línea de comando `-dbpwd=database_password` cada vez que ejecute el script `manage_users.pl`. Puede ilizar código de hardware, pero recomiendo eliminar la contraseña del código de hardware una vez que lo haya hecho con el script.
- La variable $ENCRYPTION puede asignarse en `none`, `crypt` (por defecto) o MD5. Cuando asignamos `none`, las contraseñas se almacenan en todo texto; cuando se utiliza `crypt`, las contraseñas se encriptan con el algoritmo de digestión de un único sentido utilizado en el entorno Unix tradicional; cuando se utiliza MD5, la contraseña se almacena como un valor de mensaje de digestión (MD5).

- La variable $USER_TABLE debe asignarse a la tabla de usuarios en su base de datos. Esta tabla debe tener el campo del nombre de usuario especificado por $USERNAME_FIELD y también el campo de la contraseña especificado por $PASSWORD_FIELD.
- La variable $GROUP_TABLE debería asignarse a la tabla de grupos en su base de datos. Esta tabla debe tener el campo del nombre de usuario especificado por $USERNAME_FIELD y además el campo de nombre de grupo especificado por $GROUP_FIELD.
- El tamaño máximo de $USERNAME_FIELD se determina utilizando el campo $MAXSZ_USER, que correspondería a lo que tiene que utilizar en el proceso de creación de $USER_TABLE. El tamaño de $PASSWORD_FIELD se controla de forma similar utilizando el campo $MAXSZ_PWD. Para terminar, el tamaño de $GROUP_FIELD se controla utilizando el campo $MAXSZ_GRP.

5. Guarde los cambios.

Añadir un nuevo usuario a la tabla de usuarios

Para añadir un usuario nuevo a la tabla de usuarios ejecute el siguiente comando:

```
manage_user.pl -db=user \
               -action=add \
               -user=user_name \
               -password=user_password \
               -dbpwd=database_password
```

Por ejemplo, para añadir un usuario llamado `kabir` con la contraseña de usuario `go#forward` puede ejecutar el comando siguiente:

```
manage_user.pl -db=user \
               -action=add \
               -user=kabir \
               -password=go#forward \
               -dbpwd=mydbpwd
```

Tenga en cuenta que aquí `mydbpwd` es la contraseña de la base de datos necesaria para escribir en la base de datos.

Eliminar un usuario de la tabla de usuarios

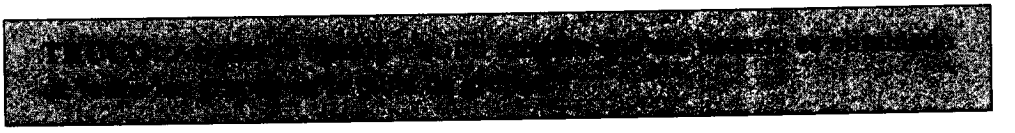
Para eliminar un usuario de la tabla de usuarios y de grupo ejecute el siguiente comando:

```
manage_user.pl -db=user \
               -action=del \
               -user=user_name \
```

```
                -dbpwd=database_password \
                 -auto=on
```

Por ejemplo, para eliminar un usuario llamado `kabir` de las tablas de usuario y de grupo utilizando una contraseña de acceso a la base de datos `mydbpwd`, puede ejecutar el comando siguiente:

```
manage_user.pl  -db=user \
                -action=del \
                -user=kabir \
                -dbpwd=mydbpwd \
                -auto=on
```



Actualizar una contraseña de usuario en la tabla de usuarios

Para actualizar una contraseña de usuarios en la tabla de usuarios, ejecute el siguiente comando:

```
manage_user.pl  -db=user \
                -action=update \
                -user=user_name \
                -dbpwd=database_password
```

Por ejemplo, para actualizar la contraseña del usuario `kabir` a `mksecret` utilizando la contraseña `mydbpwd` de la base de datos, ejecute el siguiente comando:

```
manage_user.pl  -db=user \
                -action=update \
                -user=kabir \
                -dbpwd=mydbpwd
```

Añadir un usuario a un grupo

Para añadir un usuario que ya existe a un grupo nuevo o a uno que ya existe, ejecute el siguiente comando:

```
manage_user.pl  -db=group \
                -action=add \
                -user=user_name \
                -group=group_name \
                -dbpwd=database_password
```

Por ejemplo, para añadir un usuario llamado `kabir` a un grupo llamado `administrators`, ejecute el siguiente comando:

```
manage_user.pl -db=group \
               -action=add \
               -user=kabir \
               -group=administrators \
               -dbpwd=mydbpwd
```

Aquí, `mydbpwd` es la contraseña de la base de datos necesaria para escribir en la tabla de grupos.

Eliminar un usuario de un grupo

Para eliminar un usuario de un grupo, ejecute el siguiente comando:

```
manage_user.pl -db=group \
               -action=del \
               -user=user_name \
               -group=group_name \
               -dbpwd=database_password
```

Por ejemplo, para eliminar un usuario llamado `kabir` de un grupo llamado `administrators`, ejecute:

```
manage_user.pl -db=group \
               -action=del \
               -user=kabir \
               -group=administrators \
               -dbpwd=mydbpwd
```

Aquí, `mydbpwd` es la contraseña de la base de datos necesaria para actualizar la tabla de grupos.

