La biblia del Servidor Apache

Mohammed J. Kabir

Indice

Prefacio

Bienvenido a la Biblia del Servidor Apache. Es muy probable que ya haya oído hablar de los Servidores Apache. De hecho, más del 50% de los administradores Web de Internet utilizan sistemas Apache. Sin embargo, a pesar de la gran cantidad de sistemas existentes, son muy pocos los libros dedicados a ellos. Sí, he estado en varias librerías y sé de lo que estoy hablando. Creo que no se ha escrito ninguno pensando en los administradores Web. De hecho tuve uno en mis manos que hablaba de las propiedades internas de Apache (del código C) en vez de explicar cómo utilizar este sistema; otro dedicaba gran parte de su extensión a HTML. Este libro está pensado para todos aquellos que quieren aprender a administrar un sistema Apache. Así que si éste es su caso, ya puede decir que ha encontrado su varita mágica. ¡Utilícela!

Este libro, ¿es para usted?

Esta obra ayudará a todos aquellos que tengan interés en administrar un servidor Web Apache. Como este libro no da por hecho que el lector tiene experiencia en administración, es igualmente válido para cualquiera. A diferencia de otras obras especializadas en los servidores Web, ésta tan solo se centra en un tema: el Servidor Web Apache.

El texto trata de los aspectos de la administración del servidor Web Apache. Toca temas como la compilación del código fuente, los archivos binarios de la instalación y la configuración en los primeros capítulos, para que se pueda trabajar con el servidor desde el principio para que resulte más sencillo comprender lo que se va haciendo en cada paso.

También se ven aspectos más avanzados como la configuración de SSI, CGI y FastCGI, así como su impacto en el servidor y el modelo de seguridad. Conocerá distintos peligros y las técnicas que se utilizan para minimizarlos. Entre los temas más avanzados se incluyen la mejora de la eficacia del servidor Web, autenticación del usuario, visualización, técnicas para volver a escribir URL, incorporación del intérprete de Perl con Apache y uso de SSL (Secured Socket Layer).

Organización del libro

He divido el libro en seis partes:

Parte I: Inicio

Después de una breve introducción al número uno de los servidores Web, le mostraré el proceso de obtención y compilación de Apache. Aprenderá a iniciar el servidor Apache y a aplicar unos cambios mínimos sobre la configuración básica (la idea es que lo pueda ejecutar en su sistema lo antes posible). La parte termina con una serie de referencias a las directrices del núcleo de Apache y los módulos estándar. De esta forma estará preparado desde el principio para hacerse cargo de las tareas más serias de la administración de Apache.

Parte II: Administrar su sitio Web

Esta sección le llegará a convertir en un administrador profesional. Verá temas relacionados con la administración Web, como los sitios Web virtuales, SSI (Server Side Include) y CGI (Common Gateway Interface). Aprenderá todo lo que hay que saber sobre la administración de sitios Web, creación de mensajes de correo electrónico virtuales y hacer que sus sitios tengan una apariencia profesional. Gracias a una serie de ejemplos conocerá SSI; tenga presente que éstos no se limitan a mostrarle la parte práctica de lo leído en estas páginas, sino que también le enseñarán a divertirse (por ejemplo) visualizando el contenido HTML resultante de las decisiones que se van tomando

sobre la marcha, o a utilizar variables personalizadas para que los programas CGI/SSI sepan qué es lo que tienen que hacer. En los capítulos dedicados a CGI verá cómo preparar el servidor Apache para crear una plataforma CGI, cómo prepararlo para que trabaje con los programas escritos en dicho idioma y cómo incrementar su eficacia utilizando la tecnología FastCGI.

Parte III: La seguridad

Cualquier ordenador conectado a Internet puede ser víctima de un abuso o de un uso poco ético. Siempre es conveniente tener presente una serie de medidas de seguridad. En esta parte del libro aprenderá a utilizar los estándares de autenticación basados en el protocolo HTTP. Le enseñaré a crear áreas de acceso restringido basadas en el uso de nombres/contraseñas, direcciones del servidor o IP. La verdad es que no me quedo en las bases, sino que voy más allá. Veremos programas externos especializados en la autenticación y cómo utilizar grandes bases de datos para controlar el acceso de los usuarios. El siguiente paso será descubrir quién accede a su sitio Web. Por eso aprenderá a visualizar su servidor Web a través de la Red, analizar los ficheros de registro y crear los suyos propios para hacer frente a las necesidades de seguridad y marketing. También le mostraré los peligros que tiene la ejecución de programas SSI y CGI así como unas cuantas medidas de seguridad que puede tomar para evitarlos.

Parte IV: Implementar propiedades avanzadas

En esta parte del libro veremos las propiedades avanzadas propias de Apache. Aquí, aprenderá cómo aprovechar el lenguaje de programación CGI más utilizado de todos, Perl; cómo interactúa con el servidor proxy; cómo asegurar las transacciones en las que interviene Apache; y cómo volver a escribir las URL. Una vez que se haga con estas técnicas, podrá considerarse un administrador profesional.

Parte V: Uso de Apache hoy y mañana

Aquí les comentaré cómo se puede utilizar Apache para cubrir las necesidades de hoy y las de mañana. Primero veremos los aspectos generales que debería conocer. Con ellos y utilizando las soluciones recomendadas se asegurará de que su servidor Apache está listo para hacerse cargo del trabajo de hoy y del de mañana. A continuación les presentaré conceptos como el ciclo Web, que le permitirá desarrollar sitios Web que sean capaces de hacerse

cargo del trabajo de mañana. Verá la importancia de los estándares en lo que al desarrollo de sitios Web se refiere. Conviene desarrollar políticas para contornar la calidad de sus trabajos. Como en algún momento habrá que expandir el sitio Web, le mostraré cómo Apache y otras tecnologías relacionadas con Internet pueden crear redes con varios servidores. Para terminar le echaremos un vistazo a la última versión de Apache para Windows. Con esta versión beta podrá convertir su plataforma Windows en un servidor Apache. Y después de todo esto sólo tendrá una idea en la cabeza, ¡Apache ha venido a quedarse!

Parte VI: Apéndices

Por último hay cuatro apéndices para concluir el libro en los que se tratarán de los códigos de estado de HTTP/1.1, expresiones regulares, fuentes de Internet para Apache y el contenido del CD-ROM.

Convenios utilizados en este libro

Para leer esta obra no hay que aprenderse ningún convenio nuevo. Recuerde que cuando tenga que escribir algo en la línea de comandos, tendrá que terminar pulsando Intro. También conviene que se fije en los siguientes iconos:

Nota: se utiliza siempre que hace falta dar una explicación adicional.

Truco: este icono se utiliza cuando se vaya a decir algo que ahorrará tiempo y esfuerzo.

Advertencia: aparecerá para advertirle de algún peligro potencial.

Parte I
Inicio

1 Apache, el número uno de los servidores Web

Bienvenido a Apache, el número uno de los servidores Web de todo el mundo. Más del 50% de los servidores de la Red utilizan Apache, de acuerdo con una encuesta desarrollada por Netcraft (www.netcraft.co.uk/Survey/). Si está barajando la idea de montar un servidor Apache, sepa que está en el lugar indicado. Este capítulo le mostrará cómo hacerlo.

En los primeros días de la Red, NCSA (National Center for Super Computing Applications) creó un servidor Web que se convirtió en el número uno de los servidores Web a principios de 1995. De todas formas, por esa misma época el proyecto de NCSA cayó en el olvido. Los usuarios de dicho servidor comenzaron a intercambiar los parches que tenían y en breve se dieron cuenta de que había que crear un foro que se encargase de la administración de dichos parches. Había nacido Apache Group. Utilizaron el código del servidor Web de NCSA y crearon un nuevo servidor al que llamaron Apache. Procedente del código fuente de NCSA y de una gran colección de parches, el servidor Apache se ha convertido en la comidilla de la comunidad de Internet. Apenas le ha llevado tres años convertirse en el servidor por excelencia del mercado.

La primera versión (0.6.2) apareció en Abril de 1995. La versión 1.0 el 1 de Diciembre de ese mismo año. Desde el principio, Apache Group ha crecido como una empresa sin ánimo de lucro. Trabaja a través de Internet. De

todas formas el desarrollo del servidor Apache no está limitado, ni mucho menos, al trabajo de este grupo. Cualquiera que sepa cómo participar en el desarrollo del servidor o de sus módulos, puede hacerlo, aunque es verdad que el grupo tiene la última palabra a la hora de decidir qué se incluye en la distribución estándar conocida como Servidor Web Apache. De esta forma se ha conseguido que sean miles las personas que trabajan en el desarrollo de nuevas propiedades, corrección de errores, puertos para las nuevas plataformas, etc. Cuando se envía el nuevo código a Apache Group, sus miembros investigan todos los detalles, lo someten a todo tipo de test y comprueban su calidad. Si están satisfechos con los resultados, incluyen el código en la versión que se distribuye.

Ahora que ya sabemos algo más sobre la historia de Apache, vamos a conocer sus propiedades.

Propiedades de Apache

Una de las características más importantes de Apache es que se utiliza en casi todas las plataformas. Al principio únicamente funcionaba con los servidores Web Unix, pero la situación cambió. En la actualidad Apache no sólo funciona con todas las versiones de Unix sino que también lo hace con Windows 95/NT, Amiga y OS/2.

Apache tiene otras muchas características, como la indexación de directorios, uso de sobrenombres con las carpetas, negociación de contenidos, informes configurables sobre los errores HTTP, Ejecuciones SetUID o Programas CGI, administración de recursos para los procesos emparentados, mapas de imagen para los servidores, reescritura de URL, corrección de URL y manuales online.

Entre sus principales propiedades tenemos las siguientes:

- Admite la última versión del protocolo HTTP/1.1 (Apache es uno de los primeros servidores Web que trabajó con este protocolo). Es completamente compatible con el nuevo estándar HTTP/1.1 y con su versión anterior HTTP/1.0. Está preparado para hacerse cargo de todas las nuevas propiedades que tenga que ofrecer este protocolo. Por ejemplo, antes de HTTP/1.1 un explorador Web tenía que esperar la respuesta del servidor Web antes de poder enviar otra petición. Con esta nueva versión esto ya no hace falta. Un explorador Web puede enviar peticiones en paralelo, con lo que se ahorra ancho de banda al no tener que incluir una cabecera en todas y cada una de las solicitudes. Con este sistema las

respuestas a las peticiones del usuario aparecerán antes en la pantalla de su explorador.

- Simplicidad, aún con las potentes configuraciones basadas en ficheros. El servidor Apache no dispone de una interfaz gráfica. Tiene tres archivos de configuración (en texto plano) que se pueden utilizar para ajustar los parámetros del servidor. Todo lo que necesitará será un editor de textos. Si lo desea, puede simplificar aún más la configuración uniendo los tres archivos en uno sólo.
- Puede trabajar con CGI. El servidor Apache utiliza el módulo mod_cgi. Está compilado con la versión 1.1 de CGI y entre las características que ofrece se encuentra la personalización de las variables de entorno o la búsqueda de errores, propiedad que tanto cuesta a otros servidores.
- Admite servidores virtuales. Apache es también uno de los primeros servidores Web que puede trabajar con direcciones IP y con nombres virtuales.
- Admite la autenticación HTTP. Apache también puede trabajar con la autenticación básica de la Red. Además está preparado para asimilar los mensajes basados en la autenticación, algo que cada vez es más popular y que los servidores van a tener que implementar. Para sus autenticaciones, Apache utiliza archivos de contraseñas, DBM, SQL o llamadas a programas externos especializados en la materia.
- Integra Perl. Perl se ha convertido en el lenguaje de programación por excelencia de la programación de scripts en CGI. Y posiblemente uno de los factores que ha influido para que ocurriese esto ha sido Apache. Por medio del módulo mod_Perl podrá cargar scripts CGI programados en Perl en la memoria del ordenador y utilizarlos tantas veces como desee. Este proceso elimina los problemas de inicio tan asociados con los lenguajes de interpretación (entre los que se encuentra Perl).
- Cuenta con un servidor Proxy. Puede convertir su servidor Apache en uno Proxy. Pero ha de tener en cuenta que el módulo que utiliza para ello no admite la función inversa ni el protocolo HTTP/1.1. Ya se está trabajando en la actualización de este módulo.
- Registros personalizables e información sobre el estado del servidor. Apache es muy flexible a la hora de registrar y visualizar el estado del servidor. De hecho se puede acceder a esta información a través de un explorador Web. También se pueden personalizar los archivos de registro relacionados con los enlaces.

- Puede trabajar con SSI. Apache dispone de una serie de anexos para el servidor que mejoran la flexibilidad del desarrollado de sitios Web. Utilizando los comandos eXtended Server Side Include en su servidor Apache, podrá hacer lo mismo que con SSI y más.
- Admite SSL. Debido a las leyes y restricciones del Copyright sobre la exportación e importación de productos de los EE.UU., Apache no es compatible con SSL. Pero hay una gran cantidad de parches para Apache (Apache-SSL), además de versiones comerciales del producto, que permite que pueda trabajar con este sistema de seguridad.
- Capacidad para registrar las acciones de los usuarios. Por medio de las cookies HTTP un módulo de Apache, llamado mod_usertrack puede registrar los movimientos de los usuarios según se mueven por el sitio Web Apache.
- Trabaja con FastCGI. Es verdad que no todo el mundo escribe CGI en Perl. Entonces, ¿cómo pueden hacer que sus aplicaciones sean más rápidas? Con una solución de Apache: el módulo mod_fcgi, gracias al cual se puede implementar un entorno FastCGI dentro de Apache y hacer que sus aplicaciones vayan mucho más rápido.
- Admite Java Servlets. Java Servlets es un módulo experimental en estado alpha (mod_jserv) para Apache. Cuando salga oficialmente, logrará que Apache ejecute las aplicaciones en Java propias del servidor. De esta forma podrá trabajar con multitarea.

Arquitectura 101 de Apache

El servidor oficial de Apache se compone de un núcleo en el que se encuentra el código de Apache y una serie de módulos estándar que se compilan en el mismo servidor incluso con la configuración predeterminada. Si decide eliminar uno de estos módulos, todo lo que debe hacer es editar el archivo de configuración y modificar su contenido. Este proceso es tan sencillo que lo único que tendrá que hacer será borrar una línea de texto. Cuando vuelva a recompilar el servidor Apache, el módulo habrá dejado de ser ejecutable.

Por ejemplo, si no desea que su servidor trabaje con los script CGI, puede eliminar la siguiente línea de su archivo de configuración que encontrará en la carpeta Apache:

```
Addmodule modules/standard/mod_cgi.o
```

Apache también puede trabajar con las Interfaces de Programación de Aplicaciones (API) que permiten que los desarrolladores construyan sus propios módulos para añadir más propiedades a los servidores Apache. En la figura 1.1 puede ver un ejemplo de un supuesto en el que tres servidores Apache tienen tres niveles de funcionalidad diferentes.

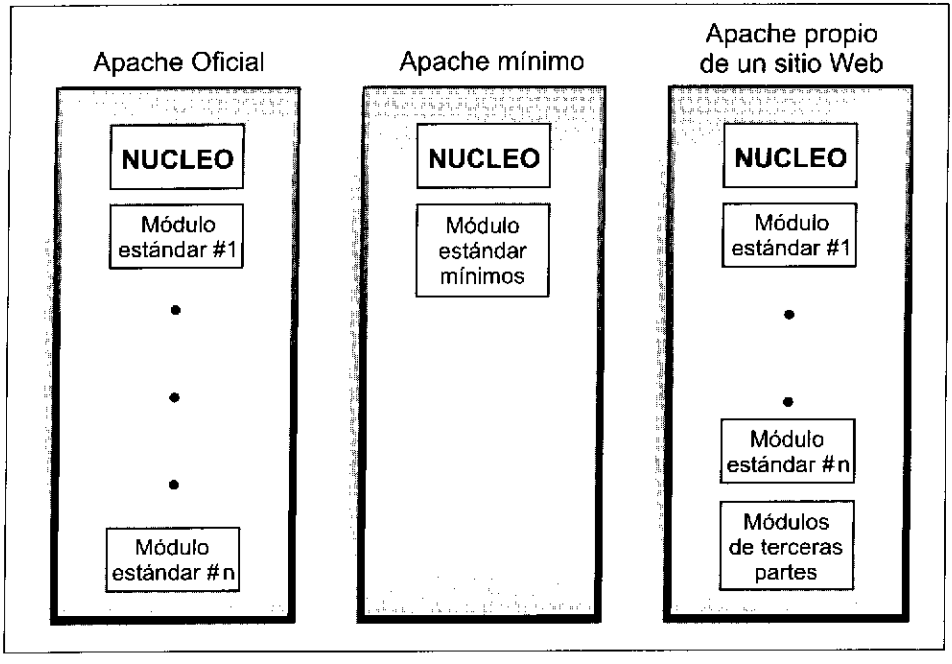


Figura 1.1. Arquitectura modular de Apache

De esta forma podrá construirse un servidor Apache que se ciña perfectamente a sus necesidades. La configuración mínima, que es la que aparece en la figura 1.1, es un ejemplo. Si necesita la presencia de un módulo que no se incluya en la distribución estándar, podrá tomar uno desarrollado por terceras empresas o simplemente programárselo usted mismo. En la figura 1.1 a este caso le hemos llamado Apache propio de un sitio Web.

La ventaja de poder añadir y quitar módulos es fantástica. Se puede llegar a compilar un servidor Apache especialmente diseñado para las exigencias de nuestro sitio Web. Los desarrolladores Web disfrutan de una flexibilidad a la hora de programar. Pueden utilizar la interfaz API de Apache sin preocuparse del código del núcleo del servidor. Además, añadir nuevos módulos no quiere decir que se esté parcheando el núcleo de Apache, con lo que se facilita considerablemente la vida del desarrollador.

El futuro de Apache

El futuro de Apache aparece de lo más brillante por varias razones. La libertad de movimientos del software a través de Internet es una de las más fuertes. Incluso empresas como Netscape Communications se ha dado cuenta de ello. Hasta poco antes de la aparición del comercio a través de Internet, en la Red únicamente se utilizaba software gratuito. Es un buen momento para afianzar mi fe en este tipo de software. Pregúntese lo siguiente: ¿cuántas veces ha oído hablar de Linux, Perl, FreeBSD o Apache? Cada vez se oye hablar más de este tipo de software porque es bueno y la verdad es que funciona bastante bien.

De todas formas, éste no es el fin del mundo del software comercial. En la Red hay sitio para todos. De hecho, el software gratuito impulsa la comercialización de la tecnología en la que se basan. Si está convencido (por lo menos en espíritu) de que Apache ha venido para quedarse, es posible que le interesen los derroteros hacia los que se encamina el futuro de esta tecnología. En las próximas secciones veremos dos técnicas relacionadas con Apache.

La interfaz GUI perdida

Siempre que se compara Apache con cualquier servidor Web "bien" se resalta su ausencia de una interfaz GUI, o una interfaz gráfica para el usuario. Bien, me complace anunciarles que ¡los días de ausencia de GUI están ya contados!

Los desarrolladores de Apache están trabajando en un proyecto que se llama Apache GUI Configuration Project cuyo objetivo es el desarrollo de una configuración para el servidor final que pueda interactuar con los clientes cuya configuración se base en Java- y tcl/tk. Están muy cerca de lograrlo y en breve tendremos una versión pública. Puede visitar la página principal del proyecto en:

```
www.esi.us.es/~ridruejo/gui/home.html
```

Un cliente con una configuración basada en Java que está prácticamente listo para bajarse de Internet lo encontrará en:

```
http://butler.disa.mil/ApacheConfigClient/
```

En la figura 1.2 que se muestra a continuación tiene la pantalla principal de un servidor Apache.

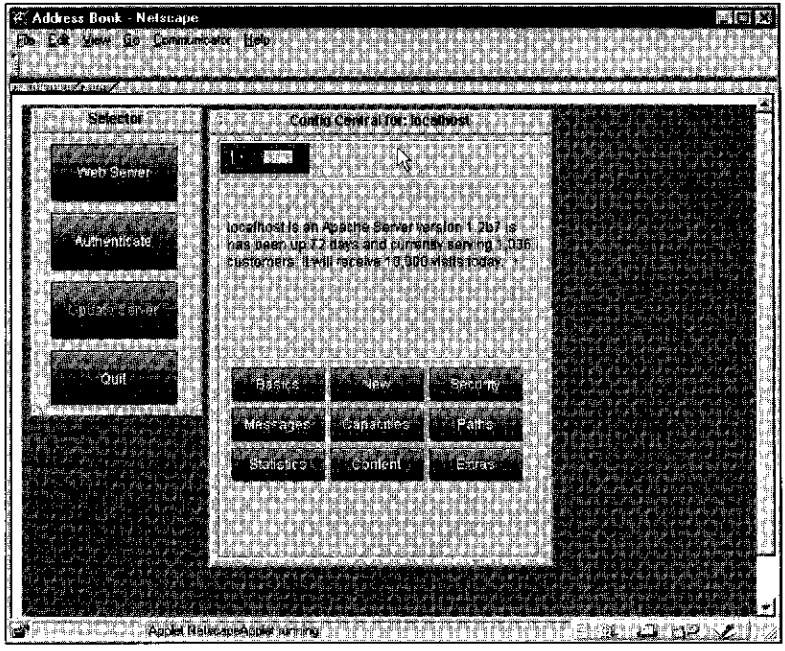


Figura 1.2. Ventana de configuración principal del servidor local

La figura 1.3 muestra la ventana de configuración del servidor para el ordenador local.

Conviene que se pase por este sitio Web y revise las actualizaciones del proyecto. Pero si le es completamente necesario disponer de una interfaz GUI para su servidor Apache, sepa que puede comprar una. Encontrará un proyecto llamado Warpaint en:

```
www.rovis.com/warpaint/
```

Otra de las propiedades que está a punto de salir (si no lo ha hecho ya) es la corrección del problema del año 2.000.

Prepararse para el próximo milenio

Según se va acercando la llegada del año 2.000 cada vez son más los administradores IT que se empiezan a preguntar qué ocurrirá con el problema asociado al cambio de fecha. Como un servidor Web suele ser la cara que muestra una empresa a los clientes que viajan por Internet, conviene que antes de desarrollar un nuevo servidor Web se pregunte si Apache tiene solucionado el problema del año 2.000.

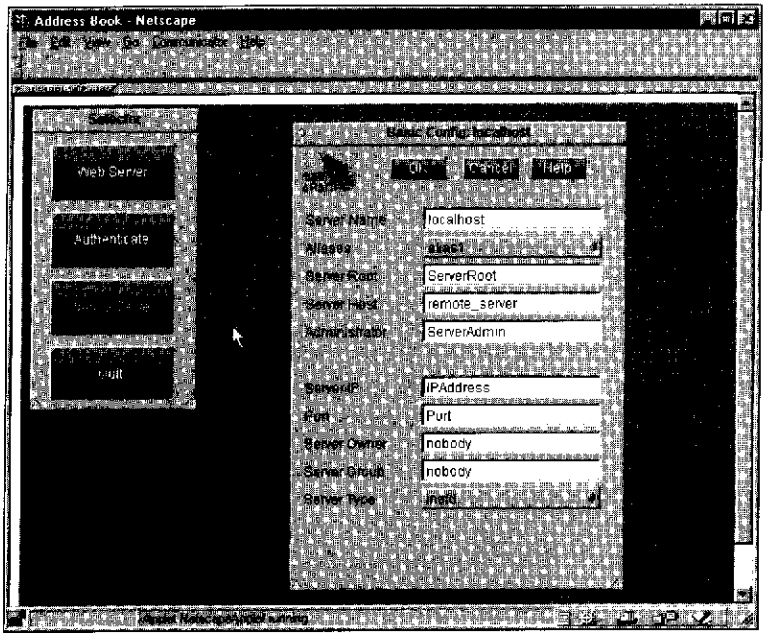


Figura 1.3. Ventana de configuración básica de un servidor

La respuesta no es nada sencilla porque hay demasiados factores involucrados. El código de Apache no guarda los años en dos dígitos, por lo que en principio sí que tiene resuelto el problema del cambio de milenio. Pero los problemas pueden aparecer a través de fuentes externas, como el sistema operativo o incluso los propios protocolos. Para suplir estos problemas externos Apache hace todo lo que puede para protegerse del famoso problema. Por ejemplo, si Apache se encuentra con un año anterior a 1970 (el año mágico que se utiliza como inicio de fechas en la mayoría de los ordenadores) asume que el siglo será el 2.000 en vez del 1.900.

Desde el punto de vista técnico, Apache es perfectamente compatible con el cambio de milenio, pero el software externo no. Si le preocupa el problema del cambio de milenio, investigue todo el software y los componentes de hardware de su entorno informático.

Licencia de Apache

Recientemente, el software gratuito (como Apache, Perl y Linux) está apareciendo con cierta frecuencia en la prensa. Todo se debe a que Netscape ha decidido regalar su famoso explorador Web, Netscape Communicator.

Desafortunadamente, el software gratuito como Apache, Perl o Linux no comparten los mismos acuerdos de licencia. Y la prensa, al meter todo el software gratuito dentro del mismo grupo, ha liado un poco las cosas.

Todo el software gratuito tiene que ser completamente gratuito. Sin embargo, hay una serie de restricciones legales. Por ejemplo, la licencia de Perl no obliga a que una empresa que modifique dicho lenguaje haga públicas sus modificaciones, mientras que Linux sí que lo hace. Apache no obliga que se publiquen los cambios.

Por eso conviene que piense en Apache como un software gratuito, protegido por las leyes del Copyright y creado por Apache Group. No es shareware ni está bajo los acuerdos de GLP (GNU Public License).

Ahora que ya tiene algo más claros los términos de la licencia de Apache se puede decir que está listo para utilizarlo. De todas formas es posible que prefiera saber quién utiliza Apache como servidor Web.

¿Quién utiliza Apache?

Como ya hemos comentado, el 50% de los servidores de la Red son Apache. Sin embargo, si le gustan los números, puede visitar el siguiente URL:

```
www.apache.org/info/apache_users.html
```

Encontrará una lista con el nombre de los usuarios de Apache. ¿Cómo es posible que todos esos administradores vivan sin servicios directos de consulta con Apache Group? Aunque no haya ningún número de teléfono al que se pueda llamar para preguntar dudas, la verdad es que Internet está lleno de sitios Web donde se tratan los problemas de estos servidores, además de grupos de news en los que se publican dudas todos los días. De todas formas si le gustaría tener un servicio de atención al cliente relacionado con un producto gratuito, conviene que llegue a un acuerdo con alguna de las muchas empresas que se dedican a prestar este tipo de servicios con los servidores Apache.

¿Está listo?

Espero que esté listo para empezar. Este libro le ayudará a convertirse en un eficiente administrador de servidores Apache. En breve se dará cuenta de la potencia de Apache y comprenderá por qué en tan sólo tres años se ha convertido en todo un número uno.

2 Cómo obtener e instalar Apache

Además de todas las razones que he expuesto con anterioridad, he de confesar que por lo que más me gusta Apache es porque su código fuente es gratuito. Así puedo explorarlo, ver cómo está hecho y si lo deseo, modificarlo a mi gusto. Es posible que para aquellos que no programen con C pero que necesiten un servidor Web potente y gratuito, la idea de enfrentarse a un código C en ANSI no sea el pasatiempo ideal. Afortunadamente, no se tiene que preocupar de nada. Apache se distribuye en código fuente y en paquetes binarios. En este capítulo veremos cómo instalar Apache a partir del código fuente y de los archivos binarios.

Fuente oficial de Apache

Siempre que desee hacerse con software gratuito de Apache (código fuente o archivos binarios), asegúrese que no se lo baja de un sitio desconocido. ¿Qué es un sitio desconocido? Lo vemos con un ejemplo. Supongamos que desea hacerse con el software de un explorador Web basado en Java, gratuito y que lo ha desarrollado Sun Microsystems. Obviamente no será muy fiable bajárselo de un sitio FTP llamado vaya.usted.a.saber.que_hay.aquí.com.sg. Lo mejor es pasarse por el sitio FTP Java.sun.com. No olvide que tendrá que

tomar algunas precauciones. A fin de cuentas, no es el único que accede a los sitios que visita. Por ejemplo, si se obtiene una utilidad para Windows de ftp.cdrom.com (un sitio FTP fiable), no puede estar completamente seguro de que no ha entrado nadie y ha introducido algo dañino en cualquiera de los programas. La otra opción es que si se conoce algún problema con el software, lo más probable es que no lo encuentre en este sitio FTP. De todas formas, si tiene alguna duda sobre la integridad de algún programa, pregunte en USENET. Publique la pregunta en el grupo de news adecuado y especifique luego la ubicación en la que se encuentra el programa sobre el que desea la información.

Por suerte para todos nosotros, los desarrolladores y los distribuidores de Apache construyeron un sitio oficial bastante seguro. El sitio Web de Apache es el siguiente:

```
http://www.apache.org
```

Aquí encontrará las últimas versiones de Apache (estable y oficial), parches, módulos, etc. Siempre que necesite algo relacionado con Apache, visite esta dirección. Además, desde aquí se le redirigirá automáticamente al sitio mirror (sitio que contiene lo mismo que el servidor principal) más cercano a su ubicación geográfica, para que se encuentre con menos congestión en la red y tarde menos en bajarse el software deseado.

Requisitos del sistema

Antes de que le aparezca en su pantalla el logotipo de Apache (véase figura 2.1), es posible que desee saber si su servidor es lo suficientemente potente como para ejecutarlo.

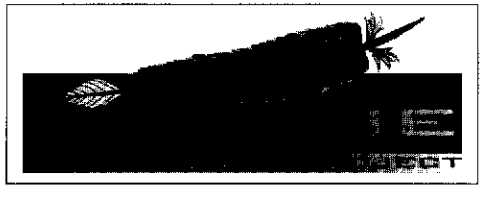


Figura 2.1. Logotipo de Apache

Apache no necesita demasiados recursos. Funciona con un sistema Linux con 6-10 MB de espacio en disco duro y unas 8MB de memoria RAM. De todas formas, lo que tenemos en mente no es precisamente "ser capaces de

ejecutar" Apache, sino servir páginas Web, abrir procesos CGI y aprovechar todo lo que nos puede llegar a ofrecer la Red. Es posible que necesite algo más de memoria RAM y de espacio de disco. Para averiguar la configuración de su ordenador puede hacer dos cosas. Hable con alguien que trabaje con un servidor Apache similar al que va a instalar o instálelo y trate de hacerse una idea de lo que realmente necesita.

En este último caso, puede usar utilidades del sistema tales como ps, top, etc. para ver la cantidad de memoria que utilizan los procesos de Apache. Puede determinar la cantidad total de memoria multiplicando uno de ellos por la cantidad total de procesos que se utilizarán en una hora punta. De esta forma se puede hacer con una idea bastante aproximada de la cantidad de memoria RAM que puede llegar a necesitar. Tenga en cuenta que si tiene previsto que su servidor ejecute varios programas CGI, tendrá que añadir a la cantidad de memoria que acaba de calcular, la que lleguen a consumir.

Los requisitos de disco de Apache no suelen ser ningún problema, ya que los archivos binarios no llegan a ocupar más de 1MB y el código fuente, unos 5MB. De todas formas sí que conviene que preste atención a los archivos de registro que genera Apache, dado que cada una de sus entradas llega a ocupar 80 bytes. Y si, en un día puede llegar a generar 10.000 entradas, sepa que el archivo ocupará unos 8MB.

Bajarse el software

Antes de que se baje el software de Apache por primera vez conviene que tome nota de unos cuantos detalles. Es bueno que busque dos versiones de Apache: una de ellas es la versión oficial y la otra es una beta en la que se incluyen los últimos cambios efectuados en el código fuente y nuevas propiedades. Por ejemplo, si encuentra una versión 1.2.4 y otra llamada 1.3b3, querrá decir que la primera es la oficial y la segunda la versión beta. Esta última es la tercera beta que sale al mercado y sus anteriores son 1.3b1 y 1.3b2. Es posible que sea más estable que las anteriores, pero yo no le recomendaría que la utilizase para crear su servidor Web.

Para hacerse con la versión que desee, o bien con algún enlace que le lleve a otra dirección, visite:

```
www.apache.org/dist/
```

En este sitio Web encontrará unas cuantas versiones beta y varios paquetes comprimidos con el software de Apache, como por ejemplo:

```
apache_1.2.4.tar.Z
apache_1.2.4.tar.gz
apache_1.3b3.tar.gz
apache_1.3b3_win32.exe
```

En esta lista de archivos aparecen varios tipos de formatos de compresión que se usan para distribuir el código fuente. Todo lo que necesitará es un sistema de compresión que pueda trabajar con ellos (en otras palabras, un programa que tenga el código de descompresión adecuado). Generalmente dichos programas son tar, gnunzip o gzip. Por ejemplo, para descomprimir el archivo Apache 1.2.4.tar.gz en Linux (RedHat 4.2) usaría el siguiente comando:

```
tar xvzf apache_1.2.4.tar.gz
```

O incluso:

```
gzip -d apache_1.2.4.tar.gz
tar xvf apache_1.2.4.tar
```

Todos ellos descomprimirían el archivo, respetando los subdirectorios originales y dejando el path de los ficheros intacto.

Nota: es posible que se encuentre con algunos archivos de descompresión automática para Windows 95/NT. Cualquiera de ellos se podrá ejecutar en el mismo momento en que se lo baje de Internet.

Los archivos binarios se suelen guardar en un directorio diferente y se crea una carpeta para cada sistema operativo. Recuerde que si no se crea la carpeta correspondiente con su sistema operativo no quiere decir que Apache no lo admita. Simplemente que nadie ha compilado el código fuente de Apache para que trabaje con él. Encontrará ejecutables para los siguientes sistemas:

- AIX.
- FreeBSD.
- HPUX.
- IRIX.
- Linux NetBSD.
- OS2, Solaris.
- SunOS.
- Ultrix.

Si decide bajarse una versión de Apache que ya haya pasado por el compilador, se puede saltar la siguiente sección.

Configuración del código fuente para su sistema

En esta sección veremos cómo compilar Apache para que trabaje con un sistema Linux. Los pasos deberían ser los mismos para todos los sistemas Unix.

Una vez que extrae el código encontrará un directorio llamado src. Ahí se encuentra el código fuente. Siga estos pasos:

1. Lea el contenido del archivo README que se encuentra en el directorio principal (es decir, el directorio en el cual se encuentra la carpeta src).
2. Cambie al directorio en el que se encuentra en código fuente de Apache.
3. Lea el archivo de texto llamado INSTALL. Observe las especificaciones relacionadas con su sistema operativo.
4. Copie el archivo modelo de configuración (Configuration.tmpl) en un archivo cuyo nombre sea Configuration y se encuentre en el mismo directorio.

Ahora puede utilizar un editor de textos (como vi) para observar las distintas opciones de configuración de dicho archivo.

Opciones de configuración

En el archivo Configuration encontrará cinco tipos de información:

- Comentarios: son las líneas que comienzan con el símbolo #.
- Opciones de construcción de archivos: líneas del tipo como CC=gcc, EXTRA_CFLAGS, etc.
- Opciones de reglas: líneas que comienzan por *Rules*.
- Opciones para añadir módulos: líneas que comienzan por *AddModule*.
- Opciones de módulos alternativos: comienzan por *%Module*.

Si desea efectuar la instalación típica, no tendrá que modificar ninguna de estas líneas. Pero si las cambia, mejor será que sepa lo que está haciendo.

Opciones de configuración relacionadas con la construcción de archivos

Estas opciones le permitirán especificar varios elementos de Makefile (Construcción de archivos) que se crearán por medio del script Configure. Para la mayoría de los sistemas no será necesario que modifique estos elementos. Este script tratará de ver qué compilador se utiliza en su sistema. En caso de que piense que no va a funcionar o que por alguna razón va a ser incapaz de localizarlo, se lo puede indicar usted mismo con:

```
CC
```

Por ejemplo:

```
CC = gcc
```

Ahora, si tiene que especificar algún parámetro externo (extra flags) al compilador de C para que los utilice al hacerse cargo de código, podrá utilizar las siguientes líneas:

```
EXTRA_CFLAGS=
EXTRA_LDFLAGS=
```

Si el sistema va a necesitar alguna librería especial o incluir un archivo determinado, también se puede especificar:

```
EXTRA_LIBS=
EXTRA_INCLUDES=
```

Observe que el script Configure automáticamente optimiza el código a -O2. Si desea utilizar una configuración diferente, quite el símbolo de comentario que se encuentra al principio de la línea:

```
#OPTIM= O2
```

Cambie el valor al deseado, siempre y cuando su compilador de C pueda trabajar con él. Para la mayoría de las instalaciones la configuración predeterminada funciona perfectamente bien. Yo lo comprobé con un sistema RedHat Linux.

Configuración de reglas

A continuación tiene que decidir la funcionalidad de Configure. Las opciones son las siguientes:

- `Rule STATUS=yes`
- `Rule SOCKS4=no`
- `Rule WANTHSREGEX=default`
- `Rule IRIXNIS=no`

La primera regla indica que desea que la funcionalidad del servidor será STATUS. Es decir, que quiere utilizar Apache en status_module, con lo que el servidor le mostrará cierta información relacionada con su efectividad. Para que esta regla sea efectiva, tendrá que añadir status_module en el área de configuración de módulos.

Por defecto, la funcionalidad STOCKS4 está desactivada. Para aquellos que no sepan qué es STOCKS4, les aclararé que es un sistema de control donde todos los datos de las aplicaciones de la red TCP/IP pasan a través del demonio STOCKS4. De esta forma STOCKS recopilará, registrará, mostrará en pantalla, filtrará y controlará todos los datos transmitidos por la red. La mayoría de la gente utiliza este software como un firewall. Si quiere que Apache sea compatible con este demonio, tendrá que activar dicha propiedad cambiando "no" y escribiendo "yes". Además ha de modificar el parámetro EXTRA_LIBS que se encuentra en el área de configuración Makefile para que se dirija a la librería STOCKS4. Si no lo hace, el script Configuration utilizará -L/usr/local/lib-lsocks.

El valor por omisión de WANTHSREGEX es Default (Predeterminado). Indica que se ha de utilizar el paquete de expresión normal que se incluye con Apache. Si desea utilizar su propio paquete, tendrá que modificar su valor y escribir "no" detrás del signo igual.

La opción IRIXNIS está pensada para aquellos que quieren utilizar Apache con sistemas Silicon Graphics en los que se utilice un procesador IRIX o bien NIS.

Configuración de módulos

Un módulo es un componente de software que añade alguna funcionalidad más a Apache. Con la configuración de módulos puede definir la funcionalidad de su servidor.

La línea de configuración es similar a la siguiente:

```
AddModule modules/standard/mod_env.o
```

Con esta línea añadirá el módulo mod_env que se encuentra en el directorio src/modules/estándar.

Nota: las antiguas versiones de Apache (anteriores a 1.2.x) utilizan el siguiente formato para la configurar los módulos:

```
Module nombre_module archivo_objeto_modulo
```

Los módulos predeterminados de Apache son los que aparecen en el listado 2.1.

Son más que suficientes para una instalación típica de Apache. No se ha de olvidar de que cada módulo añade más espacio de disco y memoria a los requisitos del sistema al ejecutable de Apache (httpd), por lo que debería repasarla atentamente para ver si puede prescindir de alguno de ellos. De esta forma su servidor será menos pesado y podrá ir más rápido. Por otro lado, si se encuentra con que uno de los módulos que le puede llegar a resultar de utilidad tiene el carácter #, quítelo para activarlo. Del mismo modo, si quiere desactivar uno de los módulos que no le va a servir de nada, añada el símbolo de comentario al principio de la línea.

```
AddModule modules/standard/mod_env.o
AddModule modules/standard/mod_log_config.o
AddModule modules/standard/mod_mime.o
AddModule modules/standard/mod_negotiation.o
AddModule modules/standard/mod_include.o
AddModule modules/standard/mod_autoindex.o
AddModule modules/standard/mod_dir.o
AddModule modules/standard/mod_cgi.o
AddModule modules/standard/mod_asis.o
AddModule modules/standard/mod_imap.o
AddModule modules/standard/mod_actions.o
AddModule modules/standard/mod_userdir.o
AddModule modules/standard/mod_alias.o
AddModule modules/standard/mod_access.o
AddModule modules/standard/mod_auth.o
AddModule modules/standard/mod_setenvif.o
```

Listado 2.1. Módulos que se cargan por defecto en el archivo Configuration

Si desea a la regla STATUS le asigna el valor "yes", tendrá entonces que eliminar el símbolo de comentario que se encuentra al principio de la línea de status_module:

```
AddModule modules/standard/ mod_status.o
```

Si no lo quita, dicha regla se ignorará. Ya está listo para ejecutar el script Configure.

Ejecutar Configure

Para ejecutar el script Configure, tendrá que irse al directorio src (si ha seguido las instrucciones que le he ido facilitando, ya debería encontrarse allí). Escriba:

```
./Configure
```

en su línea de comandos. Configure crea Makefile. Ahora puede compilar el archivo de código fuente.

Compilación de Apache

Tiene que ejecutar la utilidad make para que compile el archivo fuente. En la línea de comandos escriba:

```
make
```

Dicha utilidad usará el archivo Makefile que creó el script Configure y arrancará el compilador de C para crear el ejecutable de Apache (httpd).

La cantidad de tiempo que se empleará en la compilación dependerá de la rapidez y recursos libres del sistema que tenga su ordenador. Obviamente, cuanta más CPU y memoria RAM tenga, tanto mejor. Si por alguna razón fallase la compilación del archivo, aparecerá un mensaje de error en la pantalla. Anótelo y trate de descifrarlo. La mayoría de los errores se deben a que no se encuentra alguna de las librerías del sistema que el compilador necesita más parámetros. En cualquier caso, tendrá que ajustar las opciones de Makefile que se encuentran dentro del archivo Configuration (a menos que se atreva a modificar directamente el archivo Makefile) y repetir el proceso de compilación. Si aún así persiste el problema, búsquese a alguien que conozca el sistema operativo y el compilador de C y pídale ayuda.

Si no conoce a nadie que cumpla estos requisitos, publique un mensaje de ayuda en un grupo de news dedicado a tal tema en el que explique el problema. Incluya el mensaje de error.

Si todo va bien, se encontrará con el archivo httpd en el directorio src.

Probar Apache

Bueno, ha llegado la hora de comprobar si el archivo que acaba de compilar es realmente ejecutable. Lo primero que va a hacer es listar el contenido del directorio para ver la configuración de permisos. Por ejemplo:

```
ls -l httpd
```

Al escribir esta sentencia en el directorio src, accederá a los permisos de archivos. Asegúrese de que es el único que puede leer y ejecutar el fichero httpd. Para comprobar la versión de Apache escriba:

```
./httpd -v
```

Se le mostrará el número de la versión de su servidor Apache. Si coincide con la versión del código que se ha bajado por Internet, ¡enhorabuena!

Ahora querrá compilar las utilidades. Apache tiene unas cuantas utilidades que le ayudarán en la administración del sistema. Puede compilarlos desde el directorio de apoyo escribiendo:

```
make all
```

Se compilarán todas las utilidades y se crearán archivos como htpasswd, htdigest, httpd_monitor, rotatelogs, logresolve, etc. Obsérvese que el comando anterior no compila el programa suexec. Sabrá por qué en capítulos posteriores.

Se puede saltar la siguiente sección y empezar a trabajar en la creación de los directorios de Apache.

Uso de los archivos binarios

La verdad es que compilar el código fuente en vez de utilizar directamente los archivos binarios es una buena idea. Cuando utiliza uno de estos ficheros está dejando que una tercera persona (posiblemente alguien que no conozca) decida los módulos y propiedades que va a usar su servidor Web. Se encontrará con archivos binarios que no funcionarán en su sistema. Y es que habrá una serie de incompatibilidades entre las librerías necesarias y las que tiene su sistema. Si estos inconvenientes son aceptables, o si prefiere no meterse con la compilación del ejecutable, asegúrese de que el sitio del cual se va a bajar los archivos ejecutables es de confianza. En el sitio Web www.apache.org encontrará ficheros binarios para casi todos los sistemas.

Guarde el archivo ejecutable en un directorio. Localice el fichero httpd y verifique que sus permisos de administrador le permiten ejecutarlo. Utilice el siguiente comando para acceder al número de la versión:

```
./httpd -v
```

Si la versión coincide con la del software que ha bajado de Internet, querrá decir que está listo para trabajar en la creación de los directorios de Apache.

Truco: los usuarios de Linux RedHat pueden bajarse una versión RPM de Apache de ftp://ftp.redhat.com/. Con ella simplificarán el proceso de instalación.