Utilizar cookies para autentificar sesiones

Tal y como se ha mencionado antes en este capítulo, la autentificación HTTP `Basic`, requiere que el navegador Web pase siempre el nombre de usuario y la contraseña codificada (no encriptada) cada vez que se solicite una página que se encuentra bajo una sección restringida. Esto hace los ataques por interposición de intrusos muy sencillos. Este ataque implica un intruso interceptando paquetes entre un servidor Web y un navegador Web utilizando autentificación HTTP `Basic` para determinar las contraseñas.

La solución contra el ataque por interposición de intrusos, es utilizar una conexión secure socket layer (SSL) y un esquema de autentificación basada en sesión, en el que se autentifica al usuario una vez utilizando la autentificación HTTP `Basic`, y todas las solicitudes siguientes del recurso restringido son autorizadas utilizando una sesión segura (encriptada) de cookies, en lugar de la contraseña codificada.

Esta sección discute una solución que utiliza bases de datos MySQL (sin embargo, puede utilizar cualquier otra RDBM) y módulos `mod_perl` de CPAN. A continuación tenemos el modo de implementar este tipo de solución.

1. Si no está instalado ya, instale el módulo `mod_perl` tal y como se discute en el capítulo 16.
2. Instale dos módulos CPAN para Apache utilizando el siguiente comando como ruta:

```
perl -MCPAN -e 'install Apache::AuthCookie'
perl -MCPAN -e 'install Apache::AuthTicket'
```

3. Una vez que tiene instalado estos módulos, necesitará crear la base de datos `auth` y la tabla `wwwuser` discutida en la sección "Crear la base de datos de autentificación de usuarios en el servidor mysql" de este capítulo. Necesitará también añadir las dos tablas siguientes.

```
CREATE TABLE tickets (
  ticket_hash CHAR(32) NOT NULL,
  ts          INT NOT NULL,
  PRIMARY KEY (ticket_hash)
   );

   CREATE TABLE ticketsecrets (
  sec_version   BIGINT,
  sec_data      TEXT NOT NULL
);
```

   Siga las instrucciones dadas en la sección "Crear la base de datos de autentificación de usuarios en el servidor mysql" de este capítulo, para añadir estas tablas en la base de datos `auth`.

4. Una vez que ha creado esas tablas, debe añadir un secreto en la tabla `ticketsecrets`. El modo más sencillo de añadir un secreto es registrarse en la base de datos y conectarse a la base de datos `auth` y emitir una sentencia insertada del siguiente modo:

```
insert into ticketsecrets (sec_version, sec_data) values
('number', 'random_data');
```

5. Determine a qué localización del directorio Web quiere restringirle el acceso y emita una sesión de cookies para esto. En este ejemplo, he llamado a esta localización `/protected`. Añada la siguiente configuración a su archivo `httpd.conf`:

```
PerlModule  Apache::AuthTicket
PerlSetVar  ProtectedTicketDB
DBI:mysql:database=auth;host=localhost
PerlSetVar  ProtectedTicketDBUser  httpd
```

```
PerlSetVar ProtectedTicketDBPassword secret1
PerlSetVar  ProtectedTicketTable  tickets:ticket_hash:ts
PerlSetVar  ProtectedTicketUserTable  wwwusers:username:passwd
PerlSetVar  ProtectedTicketSecretTable
ticketsecrets:sec_data:sec_version
PerlSetVar  ProtectedTicketPasswordStyle  crypt
```

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

- Aquí la directiva `PerlModule` le dice a Apache que quiere utilizar el módulo `Apache::AuthTicket`.
- Las directivas `PerlSetVar` se utilizan para asignar los distintos pares clave=valor que el módulo necesita.
- Las claves `ProtectedTicketDB` asignan el DSN para la base de datos.
- El valor de muestra `DBI:mysql:database=auth;host=localhost` le dice al módulo `AuthTicket` que queremos utilizar el driver de la base de datos MySQL (`mysql`) y conectar con la base de datos llamada `auth`, que reside en el host local (en la misma máquina que el servidor Web). Asegúrese de que cambia el nombre de host de forma apropiada, si está ejecutando el servidor de la base de datos MySQL en la misma máquina que el servidor Web Apache.
- Las directivas `ProtectedTicketDBUser` y `ProtectedTicketDBPassword` le dicen al módulo `AuthTicket` qué bases de datos de nombres de usuarios y de contraseñas se necesitan para acceder al servidor de bases de datos.
- Las claves `ProtectedTicketTable`, `ProtectedTicketUserTable` y `ProtectedTicketSecretTable` le dicen al módulo qué tablas de ticket y de usuarios utilizar en la base de datos y qué campos son necesarios.
- `ProtectedTicketPasswordStyle` determina el tipo de encriptado. Tiene tres opciones: estilo tradicional de Unix en el que hay una encriptación por digestión en una dirección, Plaintext (no recomendado), o MD5.

6. A continuación añada las siguientes líneas de configuración:

```
PerlSetVar  ProtectedTicketExpires  30
PerlSetVar  ProtectedTicketLogoutURI  /protected/index.html
PerlSetVar  ProtectedTicketLoginHandler  /protectedlogin
PerlSetVar  ProtectedTicketIdleTimeout  15
PerlSetVar  ProtectedPath  /
PerlSetVar  ProtectedDomain  .domain_name
```

```
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`.