Creación de los directorios de Apache

Puede guardar los ejecutables de Apache donde desee, pero siempre es conveniente ser una persona organizada desde el principio; aquí tiene unas cuantas sugerencias.

Cree un directorio llamado Apache que se encuentre en una buena ubicación (por ejemplo, dentro de las carpetas /usr/local o /isr/home). A partir de ahora me referiré a este directorio como %ApacheRoot%. A continuación cree los siguientes directorios:

```
%ApacheRoot%/bin
%ApacheRoot%/conf
%ApacheRoot%/logs
```

Copie el archivo httpd y todos los ficheros compilados en el directorio %ApacheRoot%/bin. Los archivos de configuración los deberá copiar a %ApacheRoot%/conf.

Es conveniente crear un usuario y un grupo de usuarios nuevo. Así tendrá un mayor control sobre las zonas de su disco a las que podrá acceder Apache. Yo lo hice. Ahora puede configurar los permisos de los directorios y ejecutables.

No permita que nadie que no sea el usuario principal (root user) acceda a los ejecutables ni a los archivos de configuración. Utilice el comando chown. Gracias a él limitará el acceso a los ejecutables al usuario principal y a los pertenecientes al grupo que acaba de crear. Los comandos son:

```
chown -R root.root %ApacheRoot%/bin
chown -R root.root %ApacheRoot%/conf
```

Los comandos chown configuran los permisos del propietario, pero para que dichos permisos tengan vigor, hay que definirlos. Como quiere que el usuario principal tenga acceso completo (lectura, escritura y ejecución) a los directorios bin y conf y que el usuario de httpd tenga un acceso completo al directorio de registros, tendrá que utilizar el siguiente comando chmod:

```
chmod -R 700 %ApacheRoot%
```

Con esta orden determina los permisos de acceso de tal forma que sólo los propietarios de los directorios puedan acceder a su contenido.

Advertencia: es muy importante que los registros que se encuentran en el directorio %ApacheRoot% sólo los pueda leer y escribir el propietario de la cuenta que ejecutará Apache.

Instalación de Apache

Antes de instalar archivo binario httpd en la dirección deseada, conviene ajustar algunos símbolos extra. Para ello se utiliza el comando:

```
strip httpd
```

Este comando elimina la información extra del archivo binario y reduce su tamaño.

Ahora puede instalar Apache. Siga estos pasos:

1. Copie el archivo binario httpd en el directorio %ApacheRoot%/bin.
2. Si ha compilado las utilidades de ayuda, cópielas ahora en el directorio %ApacheRoot%/bin.
3. Copie los archivos de configuración de la versión de distribución de Apache (como por ejemplo, acces.conf-dist, httpd.conf-dist, mime.types y srm.conf-dist) en el directorio %ApacheRoot%/conf.

Ya ha terminado la instalación de Apache.

Esté al día con el desarrollo de Apache

Para cuando complete la instalación de Apache es posible que se pregunte si no habrá salido otra versión al mercado, o incluso si no habrá aparecido algún nuevo parche de seguridad. Yo me lo pregunto con la mayoría del software que utilizo. Los programas cambian con demasiada rapidez, algo que es bueno (aunque pueda parecer lo contrario si tiene bastante trabajo entre manos). De todas formas no hace falta que se atormente con este tipo de problemas, por lo menos en lo referente a Apache. Suscríbase, gratuitamente, a uno de los mejores recursos especializados en este servidor, cuyo nombre es

Apache Week. Le mandarán a través del correo electrónico todas las novedades que vayan apareciendo relacionadas con Apache. Su sitio Web es:

```
http://www.apacheweek.com/
```

La información aquí facilitada le será de gran ayuda a todos aquellos administradores de Apache que quieren conocer las últimas novedades. También tendrá acceso a numerosos artículos a través de los cuales aprenderá a obtener lo mejor de su servidor. Le recomiendo que le eche un vistazo.

3 Configurar y ejecutar Apache

En el último capítulo ha aprendido a compilar el servidor Web Apache para que funcione en un sistema Unix. Ahora está todo listo para proceder a su configuración y ejecución. Sin embargo, antes de nada tendrá que editar unos cuantos archivos de configuración. En este capítulo nos vamos a centrar en este trabajo.

Apache lee cuatro ficheros de configuración: httpd.conf, access.conf, srm.conf y mime.type. Los sitios Web usuales no tienen que modificar el último archivo, por lo que podrán saltarse la sección correspondiente. Sin embargo, los otros tres son los que definen el comportamiento del servidor Apache porque se leen durante el inicio del sistema.

En este capítulo les mostrará cómo configurar cada uno de estos archivos. También veremos diferentes métodos de ejecución del servidor y sus procesos de inicio y detención. Al final del capítulo le enseñaré cómo efectuar una serie de comprobaciones.

Configurar el servidor

Todas las fuentes de distribución de Apache incluyen unos cuantos archivos de configuración como ejemplos. En una fuente estándar encontrará un

directorio llamado conf en el que se encuentran los archivos con configuración con la extensión -dist.

Lo primero que tendrá que hacer antes de modificarlos será copiarlos como sigue:

httpd.conf-dist a http.conf
access.conf-dist a access.conf
srm.conf-dist a srm.conf

Aunque hay tres archivos todos comparten la misma estructura. Son de texto y tienen dos tipos de información: comentarios y órdenes. Las primeras comienzan con el símbolo # y se supone que el software las ignora; se incluyen para informar al administrador del sistema. Se pueden añadir tantos comentarios como se quiera.

A excepción de los comentarios y las líneas en blanco, el servidor considerará que el resto del documento serán órdenes que ha de cumplir. Por medio de estas órdenes se le indica al servidor que haga algo de una forma concreta. A la hora de modificar estos archivos tendrá que tomar una serie de decisiones relacionadas con el comportamiento del sistema. En la siguiente sección aprenderá qué significa cada una de estas directrices y cómo se pueden utilizar para personalizar el comportamiento del servidor.

httpd.conf

httpd.conf es el archivo principal de configuración. Se utiliza para indicarle al servidor qué programas ha de ejecutar. En el listado 3.1 se muestra el fichero httpd.conf que se ha creado a partir de http.conf-dist.

```
# Este es el fichero principal de configuración. Si desea más
# información consulte el URL http://www.apache.org/.

# No se limite a leer este archivo sin comprender qué hace. Si
# tuviese alguna duda puede consultar los ficheros de ayuda. Ha sido
# advertido.

# Creado por Rob McCool

# ServerType se puede ejecutar tanto en modo "inetd" o "standalone".

ServerType standalone

# Si ejecuta el servidor en modo inetd, salte a "ServerAdmin".

# Port: indica el puerto que se escucha. Para todos aquellos puertos
# < 1023, tendrá que ejecutar httpd como root desde el principio.
```

```
Port 80

# HostnameLookups: Registra el nombre o los números IP de los
# clientes por ejemplo www.apache.org (on) o 204.62.129.132 (off)
# Quizás sea conveniente que desactive esta opción hasta que vaya a
# utilizar esta información en sus archivos de registro o con un
# CGI. Si lo deja activado es posible que reduzca el rendimiento de
# su sitio.
HostnameLookups on

# Si desea ejecutar Httpd con un usuario o grupo diferente tendrá
# que ejecutarlo como root desde el principio.

# User/Group: el nombre (o #número) del usuario/grupo para el que
# se ejecuta httpd.
#   En SCO (ODT 3) utilice User nouser y Group nogroup
#   En HPUX es posible que nadie pueda utilizar la memoria estándar,
#   por lo que tendrá que crear un usuario www y utilizarlo para
#   solventar este impedimento.
User nobody
Group #-1

# La siguiente directriz desactiva los "recuerdos" y la limpieza de
# las cabeceras HTTP de # Netscape 2.x y de todos aquellos
# exploradores Web que los utilicen. Se sabe de la existencia de
# problemas.

BrowserMatch Mozilla/2 nokeepalive

# ServerAdmin: la dirección a la que se dirigirán todos los mensajes
# relacionados con los problemas que pueda tener el servidor.

ServerAdmin you@your.address

# ServerRoot: directorio en el que se guardan los archivos de
configuración, error y registro del servidor

ServerRoot /usr/local/etc/httpd

# BindAddress: gracias a esta opción puede trabajar con servidores
# virtuales. Se utiliza para indicarle al servidor a qué dirección
# IP tiene que prestarle atención. Dicha dirección puede contener
# comodines "*", una IP o el nombre completo de un dominio de
# Internet.
# Consulte la directiva.

#BindAddress *

# ErrorLog: ubicación del archivo de registro de errores. Si no
# comienza con /, ServerRoot está listo.

ErrorLog logs/error_log

# TransferLog: ubicación del archivo de registro de transferencias.
# Si no comienza con /, ServerRoot está listo.
```

```
TransferLog logs/access_log

# PidFile: archivo donde el servidor registrará sus propios
# problemas.
PidFile logs/httpd.pid

# ScoreBoardFile: archivo en el que se guardará la información
# relacionada con los procesos internos del servidor.
# No hay que utilizarlo con todas las arquitecturas. Pero si con la
# que está trabajando la necesita (lo sabrá porque se creará este
# archivo en el proceso de creación de Apache), tendrá que
# asegurarse de que no hay dos llamadas que compartan el mismo
# archivo.
ScoreBoardFile logs/apache_status

# ServerName le permite especificar el nombre que el servidor
# enviará a los clientes si varía de un programa a otro (por
# ejemplo, utilice "www" en vez del nombre real del servidor).
#
# Nota: no puede inventarse nombres para el servidor y esperar que
# funcionen. El nombre que se utilice tiene que ser un DNS válido.
# Si no comprende cómo funciona, consulte a su administrador de red.

#ServerName new.host.name

# CacheNegotiatedDocs: Por defecto, Apache envía Pragma: no cache
# con cada documento que se negocia basándose en el contenido. Se le
# pide a los servidores proxy que no utilicen la memoria caché con
# este documento. Al eliminar el símbolo de comentario de la
# siguiente sentencia permitirá que se utilice la memoria caché.

#CacheNegotiatedDocs

# Timeout: número de segundos que pasarán antes de que se envíe la
# caducidad

Timeout 300

# KeepAlive: independientemente de la existencia de conexiones
# persistentes (más de una petición por conexión). Para desactivarlo
# utilice el valor "Off".

KeepAlive On

# MaxKeepAliveRequests: número máximo de peticiones permitidas en
# una conexión persistente. Para que no haya límite utilice el
# valor 0. Le recomendamos que, para optimizar el rendimiento,
# utilice un valor elevado.

MaxKeepAliveRequests 100

# KeepAliveTimeout: número de segundos que se esperará la recepción
# de la siguiente petición.

KeepAliveTimeout 15
```

```
# Regulación del tamaño de la pila del servidor. En vez de tratar de
# adivinar la cantidad de procesos que necesitará el servidor, Apache
# se adapta sobre la marcha y los carga; es decir, trata de mantener
# el suficiente número de procesos de carga además de unos cuantos
# para controlar las transmisiones (por ejemplo, varias peticiones
# simultáneas para un explorador de Netscape).

# Para ello comprueba la cantidad de veces que un servidor se queda
# esperando una petición. Si son menos que el valor de
# MinSpareServers, crea una de reserva. Si supera al valor de
# MaxSpareServers, eliminará las sobrantes. Los valores aquí
# determinados pueden ser completamente válidos para la mayoría de
# sitios.

MinSpareServers 5
MaxSpareServers 10

# Número de servidores que se inician; tendrá que ser un valor
# significativo.

StartServers 5

# Límite del número total de servidores que se ejecutan, por ejemplo,
# el límite de clientes que se pueden conectar a la vez. Si se
# alcanza el valor límite, se bloqueará el acceso del resto de
# clientes por lo que no conviene utilizar un número demasiado bajo.
# Con esto se trata de darle un respiro al sistema Unix para que no
# reduzca su velocidad.

MaxClients 150

# MaxRequestsPerChild: número de peticiones por subproceso
# permitidos antes de su eliminación.
#  Después de un uso prolongado del subproceso se cierra para evitar
#  que aparezcan problemas relacionados con el servidor Apache(y
#  posiblemente con las librerías que utiliza). En la mayoría de
#  los sistemas no es necesario, pero en unos pocos (como es el caso
#  de Solaris) la reducción del rendimiento de las librerías es
#  notable.

MaxRequestsPerChild 30

# Directivas del servidor proxy. Para activar el servidor proxy,
# quite el símbolo de comentario de la siguiente sentencia:

#ProxyRequests On

# Para activar la memoria caché, edite y suprima el símbolo de
# comentario de las siguientes líneas:

#CacheRoot /usr/local/etc/httpd/proxy
#CacheSize 5
#CacheGcInterval 4
#CacheMaxExpire 24
#CacheLastModifiedFactor 0.1
```

```
#CacheDefaultExpire 1
#NoCache un_dominio.com otro_dominio.edu joes.garage_venta.com

# Listen: permite ligar el servidor Apache a unas direcciones IP
# determinadas y/o puertos, además del configurado por omisión.
# Consulte también el comando VirtualHost.

#Listen 3000
#Listen 12.34.56.78:80

# VirtualHost: permite que un demonio responda a las peticiones
# procedentes de varias direcciones (correspondientes con un
# servidor). Para ello hay que configurar la máquina para que acepte
# paquetes IP procedentes de varias direcciones. Para ello se puede
# utilizar la etiqueta de alias o a través de parches den kernel
# como VIF.

# Las directrices de los archivos httpd.conf o srm.conf pueden
# incluirse en el comando VirtualHost.
# Consulte la entrada BindAddress.

#<VirtualHost servidor.algún_dominio.com>
#ServerAdmin webmaster@servidor.algún_dominio.com
#DocumentRoot /www/docs/servidor.algún_dominio.com
#ServerName servidor.algún_dominio.com
#ErrorLog logs/servidor.algún_dominio.com-error_log
#TransferLog logs/servidor.algún_dominio.com access_log
#</VirtualHost>
```

Listado 3.1. Archivo httpd.conf predeterminado creado a partir de http.conf-dist

Nota: tenga en cuenta que el objetivo de este capítulo consiste en configurar el servidor con la mínima configuración, por lo que no se va a profundizar en los detalles de las directrices de estos tres archivos de configuración.

La primera directriz que tendrá que configurar será:

```
ServerType Standalone | inetd
```

Especifica cómo se ejecutará el servidor. Se pueden utilizar dos métodos: standalone e inetd. A primera vista puede parecer que funcionalmente ambos son idénticos, pero es cierto que existe una gran diferencia relacionada con la eficacia del sistema. El proceso de un servidor inetd se cierra en el mismo momento en el que termina de atender la petición recibida. Mientras que en el modo standalone, el subproceso se queda esperando cierta cantidad de tiempo antes de cerrarse. De esta forma se pueden volver a utilizar en el futuro.

Como no hay que dedicar recursos de sistema para volver a abrir el proceso, se mejora la eficacia del servidor.

De todas formas también el modo inetd tiene sus ventajas. Por ejemplo, se le considera más seguro que standalone. Son muchos los administradores de sistemas que utilizan programas TCP para validar las peticiones antes de entregárselas al proceso correspondiente. Si tiene algún interés en la seguridad de su sitio y no tiene demasiado tráfico es posible que le conviniese más utilizar el modo inetd que standalone.

Vamos a ver cada modelo con más detalle.

Servidor Apache standalone

En este modelo (predeterminado), el valor de la directriz ServerTyper es standalone, que quiere decir que el servidor estará atento de un puerto para ver si recibe alguna petición. Cuando la máquina del cliente usa dicho puerto para ponerse en contacto con el servidor y enviarle una consulta, el servidor inicia un subproceso para que lo atienda. En la figura 3.1 se muestra el esquema de este modelo.

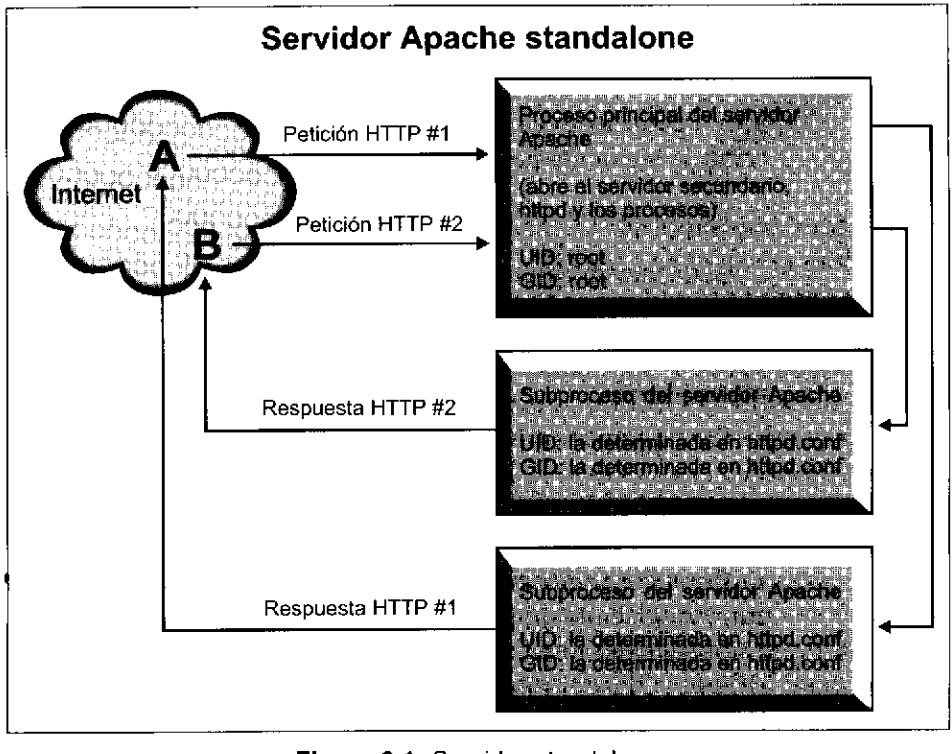


Figura 3.1. Servidor standalone

Puerto del servidor Apache standalone

La directriz le indica al servidor Apache principal el proceso al que tiene que prestar atención a través de un puerto en particular:

```
Puerto [número]
```

El puerto HTTP predeterminado es 80 y es que se utiliza en la mayoría de los sitios Web. Si no es el usuario root del sistema y desea ejecutar el servidor, tendrá que utilizar un puerto comprendido entre 1024 y 327697, ambos incluidos. Todos los puertos menores de 1024 se consideran estándares y para acceder a ellos se ha de tener el nivel inetd (nivel root). Si simplemente está practicando con un servidor Web y no tiene los permisos de la cuenta root tendrá que utilizar valores superiores al rango especificado. De esta forma se asegurará de que los valores especificados no los está utilizando ningún otro servidor. Si especificase un puerto que ya esté en uso, se encontraría con un mensaje de error al iniciar el servidor. Tenga en cuenta que si utiliza un puerto HTTP que no sea 80 (el predeterminado), tendrá que incluir este número en todas las URL dirigidas al servidor. Por ejemplo, si ha utilizado la siguiente directriz:

```
Port 8080
```

y va a pedir los recursos que se encuentran en mipágina.html, tendrá que escribir la siguiente petición:

```
http://www.suempresa.com:8080/mipagina.html
```

Si decide ejecutar los servidores Apache con el modelo standalone, tendrá que especificar los nombres del usuario y de los grupos.

Nombres de usuario y de grupos para el servidor Apache standalone

Para especificar estos nombres se utilizan dos directrices:

```
User [nombre usuario] #UID
Grupo [nombre grupo] #GID
```

Estas dos sentencias son muy importantes desde el punto de vista de la seguridad. Cuando un proceso del servidor principal abre un proceso en el servidor secundario para que se haga cargo de la petición del cliente, cambia el UID y el GID del subproceso de acuerdo con los valores de dichas directrices. Consulte la figura 3.1 para ver cómo se desarrolla el proceso. Si el subproceso se está ejecutando con los permisos root, será un peligro potencial ya que

estará a disposición de los hackers. Permitir que se pueda interactuar con los procesos del usuario root representa una brecha en la seguridad del sistema. De hecho no se recomienda que se trabaje en estas circunstancias. Mi consejo es que deje que los subprocesos tengan le mínimos permisos posibles. Para conseguirlos los podrá incluir en un grupo de usuarios que apenas tenga permiso para nada. En la mayoría de los sistemas Unix el usuario llamado nobody (generalmente UID=-1) y el grupo nogroup (GID=-1) son los que menos privilegios tienen. Para ajustar estos valores conviene que consulte los archivos que se encuentran en /etc/group y /etc/passwd.

Advertencia: si tiene previsto utilizar el servidor Web como un usuario no-root, sepa que no podrá cambiar el UID y GID de los subprocesos porque para ello necesita los permisos del usuario root. Por lo tanto, si ejecuta el servidor principal como un usuario llamado foobar, todos sus subprocesos tendrán los privilegios del usuario foobar. Lo mismo ocurre con los grupos de usuarios.

Tenga en cuenta que si su estrategia se basa en utilizar el formato numérico para los ID de los usuarios y/o grupos, tendrá entonces que colocar el símbolo # antes del valor numérico que se encuentra en los ficheros /etc/passwd y /etc/group.

Servidor Apache inetd

Otra forma de ejecutar el servidor Apache es con el método inetd. En este caso, el valor de la directriz ServerType será inetd, con lo que se modifica todo el proceso. En la figura 3.2 se puede ver cómo funciona. inetd es un demonio de Internet (un proceso del servidor) que escucha las peticiones que llegan a través de las conexiones que se establecen a través de los puertos menores a 1024. A diferencia del método anterior inetd controla qué peticiones se atienden. Cuando es uno de los clientes del sistema quien realiza la petición, inetd abre el proceso del servidor, le atiende y, cuando termina, lo cierra.

Si va a ejecutar su servidor Apache a través del modelo inetd, tendrá que editar el archivo /etc/inetd.conf para agregar un nuevo registro para Apache. Este archivo de texto tiene un formato especial. Para hacerse con él bastará con que le eche un vistazo a sus entradas. A menos que tenga un sistema Unix un tanto rarito, encontrará la siguiente sentencia en el archivo inetd.conf:

```
<nombre_del_servicio> <tipo_conexión> <protocolo> <etiquetas>
<usuario> <parth_del_servidor> <argumentos>
```

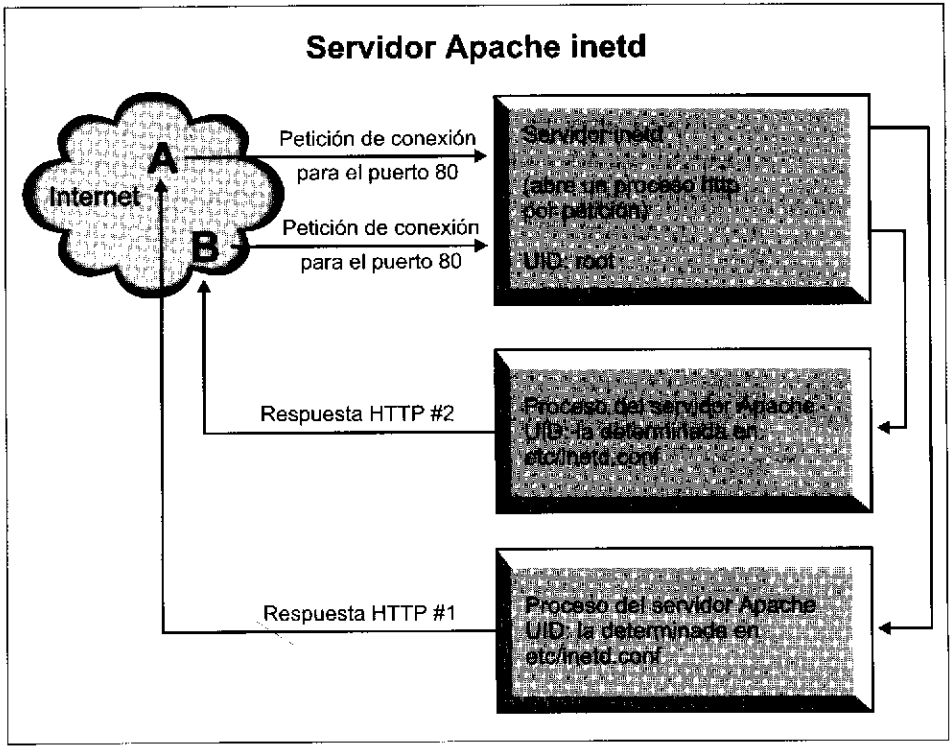


Figura 3.2. Servidor Apache inetd

Como se puede ver el servicio se ejecuta para un usuario determinado. Tendrá que decidir bajo qué usuario se ejecutará el servidor Apache. Lo más sencillo es utilizar el usuario nobody o crear uno especial llamado httpd que se encargue de la ejecución del servidor. Si utiliza el usuario nobody con otros servicios no convendrá que lo vuelva a utilizar con el servidor Apache. En caso contrario se encontrará que el servicio Web afectará a aquello a lo que se puede acceder cuando se utiliza la cuenta nobody para modificar los parámetros de archivo/directorio del resto de servicios. Mi consejo es que cree una cuenta httpd especial por medio del siguiente comando:

```
httpd stream tcp nowait httpd /usr/sbin/httpd -f
/etc/httpd/conf/httpd.conf .
```

Una vez que se modifique el archivo inetd.conf tendrá que hacer lo mismo con /etc/services:

```
<nombre_del_servicio> <número_del_puerto>/<nombre_protocolo>
<servicio_introducido_en_inetd.conf>
```

Así que la línea que ahora tiene que añadir al archivo /etc/services es la siguiente:

```
httpd 80/tcp httpd
```

La entrada anterior describe el servicio httpd que utilizará el servidor inetd. Especifica el servicio que estará a disposición del puerto 80. Si desea utilizar otro servicio Web (HTTP), sustituya 80 por el número del puerto que vaya a usar. Como todos los puertos que se encuentran por debajo de 1024 están reservados para los servicios estándar, tendrá que utilizar un número comprendido entre 1024 y 32768, ambos inclusive.

Ahora tendrá que reiniciar el proceso inetd. Primero necesitará el número ID del proceso (PID) que se puede obtener a través del comando:

```
ps auxw | grep inetd
```

Dependiendo del sistema Unix que se utilice es posible que tenga que utilizar otros argumentos con ps. Dirigiendo la salida generada por ps a grep permitirá que este último busque cualquier línea en la que aparezca *inetd* y la imprime en la pantalla (salida estándar). Aunque los formatos de la salida de ps varían de un sistema a otro, generalmente la primera columna numérica se corresponde al ID del proceso que se está ejecutando. Utilice la utilidad kill como sigue:

```
kill -HUP <PID de inetd>
```

No se olvide de sustituir la parte <PID de inetd> con el número ID del proceso. La utilidad kill enviará una señal HUP llamada PID. De esta forma se reiniciará el servidor con lo que se vuelve a leer el contenido de los archivos de configuración que acaba de modificar. Ahora la configuración del archivo inetd está completa.

Una vez que le asigna un valor a la directriz ServerType en inetd y configura los ficheros /etc/inetd.conf y /etc/services, las sentencias User y Group que se encuentran en httpd.conf dejarán de ser válidas. De todas formas, asegúrese de que el nombre de usuario que ha utilizado en /etc/inetd.conf tiene los permisos necesarios para poder acceder a los directorios Web y a aquellos en los que se encuentran los archivos de registro del servidor. Ya aprenderá a definirlo.

Mi consejo es que únicamente utilice la opción inetd con aquellos servidores Apache que tienen muy poca memoria RAM o con los que, en principio, no van a tener demasiado tráfico.

Directrices comunes

Los métodos standalone e inetd comparten una serie de directrices. La primera de la que hablaremos es la que le indica al servidor que muestre una página de error en la que aparezca una dirección de correo electrónico cada vez que aparezca un problema. De esta forma los visitantes de su sitio Web podrán informar de cualquier problema a través de un mensaje de correo electrónico. Para ello tendrá que usar la dirección electrónica de la persona encargada del mantenimiento del servidor.

Observe la siguiente sintaxis:

```
ServerAdmin [dirección e-mail]
```

Otra de las directrices comunes es ServerRoot que especifica dónde se encuentran los archivos de configuración, error y registro. Será el directorio padre de todos los archivos relacionados con el servidor. La localización predeterminada es /usr/local/etc/httpd. Si instaló el servidor en otro sitio, cámbielo por la ubicación que utilice su sistema. La sintaxis es:

```
ServerRoot [nombre completo del path]
```

Dos son las directrices que se refieren a los ficheros de información y registro de acceso del servidor. La primera le indica al servidor el path y el nombre del archivo de registro:

```
ErrorLog [nombre completo del path del archivo de registro]
```

La segunda le indica el path y el nombre del nombre del fichero en el que se guarda la información de acceso:

```
TransferLog [nombre completo del path del archivo de información
de acceso]
```

El nombre y la dirección predeterminados del archivo de registro de errores es logs/error_log, que quiere decir que el nombre del fichero es error_log y que se encuentra en un subdirectorio llamado logs. Si quiere guardarlo en una ubicación diferente, tendrá que especificar la dirección completa (es decir, que empezará con el símbolo "/"). Lo mismo ocurre con el archivo en el que se guarda la información de acceso, que se llama access_log y se encuentra dentro del subdirectorio logs.

Asegúrese de que el único proceso que tiene permiso para escribir en el directorio en el que se encuentran los archivos de registro es el proceso principal del servidor. Esta es otra medida de seguridad porque permitir que otros

usuarios o procesos puedan escribir en este directorio dejará que cualquier acceso no autorizado consiga el UID del proceso del servidor Web, que es el de la cuenta root.

Otra directriz común a ambos métodos es la que usa Apache para escribir el ID del proceso principal del servidor en un archivo. Por medio de PidFile podrá especificar el nombre y la ubicación del fichero en el que se guardará dicha información:

```
PidFile [nombre completo del path del archivo PID]
```

Recuerde que todo lo relacionado con la seguridad de los ficheros ErrorLog y TransferLog es aplicable a PidFile.

La última directriz en común del archivo httpd.conf que tendrá que configurar es la correspondiente al nombre del servidor de Internet.

```
ServerName [nombre del servidor]
```

Generalmente se utilizará un nombre del estilo www.suempresa.com. De todas formas asegúrese de que el nombre del dominio que escriba aquí coincide con el de su empresa. A continuación veremos el archivo de configuración srm.conf.

srm.conf

Este es el archivo de configuración de los recursos del sistema. Se utiliza para indicarle al servidor qué recursos puede ofrecer a través del sitio Web y cómo lo ha de hacer. En el listado 3.2 se muestra el archivo srm.conf predeterminado.

```
# Con este documento definirá el nombre del espacio que verán los
# usuarios de su servidor http. Este archivo incluye los parámetros
# del servidor que determinan cómo se atienden las peticiones y cómo
# se regula el formato de los resultados.

# Si desea más información puede consultar los tutoriales que se
# encuentran en http://www.apache.org/.

# Creado por Rob McCool. Adaptado para Apache

# DocumentRoot: directorio en el que se guardarán sus documentos.
# Por omisión, todas las peticiones se tomarán de este directorio,
# pero los enlaces simbólicos y los alias pueden apuntar a otras
# direcciones.

DocumentRoot /usr/local/etc/httpd/htdocs
```

```
# UserDir: el nombre del directorio asociado al del usuario (se
# utiliza cuando se recibe una petición ~user.

UserDir public_html

# DirectoryIndex: nombre del archivo o archivos que se utilizan
# como índice HTML ya escrito. Las entradas múltiples se separan
# por medio de espacios.

DirectoryIndex index.html

# FancyIndexing se utiliza para determinar si se trabaja con un
# directorio alternativo o con el estándar.

FancyIndexing on

# AddIcon le indica al servidor qué icono se utilizará para
# identificar a los distintos archivos o extensiones.

AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*

AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

# DefaultIcon es el icono que se utilizará con todos aquellos
# archivos que no tengan ninguno asignado.

DefaultIcon /icons/unknown.gif

# AddDescription le permite añadir una pequeña descripción en el
# índice de los archivos, después del nombre del fichero.
# Formato: AddDescription "descripción" nombre_archivo
```

```
# ReadmeName es el nombre del archivo README que buscará el servidor
# por defecto. Formato: Readmename nombre
#
# En primer lugar el servidor buscará nombre.html, si lo encuentra
# lo incluirá y, a continuación, repetirá la búsqueda de nombre en
# el texto plano. Si lo encuentra, también lo incluirá.
#
# HeaderName es el nombre del archivo en el que se encuentran los
# índices de los directorios.

ReadmeName README
HeaderName HEADER

# IndexIgnore es un conjunto de nombres de archivos que ignorará la
# indexación de directorios. Formato: IndexIgnore nombre1 nombre2...
IndexIgnore */.??* *~ *# */HEADER* */README* */RCS

# AccessFileName: en nombre del archivo que hay que buscar cuando
# se desee conocer la información de control de acceso de cada
# directorio.

AccessFileName .htaccess

# DefaultType nombre del tipo MIME predeterminado para los documentos
# cuyo tipo no pueda determinar el servidor (para ello se basa en la
# extensión de los mismos).

DefaultType text/plain

# AddEncoding permite descomprimir sobre la marcha cierta
# información relacionada con exploradores Web (Mosaic/X 2.1+).
# Nota: no todos los exploradores pueden trabajar con ella.

AddEncoding x-compress Z
AddEncoding x-gzip gz

# AddLanguage le permite que especifique el idioma de un documento.
# Gracias a ello podrá utilizar una negociación para que dicho
# archivo se le entregue al explorador adecuado. Tenga en cuenta que
# el sufijo no tiene que coincidir con la clave del idioma; los
# documentos en Polaco (aquellos con código pl) podrían utilizar
# "AddLanguage pl .po" para evitar la confusión entre el sufijo del
# idioma y la extensión de los scripts escritos en Perl.

AddLanguage en .en
AddLanguage fr .fr
AddLanguage de .de
AddLanguage da .da
AddLanguage el .el
AddLanguage it .it

# LanguagePriority le permite dar cierta precedencia a algunos
# lenguajes. Dicha preferencia se utilizará durante la negociación
# para darle cierto privilegio a algunos idiomas. Conviene que liste
# los lenguajes por orden decreciente de preferencia.
```

```
LanguagePriority en fr de

# Redirect se limita a informar a sus clientes de los documentos
# que hay en el espacio de su servidor, pero no permite nada más.
# De esta forma les podrá indicar dónde han de buscar un documento
# determinado.
# Formato: Redirect apodo url


# Aliases: añada aquí tantos alias como necesite (sin límite).
# Formato: Alias apodo nombre_real

# Obsérvese que si se incluye el símbolo / en el apodo, el servidor
# necesitará que esté presente en la URL. Por eso "/icons" no tiene
# asociado ningún alias en este ejemplo.

#Alias /icons/ /usr/local/etc/httpd/icons/

# ScriptAlias: controla los directorios que contienen scripts del
# servidor.
# Formato: ScriptAlias apodo nombre_real

#ScriptAlias /cgi-bin/ /usr/local/etc/httpd/cgi-bin/

# Si desea utilizar todo lo incluido en el servidor o los
# directorios CGI de ScriptAlias, quite el símbolo de comentario de
# las siguientes líneas.

# AddType le permite cambiar entre los distintos tipos de mime sin
# tener que editar o activar ciertos tipos.
# Formato: AddType type/subtype ext1

# AddHandler le permite asociar ciertas extensiones de archivos con
# acciones de controladores no asociadas a ese tipo de archivo. Así
# pueden construir dentro del servidor o añadir por medio del
# comando Action (ver más abajo)
# Formato: AddHandler nombre_acción ext1

# Para utilizar los scripts CGI:
#AddHandler cgi-script .cgi

# Para utilizar los archivos HTML propios del servidor
#AddType text/html .shtml
#AddHandler server-parsed .shtml

# Para activar la propiedad de envío de archivos HTTP de Apache,
# quite el símbolo de comentario de la línea siguiente
#AddHandler send-as-is asis

# Si desea utilizar archivos de mapas de imágenes propios del
# servidor, utilice la línea siguiente
#AddHandler imap-file map

# Para activar los tipos de mapas, es posible que desee utilizar:
#AddHandler type-map var
```

```
# Action le permite definir los tipos de medios que ejecutará un
# script siempre que se encuentre con una llamada a un archivo. De
# esta forma se elimina la necesidad de repetir los path de las URL
# para los procesadores de archivos CGI.
# Formato: Action medio/tipo /cgi-script/ubicación
# Formato: Action nombre_controlador /cgi-script/ubicación

# MetaDir: especifica el nombre del directorio en el que Apache
# encontrará información sobre los meta archivos. En estos ficheros
# se encuentran las cabeceras HTTP adicionales que se pueden
# utilizar al enviar el documento.

#MetaDir .web

# MetaSuffix: especifica el sufijo del archivo que contiene la meta
# información.

#MetaSuffix .meta

# La personalización de la respuesta a errores (estilo Apache) tiene
# tres inconvenientes
#
#    1) texto plano
#ErrorDocument 500 "El servidor ha hecho boo boo.
#  n.b.  las comillas (") lo marcan como texto y no se mostrará en
# la salida
#
#    2) redirección local
#ErrorDocument 404 /missing.html
#  para redirigir a la url local /missing.html
#ErrorDocument 404 /cgi-bin/missing_handler.pl
#  n.b. se puede redirigir a un script o a un documento utilizando
# las inclusiones del servidor.
#
#    3) redirección externa
#ErrorDocument 02 http://otro_servidor.com/info_subscripción.html
#
#
```

Listado 3.2. Archivo srm.conf predeterminado creado a partir de srm.conf-dist

En las próximas secciones se comentan las directrices de srm.conf que habría que configurar. Dichas órdenes se utilizan para crear la configuración de un directorio del sitio Web principal.

Configuración del directorio Web

Al igual que otros servidores Web, Apache tiene que conocer el path del directorio superior en el que se guardan las páginas Web. A este directorio se le conoce por root. Apache incluye una directriz llamada DocumentRoot que se utiliza para especificar el path de este directorio.

Esta directiva le indica al servidor que ha de considerar el directorio especificado como el path del de nivel superior. Para ello se utiliza la siguiente directriz:

```
DocumentRoot [path completo del directorio en el que se encuentran
los documentos Web]
```

La elección de este directorio es muy importante. Por ejemplo, si la directriz es la siguiente:

```
DocumentRoot /
```

entonces se podrá acceder a todos los archivos del sistema. Obviamente se puede proteger dichos archivos estableciendo los permisos oportunos, pero comprenderá que hacer que el directorio root virtual coincida con el físico del sistema supone todo un riesgo para la seguridad del servidor. Lo que se suele hacer es que el directorio root virtual coincida con uno de los múltiples subdirectorios del sistema. La configuración predeterminada es:

```
DocumentRoot /usr/local/etc/http/htdocs
```

De todas formas otra opción podría ser crear una estructura de directorios Web para su organización. En la figura 3.3 tiene la que yo utilizaría para un sistema con varios usuarios y dominios.

Como puede ver, he creado una partición llamada /www y en ella he creado los subdirectorios de cada uno de los sitios Web de mi empresa. /www/www.miempresa.com/ tiene tres subdirectorios: public, stage y development. Cada uno de ellos tiene otros dos: htdocs y cgi-bin. El primero es el root de los documentos, mientras que el segundo es el subdirectorio en el que se guardan los script CGI. De esta forma, el parámetro DocumentRoot para el sitio Web www.miempresa.com será:

```
DocumentRoot /www/www.miempresa.com/public/htdocs
```

La ventaja de esta estructura de subdirectorios es que guarda todos los documentos y las aplicaciones Web bajo una misma partición (/www). De esta forma se facilitan las copias de seguridad y se puede montar en diferentes sistemas a través de NFS (Network File System) siempre que haya que repartir las tareas de un sitio Web entre varias máquinas.

Observe que porque el root de sus documentos apunte a un directorio en particular no significa que el servidor Web no pueda acceder a los directorios que se encuentran fuera del árbol de dicho documento. Puede hacerlo a través de los enlaces simbólicos (siempre que cuente con los permisos pertinentes)

o a través de alias (los veremos en futuros capítulos). Desde el punto de vista de una empresa y de su seguridad no recomiendo que se utilicen demasiados enlaces virtuales para acceder a los documentos que se encuentran fuera del árbol de directorios. De todas formas a veces es conveniente colocar una serie de documentos fuera del árbol, incluso en los casos en los que tenga que permitir su acceso.

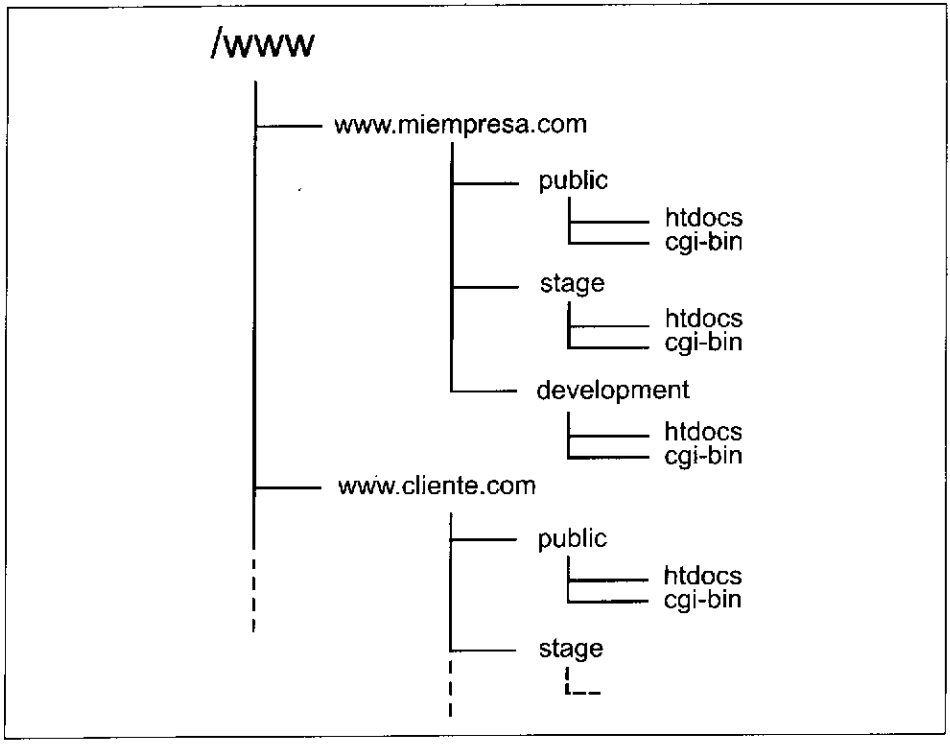


Figura 3.3. Estructura de directorios que utilizaría

Trate si tiene que añadir enlaces simbólicos a los documentos que se encuentran fuera del árbol de una partición, asegúrese de que a la hora de hacer copia de seguridad de su sitio Web, se le indica correctamente al programa cómo ha de tratar dichos enlaces.

Configuración del directorio cgi-bin

Si tiene previsto ejecutar script CGI es posible que desee guardarlos en un directorio creado para tal fin y que no coincida con aquel al que señala

DocumentRoot (o alguno de sus subdirectorios). El motivo es el de siempre, por seguridad. Si se dejan los script dentro del árbol de documentos, puede ser que alguien les eche un vistazo y localice algún punto débil. Esta es la razón por la que el árbol de directorios de la figura 3.3 incluye uno especialmente creado para los scripts, llamado cgi-bin, que se encuentra fuera del árbol de documentos de cada sitio Web.

Para que este directorio se encuentre fuera del árbol de documentos puede usar la directriz ScriptAlias para crear un alias nuevo para dicho directorio:

```
ScriptAlias [/alias/] [path completo del directorio de los scripts]
```

El alias (o apodo) es un nombre cualquiera que se asigna a un directorio físico en el que se guardan los script, applets, etc. El alias estándar es cgi-bin, pero puede utilizar cualquier otro. La orden predeterminada es la siguiente:

```
ScriptAlias /cgi-bin/ /usr/local/etc/httpd/cgi-bin/
```

Para el sitio www.miempresa.com que aparece en la figura 3.3, el alias sería el siguiente:

```
ScriptAlias /cgi-bin/ /www/www.miempresa.com/public/cgi bin
```

De esta forma cuando se quiera referir a un script que se encuentra en un URL o en un archivo HTML del árbol de documentos, tendrá que utilizar el alias en vez de su ubicación física. Por ejemplo:

```
http://www.miempresa.com/cgi-bin/finger.cgi
```

llama al script finger.cgi que se encuentra en el directorio /www/www.miempresa.com/public/cgi-bin/. Otra forma de mirar a este directorio sería:

```
ScriptAlias apodo nombre_real
```

La siguiente directriz a la que tiene que prestar atención es UserDir cuya sintaxis es la siguiente:

```
UserDir [nombre del directorio]
```

Esta directriz se utiliza para indicarle a Apache el directorio que ha de considerar como DocumentRoot para los usuarios del sistema. Solamente se utilizará si su sistema tiene varios usuarios y quiere permitir que cada uno tenga su propio sitio Web. La configuración predeterminada es:

```
UserDir public_html
```

que significa que si ha configurado el nombre de su servidor como www.suempresa.com y tiene dos usuarios (Joe y Jenny) sus URL personales serían los que aparecen en la tabla 3.1.

Tabla 3.1. Directorios Web personales de usuarios

URL	Directorio Web
`http://www.suempresa.com/~joe`	`~joe/public_html`
`http://www.suempresa.com/~jenny`	`~jenny/public_html`

Obsérvese que en los sistemas Unix, el símbolo ~ llama al directorio principal del usuario. El directorio especificado por la directriz UserDir se encuentra dentro del principal de cada usuario. Apache tiene que leer y ejecutar los permisos para leer los archivos que se encuentran dentro del directorio public_html. Para ello hay que utilizar el siguiente comando:

```
chown -R <user>.<nombre del grupo del servidor Apache>
  ~<user>/<directorio asignado en UserDir>
chmod -R 2770 ~<user>/<directorio asignado en UserDir>
```

Por ejemplo, si el nombre del usuario es joe, el grupo de Apache es httpd y la directriz UserDir ha asignado el directorio public_html, los comandos anteriores tendrán este aspecto:

```
chown -R joe.httpd ~joe/public_html
chmod -R 2770 ~joe/public_html
```

El primer comando, chown, cambia el propietario del directorio ~joe/public_html (y de todos los archivos y subdirectorios que contenga) a joe.httpd. En otras palabras, otorga al usuario joe y al grupo httpd plenos derechos sobre los archivos y directorios de public_html. El siguiente comando, chmod, ajusta los derechos de acceso a 2770, es decir, que tan solo el usuario joe y el grupo httpd tienen derecho para leer, escribir y ejecutar archivos en el directorio public_html y en todos sus contenidos. También se asegura que cuando se crea un directorio (o un archivo) nuevo en public_html, tiene el ID del grupo. De esta forma se permite que el servidor Web pueda acceder a todo lo que se acaba de crear sin la intervención del usuario.

Truco: si crea cuentas de usuarios en el sistema con un script (como /usr/sbin/adduser en los sistemas Linux), puede darse el caso de que quiera incorporar a este proceso la creación de un sitio Web. Bastará

agregar el comando mkdir para crear el directorio public_html (si es el que se ha asignado en la directriz UserDir) para crear un directorio Web. Añada los comandos chown y chmod para ajustar los permisos de lectura y ejecución de archivos a todo lo que se encuentre bajo este directorio.

La última directriz que necesita para configurar el directorio es DirectoryIndex, cuya sintaxis es la siguiente:

```
DirectoryIndex [nombre_archivo_1, nombre_archivo_2, nombre_archivo_3
... ]
```

Esta directriz especifica el archivo que tomará el servidor Apache como índice del directorio solicitado. Por ejemplo, cuando se accede a un URL como www.suempresa.com/, el servidor Apache determina si es un acceso a / (la raíz del documento), el directorio del sitio Web. Si la directriz DirectoryRoot es:

```
DirectoryRoot /www/www.suempresa.com/public/htdocs
```

entonces el servidor Apache buscará un archivo llamado /www/www.suempresa.com/public/htdocs/index.html; si lo encuentra, Apache atiende a la petición y le muestra al explorador Web del cliente el contenido de dicho archivo. Si DirectoryIndex está asignado al archivo de bienvenida welcome.html, el servidor Apache buscará /www/www.suempresa.com/public/htdocs/welcome.html. Si no lo encontrase, Apache mostraría la lista de contenido de directorios que se ha creado sobre la marcha en una página HTML. En la figura 3.4 tenemos un caso en el que se ha perdido el archivo index.html y el servidor ha generado un índice sobre la marcha para atender a la petición del servidor.

Desde la directriz se pueden especificar varios índices. Por ejemplo:

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

le indica al servidor Web que tiene que ver si está cualquiera de estos tres archivos y, si encuentra alguno de ellos, tendrá que mostrárselo al cliente que lo solicita.

Observe que mostrar varios archivos como índice implica que puede crear dos problemas. El primero es que el servidor tendrá que comprobar la existencia de todos ellos en cada uno de los directorios a los que accede, con lo que se terminará por reducir su velocidad. El segundo es que al tener varios

archivos como índices complica la organización del sitio Web. Pero si sus desarrolladores utilizan varios sistemas para crear los archivos, el uso de varios archivos puede ser la solución ideal. Por ejemplo, Windows 3.x es incapaz de trabajar con archivos cuya extensión tenga más de tres caracteres, por lo que un usuario que trabaje con dicho sistema tendrá que actualizar de forma manual todos los archivos index.htm que se encuentren en el servidor Web. Al utilizar los nombres recomendados en esta sección se elimina este problema.

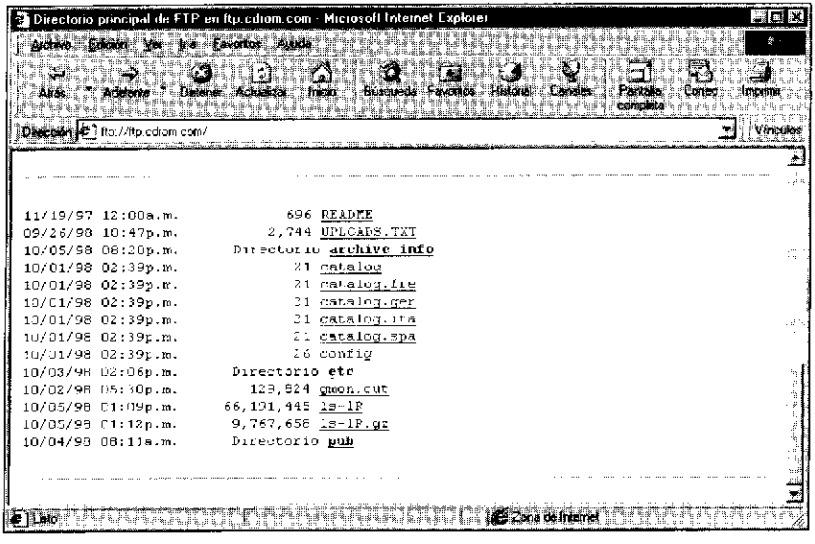


Figura 3.4. Directorio creado sobre la marcha

acces.conf

El último archivo que tiene que modificar es access.conf. Se utiliza para acceder a los permisos de los objetos (como archivos, directorios y script) que se encuentran en el sitio Web. El listado 3.3 nos muestra el contenido del archivo access.conf.

```
# access.conf: configuración global de acceso
# Documentos online en http://www.apache.org/

# Este archivo define la configuración del servidor relacionada con
# los permisos de los servicios y las circunstancias en las que se
# aplican.

# Cada directorio al que tiene acceso Apache se puede configurar
# con respecto a los servicios y las propiedades permitidas y/o
# desactivadas de un directorio (y de todos sus subdirectorios).
```

```
# Creado por Rob McCool

# Puede cambiar dependiendo el valor de la directriz DocumentRoot.

<Directory /usr/local/etc/httpd/htdocs>

# También puede ser "None" (ninguno), "All" (todo) o cualquier
# combinación de "Indexes", "Includes", "FollowSymLinks", "ExecCGI"
# o "MultiViews".

# Obsérvese que "MultiViews" se ha de nombrar *explícitamente*;
# "Options All" de momento no se lo permite.

Options Indexes FollowSymLinks

# Controla qué opciones de los archivos de .htaccess se pueden
# sobreescribir. También se pueden utilizar los valores "All" o
# cualquier combinación de "Options", "FileInfo", "AuthConfig" y
# "Limit".

AllowOverride None

# Controla quién puede acceder al material que contiene el servidor.

order allow,deny
allow from all

</Directory>

# /usr/local/etc/httpd/cgi-bin se puede cambiar a lo que haya en el
# directorio CGI definido en la directriz ScriptAlias, siempre que
# se configurase de antemano.

<Directory /usr/local/etc/httpd/cgi-bin>
AllowOverride None
Options None
</Directory>

# Permite que el servidor informe de su estado con el URL de
# http://servername/server-status
# Cambie ".your_domain.com" para que active su dominio.

#<Location /server-status>
#SetHandler server-status

#order deny,allow
#deny from all
#allow from .your_domain.com
#</Location>

# Hay informes en los que se comenta que ha habido gente que ha
# tratado de aprovecharse de un antiguo fallo anterior a la
# versión 1.1. Este fallo estaba relacionado con un script que
# se distribuía como parte de Apache.
# Quitando el símbolo de comentario de estas líneas podrá redirigir
```

```
# estos ataques al script que se encuentra en phf.apache.org. O, si
# lo desea, puede grabarlos utilizando el script
# support/phf_abuse_log.cgi.

#<Location /cgi-bin/phf*>
#deny from all
#ErrorDocument 403 http://phf.apache.org/phf_abuse_log.cgi
#</Location>

# A continuación puede colocar otros directorios o ubicaciones a
# cuya información desee acceder.
```

Listado 3.3. Archivo acces.conf predeterminado creado a partir de acces.conf-dist

Este es el único archivo de configuración que tendrá que modificar directrices de varias líneas de longitud. La primera directriz que tiene que retocar es la que presenta la siguiente sintaxis:

```
<Directorio directorio> ... </Directorio>
```

`<Directorio>` y `</Directorio>` se utilizan para encerrar un grupo de directrices. Su cantidad está limitada por el directorio nombrado (con sus subdirectorios); de todas formas se pueden utilizar únicamente las directrices permitidas en el contexto de un directorio (aprenderá más sobre las distintas directrices en futuros capítulos).

El directorio nombrado puede ser un path completo o bien una cadena de comodines. El archivo predeterminado access.conf tiene la siguiente configuración:

```
<Directory /usr/local/etc/httpd/htdocs>
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
</Directory>
```

Tiene que cambiar el directorio /usr/local/etc/httpd/htdocs al que aparezca como argumento de la directriz DocumentRoot que se encuentra en el archivo httpd.conf. En la configuración original, en la que se incluyen varias directrices como Options y AllowOverride le indica al servidor lo siguiente:

- El directorio nombrado y todos sus subdirectorios se pueden indexar. En otras palabras, si hay un archivo índice, se mostrará su contenido. Si no lo hay, se creará un índice sobre la marcha. Este comportamiento se especifica a través de Options.

- El directorio nombrado y todos sus subdirectorios pueden tener enlaces virtuales para que los siga el servidor (es decir, para que usen un path) y acceda a la información solicitada. Este comportamiento también se especifica a través de Options.
- Un archivo de control de acceso (que se especifica en la directriz AcessFileName que se encuentra en el fichero srm.conf, cuyo valor predeterminado es .htaccess) puede sobrescribir ninguna de las opciones que se especifican en la orden Directory. Este comportamiento se especifica a través de AllowOverdrive.
- Se permite acceder a todo.

De momento, la configuración predeterminada debería bastarnos. Pero si el servidor va a trabajar con Internet es posible que tenga que eliminar la opción FollowSymLinks que se encuentra en Options. Dejarlo supondría todo un riesgo para la seguridad del sistema. Por ejemplo, si uno de los directorios de sitio Web carece de archivo índice, el servidor mostrará automáticamente uno con una serie de enlaces que permitirán acceder a su contenido. De esta forma es posible acceder a la información vital o que alguien que carece de los permisos oportunos ejecute algún archivo binario que se encuentre en dicho directorio.

Iniciar y detener el servidor

Ahora que hemos completado todo el proceso de configuración estamos listos para ejecutar el servidor. Como ya he comentado, hay dos formas de hacerlo. Vamos a ver detenidamente cada método.

Servidor standalone

Si configura su servidor como standalone, las próximas secciones le enseñarán a ejecutarlo, detenerlo y reiniciarlo.

Ejecutar Apache como un servidor standalone

Para ejecutar su servidor standalone bastará con iniciar el programa httpd del servidor que ha instalado anteriormente en el sistema. Si dicho fichero binario se encuentra en el directorio /usr/sbin/, tendrá que entrar en el sistema como usuario root y ejecutarlo escribiendo:

```
/usr/sbin/httpd
```

Así abrirá httpd, el proceso principal del servidor Web, que buscará los archivos de configuración que se encuentren en la ubicación predeterminada (especificada durante la compilación), /usr/local/apache/conf/. Si los instala en otro directorio tendrá que indicarle a http dónde puede localizarlos a través del comando siguiente:

```
/usr/sbin/httpd -f /path/a/httpd.conf
```

donde /path/a/ es la dirección en la que se encuentra el archivo httpd.conf.

En cualquier caso, si no aparece ningún mensaje de error en la pantalla, accederá al prompt del shell, con lo que sabrá que el servidor funciona adecuadamente.

La forma más rápida de averiguar si el servidor está funcionando es a través del siguiente comando:

```
ps aux | grep httpd
```

Con ps se listan todos los procesos que se encuentran en la cola de procesos y el resultado se le entrega al programa grep. Se encarga de buscar la palabra httpd en la información que le acaban de entregar y muestra lo que ha encontrado en la pantalla del sistema. Si ve que una de las líneas contiene la palabra root querrá decir que es el proceso primario del servidor Apache. No olvide que cuando el servidor arranca, crea un número de subprocesos que se harán cargo de las peticiones recibidas. Si ejecuta Apache como usuario root, el proceso principal (o padre) se ejecutará como root, mientras que los subprocesos cambiarán según lo que se determine en el archivo httpd.conf.

Si httpd no puede acceder a una dirección, entones querrá decir que otro proceso está utilizando uno de los puertos que, por configuración, debería estar destinado a Apache, o que está ejecutando httpd como un usuario corriente y que el número del puerto se encuentra por debajo de 1024 (como es el caso del predeterminado, el puerto 80).

Si el servidor no se ejecuta, lea el mensaje de error que aparece en pantalla. Para obtener más información sobre el fallo tendrá que acceder al fichero error_log (registro de errores) que, según la configuración predeterminada, se encuentra en el directorio de registros, log.

Si desea que el servidor se inicie automáticamente después de que el sistema se reinicie, tendrá que añadir una llamada a los archivos de configuración del sistema (generalmente rc.local o un fichero que se encuentre en el directorio rc.N). En el listado 3.4 se muestra el contenido del script httpd.sh que se puede utilizar para que el servidor arranque automáticamente después de que se reinicie el sistema.

```
#!/bin/sh
#
# httpd   Este shell script inicia y detiene el servidor Apache
# Toma el argumento "start" o "stop" según lo que desee hacer con el
# proceso del servidor.
#
# Nota: Es posible que quiera cambiar el path que se utiliza en el
# script para reflejar la configuración del sistema.
#

[ -f /usr/sbin/httpd ] || exit 0

# Observe cómo se invoca al script.

case "$1" in
  start)
        # Inicia los demonios.
        echo -n "Starting httpd: "
        /usr/sbin/httpd
        touch /var/lock/subsys/httpd
        echo
        ;;
  stop)
        # Detiene los demonios.
        echo -n "Shutting down httpd: "
        kill -TERM `cat /usr/local/etc/httpd/httpd.pid`

        echo "done"
        rm -f /var/lock/subsys/httpd
        ;;
  *)
        echo "Usage: httpd {start|stop}"
        exit 1
esac
exit 0
```

Listado 3.4. El script httpd.sh

Para iniciar el servidor Apache habrá que ejecutar el script anterior escribiendo lo siguiente:

```
httpd.sh start
```

Para detenerlo:

```
httpd.sh stop
```

Tendrá que incluir estos comandos en los archivos rc apropiados del sistema. Puede utilizar los comandos de inicio que se encuentran en rc.local para abrir automáticamente el servidor después del arranque del sistema.

Detener un servidor Apache standalone

Para detener la versión standalone del servidor Apache tendrá que indicarle a Unix que utilice el comando kill para enviar una señal TERM. Esta le indica al programa del servidor (httpd) que se cierre. Pero antes de enviar dicha señal habrá que conocer el ID del proceso (PID) httpd que se está ejecutando como root.

Dicho número PID está escrito del archivo que se utiliza como argumento de la directriz PidFile. Este PID es del proceso httpd principal. No trate de cerrar los procesos afiliados porque el "padre" puede crearlos de nuevo. El comando típico para detener el servidor es:

```
kill -TERM 'cat /user/local/etc/apache/logs/httpd.pid'
```

Si especifica un path distinto, entonces se utilizará el predefinido en PidFile.

Reiniciar un servidor standalone

Si modifica alguno de los archivos de configuración de Apache y quiere que los servidores que está ejecutando vuelvan a leer los archivos de configuración, no hará falta que finalice los procesos del servidor. Bastará con que reinicie el proceso principal del servidor Web con la señal HUP:

```
kill -HUP 'cat /user/local/etc/apache/logs/httpd.pid'
```

De esta forma obligará a que el proceso primario del servidor Web vuelva a leer los archivos de configuración.

Servidor Apache inetd

Si ejecuta el servidor a través del modelo inetd, en las siguientes secciones aprenderá a ejecutarlo, detenerlo y reiniciarlo.

Ejecutar el servidor Apache inetd

Si ha configurado correctamente el modelo inetd de httpd no tendrá que hacer nada para ejecutar el servidor Apache ya que inetd lo inicia cuando recibe una petición del puerto HTTP.

Detener el servidor Apache inetd

Lo normal es que no tenga que preocuparse de finalizar el servidor Apache inetd porque desaparece en el mismo momento en que se termina de atender la solicitud recibida. De todas formas, si por alguna razón tiene que salir de él

por algún motivo extraño (porque se cuelgue el servidor, por ejemplo) tendrá que determinar el PID del proceso:

```
ps ausx   grep httpd
```

Recuerde que puede utilizar distintos argumentos con la utilidad ps de su sistema. Con ps se listan todos los procesos que se encuentran en la cola de procesos y el resultado se le entrega al programa grep. Se encarga de buscar la palabra httpd en la información que le acaban de entregar y muestra lo que ha encontrado en la pantalla del sistema.

Ahora puede determinar el PID a partir de la salida generada por grep. Si no puede hacerlo, ejecute la parte ps del comando y en la primera línea de la salida que genere se le indicará la columna en la que aparecen los PID de todos los procesos. Una vez que localiza el PID tendrá que ejecutar el siguiente comando:

```
kill -TERM [PID]
```

Reiniciar el servidor Apache inetd

Tampoco es necesario volver a iniciar el servidor inetd, porque se abre un nuevo servidor con cada solicitud recibida. Si modifica el archivo de configuración, los cambios serán efectivos con la próxima copia de servidor que se abra.

Comprobar el servidor Apache

Ahora está listo para comprobar el servidor Apache. La primera prueba es muy sencilla.

Ejecute su explorador favorito y diríjalo hacia el sitio Web que dirige el servidor Apache. Si el explorador se encuentra en el mismo sistema en el que está Apache, bastará con que escriba el siguiente URL:

```
http://localhost/
```

De todas formas, en cualquier otro caso tendrá que especificar el nombre exacto del servidor (como por ejemplo, www.suempresa.com).

Si no ha hecho ningún cambio a los archivos que se encuentran en el directorio htdocs se encontrará entonces con una página HTML con el logotipo de Apache en el cual se le indicará que el servidor funciona de manera satisfactoria.

Por último, si quiere asegurarse de que sus archivos de registro se actualizan correctamente, tendrá que acceder al directorio log (registros) y escribir el siguiente comando de Unix:

```
tail -f [nombre del archivo de registros]
```

La parte tail del comando es una utilidad de Unix que permite visualizar la parte nueva de un archivo que va creciendo poco a poco (sólo cuando se especifica la opción -f). Ahora utilice un explorador Web para acceder a ese sitio y, si ya se encontraba en él, actualice el contenido de la pantalla. Verá que se ha añadido una entrada a la lista que aparece en la pantalla. Pulse de nuevo el botón Actualizar y observe cómo aparece otra entrada. Esto indica que el archivo de registro funciona correctamente, Para salir de la sesión tail pulse Ctrl+C. Si no aparece ningún registro nuevo tendrá que consultar los permisos del archivo de registro y del directorio en el que se encuentra.

Otro registro que habrá que comprobar será el de errores. Utilice

```
tail -f [nombre del registro de errores]
```

para ver las entradas de dicho archivo según vayan teniendo lugar. Todo lo que tiene que hacer es solicitar algún recurso que no tenga el sistema (como un archivo inexistente) desde el explorador Web y verá cómo se van añadiendo nuevas entradas. En este caso el registro de errores está correctamente configurado.

Si el resultado de todas las pruebas fuese satisfactorio, entonces tendrá bien configurado su servidor Apache. ¡Enhorabuena!

4 Las directrices del núcleo

Una directriz no es otra cosa que un comando que le indica a Apache cómo ha de actuar. Apache lee las directrices que se encuentran en los archivos de configuración vistos en capítulos anteriores. Mediante estas órdenes, un administrador de Apache puede controlar el comportamiento del servidor Web. Son muchas las directrices que se pueden utilizar con Apache, por lo que las opciones de configuración del servidor Web son enormes. Las que forman parte de la base de la instalación de Apache reciben el nombre de directrices del núcleo. Siempre están disponibles. Otras directrices están incluidas en los módulos estándar que forman parte de la distribución estándar de Apache.

En este capítulo veremos los contextos estándar de dichas directrices, tanto las del núcleo como las otras, pero sobre todo nos centraremos en las primeras. En vez de mostrar una lista con todas las directrices distribuidas por orden alfabético, las he agrupado por utilidad; las categorías que se incluyen son configuración general, eficacia y configuración del recurso, contenedor estándar, servidor virtual específico, registro, autenticación y seguridad. En cada descripción encontrará la siguiente información:

- Sintaxis: muestra el nombre de la directriz y todos los argumentos o valores con los que puede trabajar.
- Contexto: especifica el contexto (o alcance) en el que se aplica la orden.

- Predeterminado: en esta línea de muestra el valor predeterminado. Sólo aparecerá cuando se pueda aplicar.
- AllowOverride: valor que activa la directriz en el archivo de configuración de acceso al directorio (.htaccess). Sólo aparecerá cuando se pueda aplicar.
- Compatibilidad: en esta línea se le muestran los detalles de la versión de Apache relacionados con la directriz.

Pero antes de entrar en las secciones de referencia de las directrices del núcleo, vamos a echar un vistazo al contexto en el que se pueden utilizar.

Contexto de las directrices de Apache

Antes de que se decida a utilizar ninguna de las directrices del núcleo es muy importante que comprenda el contexto en el que se pueden usar. En otras palabras, tiene que conocer el alcance de la directriz. Una vez que comprenda la terminología podrá meterse en profundidad con las órdenes.

Principalmente hay tres contextos. Una directriz puede aparecer en cualquiera de los archivos de configuración del servidor principal (contexto de configuración del servidor), fuera de los contenedores (algo parecido a las etiquetas HTML); entre contenedores (contenedores de contexto) o; en los archivos de configuración periódica (contexto por directorio).

Contexto de configuración del servidor

Las directrices pueden aparecer en cualquiera de los archivos de configuración del servidor principal fuera de cualquier contenedor. Cuando hablamos de contexto nos referimos a un alcance global. Por omisión, cuando se aplica una directriz en un contexto, afectará a todos los demás. Una regla de oro es tratar a una directriz que no esté encerrada en un contenedor como global. Se pueden utilizar en cualquiera de los archivos de configuración del servidor (como httpd.conf, srm.conf y access.conf) pero no dentro de cualquier contenedor o archivo de configuración por directorio.

Contexto del contenedor

Los contenedores se utilizan para limitar el alcance de una directriz. Una pareja de etiquetas agrupa las directrices y de esta forma restringen su alcance. Apache trabaja con los siguientes contenedores:

- <VirtualHost ...> ... </VirtualHost> se utiliza para aplicar una o más directrices al servidor especificado en la etiqueta de apertura del contenedor.
- <Directory ...> ... </Directory> se utiliza para aplicar más directrices a un directorio determinado. Obsérvese que si especifica una serie de directrices para que se apliquen a un directorio, a través de este contenedor, también se aplicarán a todos sus subdirectorios. De todas formas, si no desea que se produzca este efecto puede crear un contenedor por separado para cada subdirectorio, con lo que conseguirá que el servidor tenga un comportamiento distinto según el directorio en el que se encuentre.
- <DirectoryMatch regex> ... </DirectoryMatch regex> es igual que el contenedor <Directory>, sin embargo tiene una expresión regular (regex) que utiliza como argumento sustituyendo al nombre del directorio.

Nota: una expresión regular está compuesta por caracteres normales y especiales y se utiliza para crear un modelo. Dicho modelo se utiliza para marcar una o más subcadenas o una cadena completa.

- <Files ...> ... </Files> se utiliza para aplicar una o más directrices a un grupo de archivos.
- <FilesMatch regex> ... </FilesMatch regex> es exactamente igual que el contenedor <Files>; de todas formas utiliza una expresión regular como argumento en vez de tomar el nombre de uno o más archivos.
- <Location> ... </Location> se utiliza para aplicar una o más directrices a un URL determinado.
- <LocationMatch regex> ... </LocationMatch regex> es exactamente igual que el contenedor <Location>; de todas formas utiliza una expresión regular como argumento en vez de un URI.
- <Limit ...> ... </Limit> se utiliza para aplicar una o más directrices especializadas en el control de acceso a ciertas áreas del sitio Web o a un método en particular. Este contenedor es el que menos alcance tiene.

Nota: URI (Uniform Resource Identifier) es el término genérico que se utiliza con los identificadores de recursos uniformes, a los que pertenece URL. Los otros son URN (Uniform Resource Names), URC (Uniform

Resource Characteristics) y LIFN (Localitation Independent File Names). De todas forma, el único que se utiliza con cierta normalidad es URL.

Vamos a ver con un ejemplo del alcance de este contenedor. A continuación le mostramos un extracto del archivo httpd.conf:

```
<VirtualHost 206.171.50.50>

ServerName www.nitec.com
DocumentRoot /www/nitec/public/htdocs
DirectoryIndex welcome.html

<Location /secured/>
      DirectoryIndex login.html
</Location>

</VirtualHost>
```

En este ejemplo, se ha definido un servidor virtual llamado www.nitec.com por medio del contenedor <VirtualHost>. Las tres directrices, ServerName, DocumentRoot y DirectoryIndex se encuentran dentro del contexto del servidor virtual, por lo que se aplicarán a todo lo que éste contenga. La directriz DirectoryIndex especifica que si una petición solicita acceder a uno de los directorios del servidor para visualizar el archivo welcome.html, se le tiene que permitir.

De todas formas, el contenedor Location especifica que un archivo diferente, login.html, se le mostrará a todo el que trate de acceder a www.nitec.com/secured/. Las directrices que se encuentren en un contenedor con un alcance más limitado siempre tendrán preferencia sobre otras directrices más generales.

A la hora de utilizar contenedores conviene que tenga presente unas cuantas reglas gracias a las cuales definirá el comportamiento de una sección del espacio de su sitio Web:

- Un contenedor <VirtualHost> no se puede anidar con otro del mismo tipo.
- No puede haber contenedores dentro de <Limit>, cuyo alcance es el más limitado de todos.
- Los contenedores <Location> y <Directory> no se pueden mezclar ni utilizar uno dentro del otro.

Contexto por directorio

Puede incluir directrices dentro de los archivos de configuración por directorio. Uno de estos archivos (el nombre predeterminado es .htaccess) no es otra cosa que un fichero de texto que contiene una o varias directrices que únicamente se aplicarán en el directorio al que hagan referencia. Dichas directrices se pueden encontrar dentro de contenedores como <Files ...> o bien <Limit ...>. Por medio de este archivo puede controlar el comportamiento de Apache cuando recibe una petición dirigida a un directorio o archivo determinado. Si la directriz se encuentra en el contexto adecuado, aparecerá dentro del archivo de configuración por directorio.

Obsérvese que se puede desactivar todo o parte de lo que se puede sobreescribir en un archivo de configuración por directorio. Por lo tanto, es posible que no se procese ninguna de las directrices que se encuentran en este contexto, dependiendo de si se ha activado la opción de la sobreescritura.

Directrices de configuración general

Estas directrices se aplican tanto al servidor principal (contexto de configuración del servidor) como a los servidores virtuales (contexto de servidor virtual).

AccessConfig

```
Sintaxis: AccessConfig nombre_archivo
Predeterminado: AccessConfig conf/access.conf
Contexto: configuración servidor, servidor virtual
```

Como puede recordar del capítulo 3, durante el proceso de inicio de un servidor se leen varios archivos de configuración. Uno de estos archivos es el de configuración global de acceso y su nombre access.conf. Se encuentra en el directorio conf predeterminado del servidor, especificado en la directriz ServerRoot.

Por medio de la directriz AccessConfig se le puede instruir a Apache para que lea un archivo de configuración distinto a conf/access.conf. Si quiere cargar la información global de acceso a partir de un archivo que se llame conf/globalaccess.conf, tendrá que usar la siguiente sentencia en el fichero httpd.conf:

```
AccessConf conf/globalaccess.conf
```

Nota: el nombre del fichero está expresado en relación al path determinado en ServerRoot.

Truco: si desea trabajar con un archivo de configuración distinto para cada servidor Apache que ejecute, puede desactivar la carga del fichero de configuración conf/access.conf escribiendo lo siguiente:

```
AccessConfig /dev/null
```

También tendrá que desactivar la carga del archivo conf/srm.conf utilizando la directriz ResourceConfig de forma parecida.

Hasta hace poco el archivo access.conf solamente contenía secciones <Directory>: ahora puede tener cualquier directriz destinada a configurar el servidor ante cualquier contexto. También es válido con los contextos de los servidores virtuales. Sin embargo, el único caso en el que se puede utilizar AccessConfig será cuando se encuentre dentro de la sección <VirtualHost>, algo que puede resultar especialmente útil sobre todo cuando se tiene que trabajar con varios servidores a la vez. Gracias a esta directriz podrá cargar por separado los archivos de configuración de un servidor virtual. De esta forma conseguirá incrementar la legibilidad del archivo principal de configuración del servidor.

Obsérvese que el archivo que se especifica en la directriz AccessConfig se lee después del indicado en ResourceConfig.

AccessFileName

```
Sintaxis: AccessFileName nombre_archivo
Predeterminado: AccessFileName .htaccess
Contexto: configuración servidor, servidor virtual
Compatibilidad: se puede utilizar más de un nombre de archivo en
Apache 1.3 y posteriores.
```

La directriz especifica el nombre del archivo de control por directorio. Los parámetros predeterminados (.htaccess) hacen que Apache busque el archivo .htaccess cada vez que el cliente del sistema emite una petición de acceso. Por ejemplo, la directriz DocumentRoot de un sitio Web llamado www.miempresa.com, asistida por un servidor Apache sería:

```
DocumentRoot /www/miempresa/public/htdocs
```

y un servidor Web solicitaría el URL siguiente:

```
www.miempresa.com/ feedback.html
```

Esto hace que el servidor Apache busque los siguientes archivos de control de acceso:

- /.htaccess
- /www/.htaccess
- /www/miempresa/.htaccess
- /www/miempresa/public/.htaccess
- /www/miempresa/public/htdocs/.htaccess

Una vez que los ha revisado, busca el archivo feedback.html. Si no quiere que se use el archivo de control de acceso por directorio y le gustaría que Apache dejase de buscarlo, todo lo que tiene que hacer es utilizar la directriz <Directory> para desactivar los privilegios:

```
<Directory />
AllowOverride None
</Directory>
```

BindAddress

```
Sintaxis: BindAddress dirección IP
Predeterminado: BindAddress *
Contexto: configuración servidor
```

Un Apache que se ejecute en un sistema Unix puede escuchar las conexiones que tengan lugar a través de más de una dirección IP. De esta forma puede especificar una dirección IP para que Apache trabaje con ella. El valor predeterminado es * (comodín) que indicará que Apache escuchará todas las direcciones IP con las que trabaja la máquina. Por ejemplo:

```
BindAddress 206.171.50.50
```

hará que Apache únicamente escuche la dirección IP definida. Sólo se puede utilizar una directriz BindAddress. Si desea trabajar con más de una dirección tendrá entonces que usar la directriz Listen en vez de BindAddress. También se puede utilizar con varios servidores virtuales para que se comporten como servidores independientes. De esta forma sustituirá las secciones <VirtualHost>.

CoreDumpDirectory

```
Sintaxis: CoreDumpDirectory directorio
Predeterminado: la misma ubicación que ServerRoot
Contexto: configuración servidor
```

Si puede explorar los archivos que se graban cada vez que se produce un fallo en el sistema y quiere que Apache aproveche esta capacidad, tendrá que utilizar esta directriz para definir la ubicación de dicho fichero. Por defecto se encuentra en el directorio ServerRoot; sin embargo, como el usuario que lo ejecuta no tiene permiso para escribir en él, este valor predeterminado dejará de tener validez.

DocumentRoot

```
Sintaxis: DocumentRoot directorio
Predeterminado: DocumentRoot /usr/local/apache/htdocs
Contexto: configuración servidor, servidor virtual
```

El directorio especificado en esta directriz se convierte en el directorio del nivel más alto de todos los documentos de Apache.

Por ejemplo, si

```
DocumentRoot /www/miempresa/public/htdocs
```

se apunta al servidor www.miempresa.com, entonces cualquier solicitud de acceso recibida para www.miempresa.com/corporate.html hace que el servidor busque:

```
/www/miempresa/public/htdocs/info/corporate.html
```

Advertencia: si se produce un problema en uno de los módulos del núcleo (mod_dir, mod_log_config) cuando DocumentRoot [se ejecuta] con una barra invertida (por ejemplo DocumentRoot /usr/web/) falla. Por eso se ha de evitar el uso de este carácter al final del path.

Obsérvese que se puede hacer que un servidor busque archivos que se encuentran fuera del directorio DocumentRoot. En otras palabras, si quiere acceder a ficheros que se encuentren fuera del directorio DocumentRoot, tendrá que utilizar la directriz Alias para crear un nombre virtual que apunte a cualquier directorio físico del servidor.

ErrorDocument

```
Sintaxis: ErrorDocument error-code nombre_archivo | mensaje error | URL
Contexto: configuración servidor, servidor virtual, directorio,
por-directorio
Override: FileInfo
Compatibilidad: A partir de la versión 1.1 de Apache se dispone de
los contextos directorio y por directorio (.htaccess)
```

Cuando un servidor se encuentra un problema, genera un mensaje estándar de error incluyendo el código correspondiente. De todas formas este sistema no le gusta a demasiada gente por eso se pueden personalizar este tipo de mensajes. Para eso mismo se utiliza la directriz ErrorDocument.

ErrorDocument tiene dos argumentos. El primero es el código estándar de error HTTP (que encontrará al final del libro) y el segundo es la acción que se desencadenará con dicho error. Dependiendo de sus necesidades puede hacer que el servidor actúe de una forma o de otra.

Por ejemplo, si desea mostrar un mensaje de error que diga "fichero no encontrado" a todos aquellos que le soliciten un archivo en particular, tendrá que averiguar el código de dicho error y utilizarlo con la directriz ErrorDocument. Como este código es el 404, utilizará la siguiente sentencia para hacer que Apache le muestre este mensaje de error:

```
ErrorDocument 404 "Lo siento, petición no válida por %s "
```

Observe que el mensaje aparece entre comillas y que el servidor sustituye %s por cualquier información relacionada con el error que tenga en su poder. De todas formas, si encuentra esta directriz algo limitada para sus pretensiones, puede utilizar un fichero como mensaje de error. Por ejemplo:

```
ErrorDocument 404 /errors/404.html
```

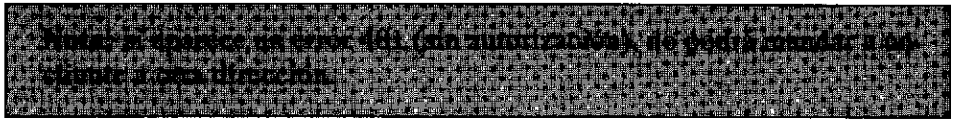
Cada vez que no se encuentre un archivo se dará este error y se le mostrará al cliente el contenido del archivo 404.html (a través del explorador Web). Si quiere hacer algo más que limitarse a mostrar una página estática, puede utilizar un script CGI para aplicar una acción en particular. En este caso utilice el nombre del archivo en el que se encuentre en script:

```
ErrorDocument 404 /cgi-bin/fallourl.cgi
```

De esta forma, cada vez que tenga lugar un fallo 404 se llamará al script fallourl.cgi. También se puede redirigir el cliente a otro URL utilizando éste en vez del nombre del archivo:

```
ErrorDocument 404 http://www.newsite.com/otra.dirección.html
```

Se puede utilizar en los casos en que se haya cambiado la dirección de un sitio Web.



Include

```
Sintaxis: Include nombre_archivo
Contexto: configuración servidor
Compatibilidad: A partir de la versión 1.3 de Apache
```

Si desea incluir otros archivos de configuración a parte de los propios del servidor, utilice esta directriz.

Listen

```
Sintaxis: Listen [dirección IP] número del puerto
Contexto: configuración servidor
```

Por defecto, Apache responde a las solicitudes procedentes de las direcciones IP asociadas al servidor, pero sólo a las especificadas en la directriz Port. Para que la situación sea algo más configurable hay que utilizar la directriz Listen. Con ella se le indica al servidor que responda a ciertas direcciones IP, a una combinación de direcciones y puertos o simplemente a un puerto.

Aunque Listen se puede utilizar para sustituir BindAddress y Port, convendrá utilizar esta última con aquellos servidores que generen URL que apunten a sí mismos.

Se pueden utilizar varias directrices Listen para especificar un número de direcciones y puertos a los que se quiere prestar atención. Por ejemplo, para que acepte las conexiones de los puertos 80 y 8080 utilice:

```
Listen 80
Listen 8080
```

Con los ejemplos siguientes Apache aceptará las conexiones procedentes de dos direcciones IP y dos puertos:

```
Listen 11.22.3311.22.33.1:80
Listen 11.22.3311.22.33.2:8080
```

Port

```
Sintaxis: Port número
Predeterminado: Port 80
Contexto: configuración servidor
```

Esta directriz asigna un número de puerto comprendido entre 0 y 65535 al servidor. Si no se utiliza ni Listen ni BindAddress para especificar ningún número de puerto, Port indica el puerto de la red al que hay que prestar atención. Pero si Listen o BindAddress sí que se utilizan para especificar un número de puerto, la directriz Port dejará de tener efecto. Port utiliza la variable de entorno SERVER_PRTO (para CGI y SSI) y la utiliza siempre que el servidor tiene que generar un URL que se refiera a sí mismo.

Aunque se pueda especificar cualquier número de puerto comprendido entre 0 y 65535, conviene que recuerde que hay una limitación. Todos los números que se encuentren por debajo de 1024 están reservados para los servicios estándar como TELNET, SMTP, POP3 HTTP y FTP. Encontrará la asignación de estos números dentro del archivo /etc/services. O, si quiere curarse en salud, utilice cualquier número de puerto que no sea 80 (por ejemplo direcciones superiores a 8000).

Tenga en cuenta que si no es el usuario root pero quiere ejecutar Apache para escuchar en alguna ranura de E/S, necesitará configurar los puertos superiores a 1023 (porque sólo los usuarios root pueden utilizar los servicios estándar).

Notar: el contenedor <VirtualHost> también se puede utilizar para configurar el puerto que se utilizará con el servidor virtual.

User

```
Sintaxis: User unix-userid
Predeterminado: User #-1
Contexto: configuración servidor, servidor virtual
```

Esta directriz se utiliza para configurar el ID del servidor que se utilizará con los subprocesos que creará el servidor Apache para responder a las peticiones recibidas. Se ha de configurar Apache para que se ejecute como un servidor independiente (véase la directriz ServerType) y para que se encargue de abrir dichos subprocesos. Una vez que se inicia el servidor, no se

ejecutará como root. El proceso Apache asociado (al que se le llama demonio) cambia el ID de los subprocesos del usuario independientemente de lo que se haya definido en la directriz User (y siempre que el ID del usuario siga teniendo validez).

Advertencia: si inicia un servidor como un usuario que no tiene los permisos root, no podrá cambiar el ID del usuario que se ha especificado en la directriz User (seguirá utilizando la identificación original). Si inicia el servidor como root, es normal que el subproceso de Apache se ejecute como root; de todas formas los subprocesos del usuario se ejecutarán con la identificación que se especifique en la directriz User.

Puede emplear los números ID de los usuarios. Los encontrará en el archivo /etc/password. Si va a trabajar con valores numéricos en vez de utilizar los nombres de los usuarios recuerde que tendrá que colocar el carácter # antes del número.

Muchos administradores Apache utilizan la opción del usuario predeterminado. Esta opción no se puede utilizar en todos los sistemas Unix y no siempre es aconsejable. Mi consejo es que se utilice una única identificación para usuarios y grupos (consulte la directriz Group) del servidor. El número ID que decida utilizar para los subprocesos debería tener muy pocos permisos. No es conveniente que se pueda acceder a los archivos que se quiere mantener apartados de los ojos del resto del mundo. Del mismo modo, tampoco se debe permitir que se ejecuten aplicaciones que no tengan nada que ver con las peticiones HTTP.

Para utilizar esta directriz dentro del contenedor <VirtualHost> habrá que configurar correctamente el wrapper suEXEC. Cuando se utilice el wrapper en vez del contenedor <VirtualHost> sólo afectará al CGI del usuario que se esté ejecutando. Las peticiones que no sean CGI se procesarán con el usuario especificado en la directriz User. Como se puede ver, la directriz User no es completamente suprimible.

Por último, nunca configure la directriz User (o bien Group) como root a menos que sepa perfectamente lo que está haciendo y los peligros que ello entraña.

Group

```
Sintaxis: Group unix-group
Predeterminado: Group #-1
Contexto: configuración servidor, servidor virtual
```

La directriz Group se puede utilizar junto con User. Determina el grupo bajo el que se atenderán las peticiones que reciba el servidor independiente. Para utilizarla, dicho servidor se tendrá que ejecutar como root. A la directriz Group también se le puede asignar un número. Si desea conocer la relación existente entre los nombres de los grupos y sus números, consulte el archivo /etc/group.

Notar todas las recomendaciones y avisos que le indiqué para la directriz User, son aplicables para Group.

<IfModule>

```
Sintaxis: <IfModule [!]nombre_módule> ... </IfModule>
Predeterminado: ninguno
Contexto: todos
Compatibilidad: sólo a partir de la versión 1.2 (inclusive)
```

Utilícela si utiliza directrices que se encuentren dentro de un módulo personalizable que puede que no esté presente en la instalación de Apache. Por ejemplo, si quiere usar ciertas directrices sólo si aparece un módulo, entonces tendrá que utilizar las siguientes sentencias:

```
<IfModule nombre-módulo>
# asígnele a las siguientes directrices sus respectivos valores
# las directrices aparecerían aquí
</IfModule>
```

Por otro lado, si se encuentra en una situación que sea exactamente la contraria de la aquí expuesta, todo lo que tendrá que hacer es colocar un símbolo ! antes del nombre del módulo. Obsérvese que el argumento nombre_módulo es el nombre del archivo que tenía el módulo en el momento de la compilación (por ejemplo, mod_rewrite.c). Las secciones de <IfModule> son *netstable* (método que se puede usar para implementar varias pruebas varios módulos).

Options

```
Sintaxis: <Options [+|-]opción [+ -]opción ...
Contexto: configuración servidor, servidor virtual, directorio,
.htaccess
Override: Options
```

Esta directriz controla las propiedades del servidor disponibles para un directorio en particular. Cuando su valor es None (ninguno), no se utilizará

ninguna propiedad extra dentro del contexto en el que entra en funcionamiento la directriz. Los valores posibles son los siguientes:

- None: sin opciones.
- All: todas a excepción de MultiViews.
- ExecCGI: se permite la ejecución de CGI.
- FollowSymLinks: el servidor seguirá los vínculos simbólicos que apunten a los directorios del sistema. Sin embargo, no cambiará el nombre del path utilizado en las secciones <Directory>.
- Includes: se pueden utilizar los comandos SSI (Server Side Include).
- IncludesNOEXEC: se puede asignar una restricción de comandos SSI a las páginas de este tipo. Dichos comandos son #include y #exec.
- Indexes: si se solicita la URL de un directorio y éste carece de DirectoryIndex, es decir, de un directorio, entonces el servidor muestra una lista con el contenido de dicha carpeta.
- SymLinksIfOwnerMatch: el servidor sólo seguirá los enlaces que pertenezcan al usuario que emite la solicitud.
- MultiViews: permite negociar el contenido basándose en el lenguaje de un documento.

Utilice los signos + y - para activar o desactivar las distintas opciones. Vamos a ver un ejemplo para tratar de aclarar el concepto. La configuración que aparece a continuación muestra dos contenedores de directorio que se encuentran en un mismo archivo, como por ejemplo access.conf:

```
<Directory /www/micliente/public/htdocs >
Options Indexes MultiViews
</Directory>

<Directory /www/micliente/public/htdocs >
Options Includes
</Directory>
```

Al directorio /www/micliente/public/htdocs/ tan solo se le han asignado la opción Include. De todas formas, si la segunda sección utilizase los signos + y - como sigue:

```
<Directory /www/micliente/public/htdocs >
Options +Includes -Indexes
</Directory>
```

entonces se utilizarían las opciones MultiViews e Includes.

Cuando se aplican varias opciones hay que tener cuidado con que el contexto más restrictivo tenga preferencia sobre el más general. Por ejemplo:

```
ServerName
Options ExecCGI Includes
<VirtualHost 11.22.3311.22.33.1>
ServerName www.micliente.com
Options -ExecCGI -Includes
<Directory /www/micliente/public/htdocs/ssi >
      Options Includes
</Directory>
</VirtualHost>
```

Por ejemplo, el servidor principal permite que la ejecución de los script CGI y de los comandos SSI al activar las opciones ExecCGI e Includes. Sin embargo, el servidor virtual las desactiva al utilizar los argumentos anteriores con el signo -. Por último, el servidor virtual tiene una opción específica para el directorio /www/micliente/public/htdocs/ssi que permite la ejecución de comandos SSI. Observe que Includes es la única opción que se utiliza en el directorio /www/micliente/public/htdocs/ssi.

Como puede ver, la directriz Options utiliza los signos + y - seguidos de los valores que se tienen que agregar o eliminar de la lista de opciones. Por otro lado, si dicha directriz no usa los signos + y -, entonces los valores especificados sobreescribirán a los que aparezcan en las directrices Options anteriores.

ResourceConfig

```
Sintaxis: ResourceConfig nombre_archivo
Predeterminado: ResourceConfig conf/srm.conf
Contexto: configuración servidor, servidor virtual
```

Esta directriz es igual que AccessConfig. Si desea desactivar la carga predeterminada del archivo conf/srm.conf, añada entonces esta línea en el fichero httpd.conf:

```
ResourceConfig /dev/null
```

ServerAdmin

```
Sintaxis: ServerAdmin email
Contexto: configuración servidor, servidor virtual
```

Esta directriz muestra la dirección de correo electrónico con los mensajes de error que genera el servidor. Si tiene que trabajar con cierta cantidad de

sitios Web virtuales, es posible que desee utilizar distintas direcciones de correo electrónico para cada servidor virtual, de tal forma que sepa en todo momento con qué servidor está relacionado el problema.

Truco: para que sus sitios Web tengan ese aspecto profesional, no use direcciones de correo electrónico que no incluyan el sitio Web virtual como parte de la dirección. Por ejemplo, si su empresa es un proveedor de Internet (ISP) que se llama miempresa.net y tiene un cliente que se llame www.micliente.com, entonces configure el ServerAdmin del sitio www.micliente.com a una dirección del estilo usuario@micliente.com, tal como master@micliente.com en vez de utilizar la general master@miempresa.net. De esta forma, cuando el servidor muestre un mensaje de error a alguien que esté visitando www.micliente.com, el usuario verá una dirección que se corresponda con el sitio por el que está navegando. Se considera que este formato es más profesional.

ServerName

```
Sintaxis: ServerName nombre de dominio válido
Contexto: configuración servidor, servidor virtual
```

La directriz ServerName es la encargada de definir el nombre del servidor. Cuando no se utiliza esta directriz, Apache trata de averiguar el nombre por medio de una petición DNS que lanza durante el proceso de inicio. De todas formas, dependiendo de la configuración DNS que se tenga, puede llegar a desaconsejarse esta práctica puesto que el servidor puede escoger un nombre poco apropiado. Por lo tanto, lo mejor es definir el nombre que se quiera utilizar.

Asegúrese de entrar un nombre completo de dominio en vez de un simple atajo. Por ejemplo, si desea que el nombre de su servidor sea wormhole.miempresa.com, no debería configurar la directriz SeverName para que utilice wormhole. Lo correcto sería:

```
ServerName wormhole.miempresa.com
```

ServerRoot

```
Sintaxis: ServerRoot directorio-nombre_archivo
Predeterminado: ServerRoot /usr/local/apache
Contexto: configuración servidor
```

La directriz ServerRoot se utiliza para definir el directorio en el que se encuentran los archivos del servidor. No la confunda con la directriz DocumentRoot que se utiliza para indicarle al servidor donde se encuentran los documentos Web. ServerRoot se utiliza para localizar los archivos de configuración y registro. En la distribución estándar, en el directorio ServerRoot, encontrará los archivos conf, srm y los subdirectorios de registro. De todas formas se puede utilizar el parámetro -d en la línea comandos para indicarle a Apache cuál es el directorio ServerRoot.

Obsérvese que AccessConfig y ResourceConfig utilizan la directriz ServerRoot para conocer la ubicación de los archivos de configuración. Por ejemplo, si la configuración de las directrices es:

```
ServerRoot /usr/local/httpd
AccessConfig conf/access.conf
ResourceConfig conf/srm.conf
```

entonces AccessConfig y ResourceConfig le indicarán al servidor que cargue los archivos /usr/local/httpd/conf/access.conf y /usr/local/httpd/conf/srm.conf respectivamente.

DefaultType

```
Sintaxis: DefaultType mime-type
Predeterminado: DefaultType text/html
Contexto: configuración servidor, servidor virtual, directorio,
.htaccess
Override: FileInfo
```

Esta directriz se utiliza para establecer el tipo de contenido predeterminado. De esta forma, cuando Apache recibe una solicitud de un documento cuyo tipo se desconozca (en otras palabras, que el mapa MIME del servidor no puede precisar de qué se trata), utiliza el tipo predefinido.

Por ejemplo, si se tiene un directorio en el que se guarden todos los archivos de texto que no tengan extensión, se puede utilizar la directriz DefaultType dentro de un contenedor <Directory> que señale a dicha carpeta. En este caso, la configuración de DefaultType a text/plain permitiría que el servidor le indicase a la otra parte (el explorador Web) que se tratan de archivos en texto plano.

Aquí tiene un ejemplo:

```
<Directory /www/miempresa/public/htdocs/plaindata>
DefaultType plain/text
</Directory>
```

En este ejemplo, todos los archivos que se encuentran dentro del directorio www/miempresa/public/htdocs/plaindata/ se tratan como si fuesen de texto plano.

Directrices para la configuración de recursos y eficacia

Estas directrices le permiten afinar el servidor Apache para obtener un rendimiento mayor y optimizar el control. Puede hacerlo de varias formas. Observe que con la mayoría de estas directrices hay que tener muy claro cuál es el funcionamiento del sistema, en términos de sistema operativo, hardware, etc. Además, siempre es conveniente que le eche un vistazo a los manuales de su sistema operativo o a cualquier ayuda que le indique cuáles son los límites de los recursos del sistema, cómo controlar las conexiones TCP/IP, etc.

Las directrices de esta sección se agrupan en subfunciones.

Control de los procesos de Apache

Las directrices que aparecen a continuación se utilizan para controlar la forma en que se ejecuta Apache dentro del sistema operativo. Además, utilizándolas podrá controlar también la forma en que se utilizan los recursos del sistema. Por ejemplo, puede decidir cuántos subprocesos se ejecutarán en el sistema, o la cantidad de threads que utilizará Apache cuando trabaje con Windows.

A la hora de configurar estas directrices hay que tener presente los siguientes puntos:

- Cuantos más procesos ejecute, mayor carga tendrá la CPU.
- Cuantos más procesos ejecute, mayor cantidad de memoria RAM necesitará.
- Cuantos más procesos ejecute, más cantidad de recursos del sistema consumirá (como por ejemplo descriptores de archivos, búferes compartidos, etc.).

Obviamente, cuantos más procesos se ejecuten, tantas más peticiones podrá atender y mayor éxito tendrá su sitio Web. De todas formas, la configuración de estas directrices se basa en una combinación de experimentación, requisitos y recursos disponibles.

ServerType

```
Sintaxis: ServerType inetd | standalone
Predeterminado: ServerType standalone
Contexto: configuración servidor
```

La directriz ServerType le indica al sistema Unix cómo se ha de ejecutar el servidor Apache. Se pueden utilizar dos valores: inetd y standalone.

Utilice el primero para ejecutar el servidor como un proceso del sistema llamado -inetd. En este sistema, se inicia una nueva copia del servidor Apache para cada solicitud HTTP recibida. Así se aumenta la cantidad de trabajo del que se tiene que hacer cargo el sistema para atender una solicitud y el rendimiento obtenido no es que sea muy grande. De todas formas, inetd se lleva utilizando bastante tiempo y dispone de varias opciones de seguridad (como los wrappers TCP). Son muchos los administradores de servidores de Internet que prefieren trabajar con este tipo de servicios. Para los sitios Web que no tengan demasiada densidad de tráfico, la diferencia es mínima.

Nota: ejecutar Apache como un servidor inetd no hace que automáticamente sea más seguro que ejecutarlo en modo standalone. Lo único que ocurre es que se lleva utilizando bastante tiempo, hay bastantes administradores que piensan que es menos propenso a ataques.

Utilice el valor standalone cuando quiera ejecutar un servidor independiente en el que Apache se ejecute como un demonio. En otras palabras, un servidor Apache principal escucha las solicitudes de conexión que llegan a través de una serie de puertos y abre los subprocesos Apache para atender a tales peticiones; nunca atiende él mismo a una petición HTTP. Los subprocesos se ejecutan bajo los permisos correspondientes al ID especificado en las directrices User y Group. Para aprender más sobre estas configuraciones conviene que consulte estos parámetros. Se recomienda su uso con los sitios Web cuya densidad de tráfico sea elevada.

StartServers

```
Sintaxis: ServerType número
Predeterminado: ServerType 5
Contexto: configuración servidor
```

La directriz ServerType sólo se utiliza cuando el servidor Apache se ejecuta en modo independiente, es decir, standalone. En otras palabras, para que esta directriz sea válida tendrá que configurar la directriz ServerType como standalone.

Esta directriz determina el número de subprocesos que podrá crear el servidor Apache en el inicio. Obsérvese que el número de subprocesos que se va a utilizar durante cierta cantidad de tiempo se controla dinámicamente. El servidor Apache principal (el demonio) abre nuevos subprocesos según va haciendo falta. El número actual de procesos se controla a través de las directrices MinSpareServers, MaxSpareServers y MaxClients. Como se puede ver tampoco se gana mucho modificando el valor de este parámetro.

Cuando se ejecuta el servidor con Microsoft Windows, esta directriz determinará el número total de subprocesos en ejecución. Como la versión de Apache para Windows es multitarea, un proceso controla todas las solicitudes. El resto de procesos se reservan para cuando se apague el proceso principal.

ThreadsPerChild

```
Sintaxis: ThreadsPerChild número
Predeterminado: ThreadsPerChild 50
Contexto: configuración servidor (Windows)
Compatibilidad: sólo con las versiones Apache 1.3 y posteriores con
Windows
```

La versión de Apache para Windows 95/NT es un servidor multitarea. La directriz ThreadsPerChild le indica la cantidad de threads que puede utilizar. También se utiliza para determinar el número máxima de conexiones que admitirá. Por lo tanto, conviene que este valor sea razonablemente alto para que se pueda atender la mayor cantidad de solicitudes posibles.

SendBufferSize

```
Sintaxis: SendBufferSize bytes
Contexto: configuración servidor
```

Esta directriz ajusta el tamaño del búfer de envío TCP al número de bytes determinado. En una red de alto rendimiento, al utilizar un número de bytes elevado, se puede mejorar el rendimiento del servidor.

ListenBacklog

```
Sintaxis: ListenBacklog trabajos_pendientes
Predeterminado: ListenBacklog 511
Contexto: configuración servidor
Compatibilidad: sólo disponible en las versiones de Apache
posteriores a 1.2.0.
```

Esta directriz le permite defenderse ante acciones que atenten contra la seguridad del sistema, como pueden ser los ataques de negación de servicio (DOS). Para ello se puede determinar la cantidad de trabajos pendientes que

esperarán a ser realizados. Si se encuentra que están bajo los efectos de un ataque DOS, tendrá que incrementar el valor del argumento de esta directriz. En cualquier otro caso, déjelo como está.

TimeOut

```
Sintaxis: TimeOut número
Predeterminado: TimeOut 300
Contexto: configuración servidor
```

Como ya sabe, la Red es un sistema cliente-servidor en el que Apache se encarga de responder a las peticiones. Las peticiones (o solicitudes) y sus respuestas se transmiten en forma de paquetes de datos. Apache tiene que saber qué cantidad de tiempo ha de esperar la recepción de un paquete. Y esta directiva se encarga de determinar el tiempo de espera, midiéndolo en segundos. Con el valor del parámetro de esta directriz se define la cantidad de segundos que esperará Apache antes de cortar la conexión. La configuración predeterminada permite un tiempo de espera de 300 segundos. Si trabaja en una red lenta, es posible que desee aumentar dicho tiempo de espera.

El tiempo de espera se aplica a:

- Tiempo transcurrido hasta que se recibe la petición GET.
- Tiempo transcurrido entre la recepción de los paquetes TCP correspondientes con una solicitud POST o PUT.
- Tiempo transcurrido entre la confirmación de la transmisión de los paquetes TCP de respuesta.

MaxClients

```
Sintaxis: MaxClients número
Predeterminado: MaxClients 256
Contexto: configuración servidor
```

Esta directriz limita el número de peticiones simultáneas de las que Apache se puede hacer cargo. Como Apache utiliza un subservidor para cada petición, el número de éstos que puede llegar a utilizar a la vez influirá en el rendimiento final del sistema. El valor predeterminado es el que aparece en el archivo httpd.h de la distribución estándar de Apache. Debería bastar para la mayoría de los sitios, pero la verdad es que lo considero limitado por las dos razones siguientes: los diseñadores de Apache no querían que el servidor colgase el sistema rellenando alguna tabla del núcleo y, con este número, se obtiene una tabla de resultados lo suficientemente pequeña como para que se pueda leer.

Si su sistema cuenta con un servidor de gran efectividad y dispone del ancho de banda suficiente, puede modificar este límite, que se encuentra en el archivo httpd.h y compilar el servidor de nuevo.

Observe los siguientes estamentos. Cambie el número 256 por otro más grande:

```
#ifndef HARD_SERVER_LIMIT
#define HARD_SERVER_LIMIT 256
#endif
```

Entonces, ¿qué pasa cuando el número de solicitudes alcanza este límite? Bueno, pues que todas las que lleguen pasarán a un estado de espera hasta que el servidor pueda atenderlas.

MaxRequestPerChild

```
Sintaxis: MaxRequestPerChild número
Predeterminado: MaxRequestPerChild 0
Contexto: configuración servidor
```

Apache abre un subproceso servidor para atender una petición. De todas formas, un subservidor se puede hacer cargo de muchas peticiones. Dicho número de peticiones lo define la directriz MaxRequestPerChild. Después de atender el número de solicitudes determinado por esta directriz, el subproceso se termina.

Si el valor de MaxRequestPerChild es 0, entonces el proceso no expirará nunca. Si cree que hay librerías en su sistema operativo (como Solaris) que tienen código de memoria, es posible que desee trabajar con el valor 0. También puede definir un ciclo vital para los subprocesos y reducir de esta forma el consumo de memoria. Así reducirá la carga de su sistema y el servidor Apache dejará de estar tan ocupado.

MaxSpareServers

```
Sintaxis: MaxSpareServers número
Predeterminado: MaxSpareServers 10
Contexto: configuración servidor
```

Esta directiva le permite ajustar el número de subprocesos Apache parados que desea que haya en su servidor. Si el número de procesos detenidos supera al valor de la directriz MaxSpareServers, entonces un proceso principal se encargará de eliminar los excedentes. Unicamente habrá que retocar este valor cuando se trabaje con sitios Web con mucho tráfico. A menos que sepa lo que está haciendo, mi consejo es que no modifique este valor.

MinSpareServers

```
Sintaxis: MinSpareServers número
Predeterminado: MinSpareServers 5
Contexto: configuración servidor
```

La directriz MinSpareServers determina el número mínimo de procesos en espera que habrá en el servidor. Un proceso en espera es aquel que no se está haciendo cargo de ninguna solicitud. Si el número de procesos en espera fuese inferior que el determinado en esta directriz, un proceso principal se encargaría de crear nuevos procesos a una velocidad de 1 por segundo. Unicamente habrá que retocar este valor cuando se trabaje con sitios Web con mucho tráfico. A menos que sepa lo que está haciendo, mi consejo es que no modifique este valor.

Establecer conexiones persistentes

Por medio de las directrices KeepAlive que se verán en esta sección puede hacer que su servidor Apache utilice conexiones persistentes de tal modo que una misma conexión TCP se pueda usar para efectuar varias transacciones. Normalmente, cada petición HTTP y su respuesta utiliza una conexión distinta. Es decir, que cada vez que el servidor recibe una solicitud, abre una conexión para recibirla y luego la cierra. Cuando genera la respuesta, abre una conexión TCP, la envía y luego la cierra. Al utilizar una misma conexión para efectuar varias transacciones se reduce el trabajo de abrir y cerrar conexiones TCP constantemente, con lo que se mejora el rendimiento del sistema.

De todas formas, para establecer una conexión persistente, tanto el servidor como el cliente tendrán que ser capaces de trabajar con ellas. Los exploradores Web más populares, como Netscape Navigator y Microsoft Internet Explorer disponen de propiedades KeepAlive.

Tenga en cuenta que no todas las transacciones puedan aprovecharse de las ventajas que ofrecen las conexiones persistentes. Uno de los requisitos de este tipo de conexiones es que se ha de conocer de antemano el tamaño de los recursos transmitidos. Por esta misma razón habrá muchos script CGI, comandos SSI y transmisiones que no podrán beneficiarse de este tipo de conexiones.

KeepAlive

```
Sintaxis: (Apache 1.2) KeepAlive On | Off
Predeterminado: (Apache 1.2) KeepAlive On
Contexto: configuración servidor
Compatibilidad: sólo disponible para Apache 1.1 y posteriores
```

Esta directriz le permite activar y desactivar el uso de las conexiones TCP persistentes.

Nota: es posible que con los servidores Apache antiguos (anteriores a la versión 1.2) haya que utilizar un valor numérico en vez de On/Off. Este valor se corresponde con el número máximo de solicitudes que atenderá Apache por petición. Con un límite es imposible evitar que un cliente se haga con todos los recursos del servidor. Para desactivar KeepAlive en las primeras versiones de Apache hay que utilizar el valor 0 (cero).

KeepAliveTimeout

```
Sintaxis: KeepAliveTimeout segundos
Predeterminado: KeepAliveTimeout 15
Contexto: configuración servidor
```

Si tiene activada la directriz KeepAlive (on) puede utilizar esta otra para limitar la cantidad de tiempo (medido en segundos) que Apache esperará una petición antes de cerrar la conexión. Una vez que se ha recibido, el valor de esta orden especifica el tiempo de caducidad.

MaxKeepAliveRequests

```
Sintaxis: MaxKeepAliveRequests número
Predeterminado: MaxKeepAliveRequests 100
Contexto: configuración servidor
Compatibilidad: únicamente disponible a partir de la versión 1.2
y posterior
```

La directriz MaxKeepAliveRequests limita el número de peticiones que se pueden efectuar por conexión mientras KeepAlive está activada. Si el valor es 0 (cero), entonces no hay ningún límite. Le recomiendo que para optimizar el rendimiento del servidor, utilice un número elevado.

Control de los recursos del sistema

Apache es muy flexible ya que le permite controlar la cantidad de recursos de sistema (como reloj de la CPU y memoria) que va a consumir. Estas propiedades de control son realmente útiles ya que permiten mejorar el rendimiento y efectividad del servidor Web. Algunos de los ataques que efectúan los piratas informáticos contra los servidores Web se basan en el consumo de recursos del sistema. Consumen todos los recursos e inutilizan el servidor.

Apache dispone de una serie de directrices especialmente diseñadas para combatir este comportamiento. Las vemos a continuación.

RLimitCPU

```
Sintaxis: RLimitCPU n    'max' [ n | 'max']
Predeterminado: Unset; utiliza las opciones predeterminadas del
sistema operativo
Contexto: configuración servidor, servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Esta directriz le permite controlar el uso que hace Apache de la CPU. Tiene dos parámetros. El primero limita la cantidad de recursos que se utilizarán con todos los procesos, mientras que el segundo limita el número máximo de recursos disponibles. Elevar este número indicaría que el servidor se está ejecutando como root o que está en su fase de inicio. El segundo parámetro es opcional. Cada parámetro tiene dos valores:

- n es el número de segundos por proceso.
- max es el número máximo de recursos que permite el sistema operativo.

RLimitMEM

```
Sintaxis: RLimitMEM n    'max' [ n | 'max']
Predeterminado: Unset; utiliza las opciones predeterminadas del
sistema operativo
Contexto: configuración servidor, servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Esta directriz se utiliza para limitar la cantidad de memoria RAM que utilizan los procesos de Apache. Tiene dos parámetros. El primero define un límite para todos los procesos y el segundo uno para los recursos. Elevar este número indicaría que el servidor se está ejecutando como root o que está en su fase de inicio. El segundo parámetro es opcional.

Cada parámetro tiene dos valores:

- n es el número de segundos por proceso.
- max es el número máximo de recursos que permite el sistema operativo.

RLimitPROC

```
Sintaxis: RLimitPROC n    'max' [ n | 'max']
Predeterminado: Unset; utiliza las opciones predeterminadas del
sistema operativo
Contexto: configuración servidor, servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Esta directriz se utiliza para limitar el número de procesos simultáneos que puede haber por usuario. Tiene dos parámetros. El primero define un límite para todos los procesos y el segundo uno para los recursos. Cada parámetro tiene dos valores:

- n es el número de segundos por proceso.
- max es el número máximo de recursos que permite el sistema operativo.

Elevar este número indicaría que el servidor se está ejecutando como root o que está en su fase de inicio.

Advertencia: Si los procesos CGI se ejecutan bajo el mismo ID de usuario que los procesos del servidor, podrá utilizar esta directriz para limitar el número de procesos que puede abrir el servidor. Si el límite impuesto es demasiado bajo se encontrará con el siguiente mensaje:

```
"Can not fork: process" (no se puede abrir proceso)
```

en el archivo de registro del servidor. En este caso tendrá que aumentar dicho límite o dejarlo en su valor predeterminado.

Uso de módulos dinámicos

Apache carga todos los módulos precompilados durante el inicio. De todas formas, dispone de un módulo dinámico de carga y descarga que en ciertas ocasiones puede resultar de utilidad. Cuando usa las siguientes directrices del módulo dinámico, puede modificar la lista de módulos activos sin tener que volver a compilar el servidor.

ClearModuleList

```
Sintaxis: ClearModuleList
Contexto: configuración servidor
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Puede utilizar esta directriz para limpiar la lista de módulos activos con lo que podrá utilizar el módulo dinámico. Para cargar los módulos se usa la directriz AddModule.

AddModule

```
Sintaxis: AddModule módulo módulo ...
Contexto: configuración servidor
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Esta directriz se puede utilizar para activar cualquier módulo precompilado que esté desactivado. Es posible que el servidor tenga módulos listos para usarse pero que estén desactivados. Con esta directriz los activará. El servidor tiene una lista de módulos cargados de antemano. Para limpiarla se usa la directriz vista en la sección anterior, ClearModuleList. A continuación añadirá los módulos que desee utilizar por medio de AddModule.

Directrices de contenedores estándar

En esta sección veremos algunos contenedores estándar que forman parte de la base del servidor Apache. Estos contenedores se utilizan para aplicar un grupo de directrices a un directorio, archivo o dirección determinada.

Nota: el contenedor <VirtualHost> se tratará en una sección aparte dentro de este mismo capítulo.

No puede mezclar aleatoriamente dichos contenedores. Las guías generales de uso son las siguientes:

- Los contenedores <Directory> y <Files> se utilizan para especificar directrices relacionadas con objetos del sistema como archivos y directorios. No se puede utilizar <Directory> en el archivo .htaccess porque el contenido de dicho fichero se aplica únicamente al directorio en el que se encuentra.
- Utilice el contenedor <Location> para marcar objetos URL. No se puede usar dentro del archivo .htaccess.
- Al utilizar una versión regular de una directriz (por ejemplo, <DirectoryMatch>), siga las mismas reglas que para la versión regular. Sólo se usarán las versiones regulares cuando esté completamente seguro de que la expresión es exacta.

Nota: debido a un error encontrado en las primeras versiones de Apache, el control proxy aún se efectúa desde el contenedor <Directory>, independientemente de que <Location> sea el contenedor apropiado para tal fin. Es posible que se corrija en futuras versiones. De todas formas, el único problema que puede llegar a causar es una mayor dificultad a la hora de definir conceptos.

<Directory>

```
Sintaxis: <Directory directorio> ... </Directory>
Contexto: configuración servidor, servidor virtual
```

<Directory> y </Directory> se utilizan para encerrar un grupo de directrices que únicamente se aplicarán al directorio especificado y a todos sus subdirectorios.

Cualquier directriz que se pueda utilizar con un directorio podrá incluirse en este conjunto. El argumento podrá ser el nombre completo de un path. Por ejemplo:

```
<Directory /www/miempresa/public/htdocs/download>
Options +Indexes
</Directory>
```

Aquí, el directorio /www/miempresa/public/htdocs/download se utiliza como un path completo. Con este ejemplo se activa la indexación de dicho directorio. También se pueden utilizar comodines para especificar el path. Por ejemplo:

```
<Directory /www/miempresa/public/htdocs/downloa?>
Options +Indexes
</Directory>
```

Aquí, ? sustituye a cualquier carácter; es más, se marcarán los directorios tales como /www/miempresa/public/htdocs/download y www/miempresa/public/htdocs/downloaD. También se puede utilizar * (asterisco) para sustituir a cualquier secuencia de caracteres que no sean la barra inclinada /. También es posible utilizar expresiones regulares ampliadas, en las que aparezca el símbolo ~. Por ejemplo:

```
<Directory ~ "^/www/w.*/">
```

con ella marcaría cualquier subdirectorio de /www/.

Observe que la expresión regular basada en los contenedores <Directory> no se aplicará hasta que se hayan utilizado todas las expresiones normales (es decir, las que no tienen expresiones regulares) de los contenedores <Directory> y del archivo .htaccess.

En este caso se probarán las expresiones regulares para ver el orden en el que aparecen en el archivo de configuración.

Tenga en cuenta que si especifica más de un contenedor <Directory> para el mismo directorio, únicamente se aplicará el más restrictivo. Por ejemplo:

```
<Directory /www>
AllowOverride None
</Directory>

<Directory ~ "/www/miempresa/public/htdocs/*">
AllowOverride FileInfo
</Directory>
```

De acuerdo con esto, cuando se recibe una petición para el àrchivo /www/miempresa/public/htdocs/algúnarchivo.cvs, Apache desactiva el control de acceso para el directorio /www configurado en el archivo .htaccess y lo activa para /www/miempresa/public/htdocs. También acepta cualquier directriz FileInfo como DefaultType, procedente del archivo /www/miempresa/public/htdocs/.htacces.

<DirectoryMatch>

```
Sintaxis: <DirectoryMatch regex> ... </DirectoryMatch>
Contexto: configuración servidor, servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.3
```

Es exactamente igual que el contenedor <Directory>, con la salvedad de que como argumento toma una expresión regular y no necesita el carácter ~. <DirectoryMatch> y </DirectoryMatch> se utilizan para englobar un grupo de directrices que únicamente se aplicarán al directorio nombrado y a todos sus subdirectorios. Por ejemplo:

```
<DirectoryMatch "^/www/miempresa/public/htdocs/[A-Z]{8}/*">
```

marcaría todos los subdirectorios de /www/miempresa/public/htdocs cuyos nombres tuviesen exactamente ocho letras mayúsculas; por lo tanto, /www/miempresa/public/htdocs/AAAABBBB/ cumpliría con lo expresado en la directriz.

<Files>

```
Sintaxis: <Files nombre_archivo> ... </Files>
Contexto: configuración servidor, servidor virtual, .htaccess
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Para controlar el acceso que se tiene a partir de los nombres de ficheros se utiliza esta directriz. Las secciones <Files> se procesan en el mismo orden en que aparecen en el archivo de configuración, después de que se lean las secciones <Directory> y los archivos .htaccess, pero antes de <Location>. El

argumento debe incluir el nombre de un archivo o un comodín relativo a una cadena de caracteres. Recuerde que ? equivale a un único carácter y * a una cadena a excepción de /. El símbolo ~ le permite ampliar las expresiones regulares del argumento. Por ejemplo:

```
<Files ~ "\.(zip|tar|tgz|arj|zoo)$">
```

marcarían cualquier archivo con la extensión zip, tar, tgz, arj y zoo. A diferencia de las secciones <Directory> y <Location>, las <Files> se pueden utilizar dentro del archivo .htaccess. En estos casos no será necesario adjuntar el nombre del archivo porque la configuración de estos ficheros únicamente se aplica en el directorio en el que se encuentran.

<FilesMatch>

```
Sintaxis: <FilesMatch regex> ... </FilesMatch>
Contexto: configuración servidor, servidor virtual, .htaccess
Compatibilidad: únicamente disponible a partir de Apache 1.3
```

Exactamente igual que la directriz <File> con la salvedad de que permite utilizar expresiones regulares como argumento. Por ejemplo:

```
<FilesMatch "\.(zip tar|tgz|arj|zoo)$">
```

marcarían cualquier archivo con la extensión zip, tar, tgz, arj y zoo. Obsérvese que se ha prescindido del carácter ~ para identificar una expresión regular.

<Location>

```
Sintaxis: <Location URL> ... </Location>
Contexto: configuración servidor, servidor virtual
```

Esta directriz se utiliza para controlar el acceso a través de un URL. Los contenedores <Location> se procesan en el mismo orden en que aparecen en el archivo de configuración, después de que se lean las secciones <Directory> y los archivos .htaccess.

No hace falta que el argumento URL tenga el formato http://nombre_servidor. Se pueden usar comodines como ? (que sustituye a un único carácter) o * (que sustituye a una secuencia de caracteres, a excepción de /). También se pueden ampliar las expresiones regulares con el prefijo ~. Por ejemplo:

```
<Location ~ "/(mi|su)/archivo">
```

marcaría los URL tales como /mi/archivo y /su/archivo.

<LocationMatch>

```
Sintaxis: <LocationMatch regex> ... </LocationMatch>
Contexto: configuración servidor, servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.3
```

Exactamente igual que la directriz <Location> con la salvedad de que permite utilizar expresiones regulares como argumento.

Por ejemplo:

```
<Location "/(mi|su)/archivo">
```

marcaría los URL tales como /mi/archivo y /su/archivo.

Directrices específicas del servidor virtual

Estas directrices se utilizan para crear servidores virtuales. Por defecto, Apache únicamente trabaja con el sitio Web cuyo nombre se especifica a través de la directriz ServerName. De todas formas, es posible hacer que trabaje con otros sitios Web gracias a un contenedor de un servidor virtual. Obsérvese que muchas de las directrices que se han visto en las secciones anteriores también se pueden utilizar con los servidores virtuales.

<VirtualHost>

```
Sintaxis: <VirtualHost add[:port] ...> ... </VirtualHost>
Contexto: configuración servidor
Compatibilidad: acepta varias direcciones a partir de Apache 1.2
```

Este contenedor engloba las directrices de configuración de un servidor virtual. Las órdenes que se encuentren entre <VirtualHost> y </VirtualHost> únicamente tendrán vigencia en el servidor para el que se configuran. Se puede utilizar cualquier directriz aplicable a un servidor virtual. Cuando el servidor recibe una solicitud relacionada con un documento que se encuentra en un servidor virtual, utiliza la configuración especificada en esta sección.

Para determinar las direcciones y el nombre IP que se tiene que utilizar con un servidor en particular, puede utilizar:

- Una dirección IP. Ejemplo: <VirtualHost 11.22.33.44> ... </VirtualHost>
- Una dirección IP con un número de puerto. Por ejemplo: <VirtualHost 11.22.33.44:8080> ... </VirtualHost>

- Varias direcciones IP. Ejemplo: <VirtualHost 11.22.33.1 11.22.33.2> ... </VirtualHost>
- Varias direcciones IP con números de puerto. Por ejemplo: <VirtualHost 11.22.33.1:8000 11.22.33.2:10000> ... </VirtualHost>

Aunque se pueden sustituir las direcciones IP por nombres IP, no es una práctica muy recomendada; si la comprobación DNS fallase por alguna razón, el servidor se encontraría ante una situación confusa y no trabajaría con el sitio virtual.

Se puede especificar un nombre predeterminado en cuyo caso el servidor virtual tendrá que coincidir con cualquier dirección que no corresponda con cualquier otro servidor virtual. En el caso en que no se tenga un servidor virtual predeterminado, la configuración del servidor principal (consistente en una serie de definiciones que se encuentran fuera de la sección VirtualHost, será la que se utilice cuando no haya ninguna coincidencia).

Si se deja un puerto sin especificar, se tomará como predeterminado el puerto definido en la última directriz Port del servidor principal. También se puede especificar :* para marcar todos los puertos de dicha dirección.

NameVirtualHost

```
Sintaxis: NameVirtualHost addr[:port]
Contexto: configuración servidor
Compatibilidad: únicamente disponible a partir de Apache 1.3
```

Si tiene pensado utilizar servidores virtuales basándose en sus nombres, tendrá que utilizar esta directriz. Aunque addr puede ser el nombre del servidor, mi consejo es que siempre se utilice la dirección IP. Por ejemplo, para un servidor virtual cuyo nombre sea www.miempresa.com que utilice la dirección IP 11.22.33.44, la directriz que defina la dirección virtual será:

```
NameVirtualHost 11.22.33.44
<VirtualHost 11.22.33.44>
ServerName www.miempresa.com
</VirtualHost>
```

Si tiene varios nombres de servidores basados en varias direcciones, repita la directriz para cada una de ellas. Por ejemplo:

```
NameVirtualHost 11.22.33.44

# Primer servidor virtual que se corresponde con la directriz
# anterior
```

```
<VitualHost 11.22.33.44>
ServerName www.miempresa.com
</VirtualHost>

# Segundo servidor virtual que se corresponde con la directriz
# anterior
<VitualHost 11.22.33.44>
ServerName www.amigosdelaempresa.com
</VirtualHost>

# Otra directriz NameVirtualHost que se corresponde con otro servidor
# que utiliza una dirección IP diferente
NameVirtualHost 11.22.33.55

<VitualHost 11.22.33.55>
ServerName www.micliente.com
</VirtualHost>
<VitualHost 11.22.33.55>
ServerName www.sucliente.com
</VirtualHost>
```

En este caso, la primera directriz NameVirtualHost se utiliza con los servidores www.miempresa.com y www.amigosdelaempresa.com. La segunda se usa con www.micliente.com y www.sucliente.com, que también son servidores virtuales. Si lo desea puede especificar el número de un puerto que se utilizará en vez del nombre del servidor. Por ejemplo:

```
NameVirtualHost 11.22.33.44:8080
```

ServerAlias

```
Sintaxis: ServerAlias host1 host2 ...
Contexto: servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.1
```

Cuando tenga que referirse a un servidor que tiene varios nombres IP (entradas CNAME en la base de datos DNS) puede utilizar una única definición de servidor virtual para definir el servicio de todos ellos. Por ejemplo:

```
NameVirtualHost 11.22.33.55

<VitualHost 11.22.33.55>
ServerName www.micliente.com
ServerAlias www.sac-state.edu www.csu.sacramento.edu
</VirtualHost>
```

En este caso www.sac-state.edu y www.csu.sacramento.edu son sobrenombres (alias) del servidor virtual www.micliente.com. Para definir alias también se pueden utilizar comodines.

ServerPath

```
Sintaxis: ServerPath nombre_del_path
Contexto: servidor virtual
Compatibilidad: únicamente disponible a partir de Apache 1.1
```

Esta directriz se utiliza para definir el URL del path del servidor para utilizarlo con los servidores virtuales basados en el nombre. Generalmente, se utiliza para que se pueda trabajar con los exploradores Web que no son compatibles con HTTP/1.1.

Directrices de registro

El registro de las transacciones del servidor es algo que resulta obligado para cualquier servidor Apache. Los registros le proporcionan una información de inestimable valor, como por ejemplo, para saber quién ha accedido a su sitio Web, las páginas que se han visitado y qué errores ha generado el servidor.

ErrorLog

```
Sintaxis: ErrorLog nombre_archivo
Predeterminado: ErrorLog logs/error_log
Contexto: configuración servidor, servidor virtual
```

Esta directriz especifica el nombre del archivo de registro donde se guardan los mensajes de error que genera el servidor. Si el nombre del archivo no comienza con una barra inclinada (/), entonces se asume que es relativo a ServerRoot.

Si tiene que desactivar el registro de errores tendrá entonces que escribir lo siguiente:

```
ErrorLog /dev/null
```

Advertencia: es muy importante que la configuración de permisos correspondiente al directorio en el que se guardan los archivos de registro indique que únicamente el usuario de Apache (especificado en la directiva User) posee los permisos de escritura y lectura. Si se permite que cualquier pueda leer y escribir en este directorio, estará abriendo una brecha en el sistema de seguridad.

ScoreBoardFile

```
Sintaxis: ScoreBoardFile nombre_archivo
Predeterminado: ScoreBoardFile logs/apache_status
Contexto: configuración servidor
```

Esta directriz determina el path del archivo que se utiliza para guardar todos los datos relacionados con los procesos internos del servidor. Si el nombre del archivo no comienza con una barra inclinada (/), entonces se asume que es relativo a ServerRoot.

Este fichero lo utilizan los procesos del servidor principal para comunicarse con los subprocesos. Si quiere averiguar si su sistema precisa de la existencia de dicho archivo, tendrá que ejecutar el servidor Apache y ver si crea este archivo en la ubicación especificada. Si la arquitectura de su sistema necesita dicho fichero, tendrá que asegurarse de que Apache no lo llama a la vez desde dos puntos distintos. Además, tendrá que verificar que ningún otro usuario tiene acceso de lectura o escritura a dicho archivo (ni siquiera al directorio en el que se encuentra).

Como los procesos utilizan la I/O del disco para establecer sus comunicaciones, ha de tener en cuenta que en un momento dado se puede encontrar con un cuello de botella. Por eso es conveniente crear un disco RAM. Para más detalles, consulte los manuales de su sistema.

PidFile

```
Sintaxis: PidFile nombre_archivo
Predeterminado: logs/httpd.pid
Contexto: configuración servidor
```

Al usar esta directriz le puede indicar a Apache que escriba el número de identificación ID del proceso del servidor principal (es decir, del demonio), también conocido como PID, en un archivo. Si el nombre del archivo no empieza por /, entonces se da por supuesto que es relativo a ServerRoot. PidFile se utiliza sólo con el modelo standalone.

Obsérvese que se utiliza para ayudar al administrador de Apache a localizar el PID principal, que es necesario siempre que se envía una señal al servidor. Por ejemplo, si el archivo PID se guarda en el directorio /usr/local/httpd/logs y su nombre es httpd.pid, un administrador puede forzar a Apache a leer su configuración gracias a la orden SIGHUP a través de la línea de comandos (siempre como root):

```
kill -HUP 'cat /usr/local/httpd/logs/httpd.pid'
```

El mismo comando puede hacer que Apache vuelva a abrir ErrorLog y TransferLog.

Advertencia: al igual que ocurre con otros archivos de registro, asegúrese de que nadie, a excepción del proceso del servidor, pueda leer ni escribir en el fichero PID. Por cuestiones de seguridad conviene que haga que el directorio de registro (log) sólo lo pueda leer el usuario del servidor Apache.

LockFile

```
Sintaxis: LockFile nombre_archivo
Predeterminado: LockFile logs/accept.lock
Contexto: configuración servidor
```

Si se compila Apache utilizando las opciones USE_FCNTL_SERIALZED_ACCEPT o USE_FLOCK_SERIALZED_ACCEPT, se usará el bloqueo de archivos. Se puede utilizar esta directiva para definir el path del archivo de bloqueo. Asegúrese de que tan solo el servidor Apache puede leer y escribir en este fichero.

Advertencia: no es conveniente almacenar el archivo de bloqueo en un sistema de red NFS con una partición porque precisamente el sistema NFS destaca por la cantidad de problemas de seguridad que tiene en relación al bloqueo de archivos.

Directrices de autenticación y seguridad

Las directrices de autenticación y seguridad aquí descritas le permiten definir las políticas de acceso que regirán el servidor.

AllowOverride

```
Sintaxis: AllowOverride sustitución sustitución ...
Predeterminado: AllowOverride All
Contexto: directorio
```

Esta directriz le indica al servidor qué directrices, de las declaradas en el archivo .htaccess (como se especificó en AccessFileName), pueden sustituir

a las que se encontró en los primeros archivos de configuración. Cuando el argumento de Override sea None, el servidor no leerá el archivo especificado en AccessFileName (por defecto .htaccess). De esta forma se puede acelerar el tiempo de respuesta ya que el servidor no tendrá que consultar dicho archivo en cada una de las peticiones recibidas.

De todas formas, si desea trabajar con un control basado en AccessFileName tendrá que especificar una o más opciones. Estas son las siguientes:

- AuthConfig: permite el uso de directrices de configuración (tales como por ejemplo AuthDBMGroupFile, AuthDBMUserFile, AuthGroupFile, AuthName, AuthType, AuthUserFile y require).
- FileInfo: permite utilizar directrices de configuración de los tipos de documentos (como AddEncoding, AddLanguange, AddType, DefaultType, ErrorDocument y LanguagePriority).
- Idexes: permite usar directrices para controlar la indexación de directorios (como AddDescription, AddIcon, AddIconByEncoding, AddIconByType, DefaultIcon, DirectoryIndex, FancyIndexing, HeraderName, IndexIgnore, IndexOptions y ReadmeName).
- Limit: permite utilizar directrices para controlar el acceso al servidor (permitir, denegar y ordenar).
- Options: permite utilizar directrices para controlar las características propias del directorio (Options y XBitHack).

AuthName

```
Sintaxis: AuthName etiqueta
Contexto: directorio, .htaccess
Override: AuthConfig
```

Esta directriz define la etiqueta de un recurso (por ejemplo, un directorio) que se utiliza en la autenticación. Dicha etiqueta aparece en los cuadros de diálogo de los exploradores Web que muestran al usuario cuándo éste tiene que introducir su nombre y contraseña para acceder al recurso solicitado. No hay ninguna etiqueta predeterminada. Su principal función es informar a los clientes sobre el recurso al que intentan acceder. Por ejemplo:

```
AuthName Secured Game Zone
```

le indica al usuario que está intentando acceder a la zona de seguridad de juegos del servidor.

Obsérvese que para que funcione esta directriz ha de ir acompañada de AuthType y de las directrices require, AuthUserFile y AuthGroupFile.

AuthType

```
Sintaxis: AuthType tipo
Contexto: directorio, .htaccess
Override: AuthConfig
```

Esta directriz selecciona el tipo de autenticación de un directorio. En la actualidad Apache incorpora la autenticación HTTP. No conviene utilizarla en casos muy serios porque tanto la contraseña como el nombre del usuario se transmiten en texto plano. Dichos datos se envían en cada solicitud referida al directorio restringido o a cualquiera de sus subdirectorios.

Para que tenga efecto ha de ir acompañada de Authname, require y otras directrices como AuthUserFile y AuthGroupFile.

require

```
Sintaxis: require nombre_entidad entidad entidad ...
Contexto: directorio, .htaccess
Override: AuthConfig
```

Al utilizar esta directriz, Apache determina qué usuarios o grupos pueden acceder a un directorio restringido. Hay tres tipos de nombres de entidades válidos:

- user (usuario)
- group (grupo)
- valid-user (usuario válido)

Por ejemplo, esta línea:

```
require user joe jenny
```

le indica al servidor Apache que únicamente joe y jenny pueden acceder al área restringida después de que el proceso de autenticación tenga éxito. Un ejemplo de acceso de grupo podría ser éste:

```
require group mi_grupo su_grupo
```

Sólo los usuarios pertenecientes a dichos grupos podrán acceder a la zona restringida.

Si dicha directriz apareciese dentro de una sección <Limit>, entonces la restricción únicamente sería válida para los métodos nombrados. En caso contrario se restringiría el acceso de todos los métodos. Por ejemplo:

```
AuthType Basic
AuthName Game Zone Drop Box
AuthUserFile /www/netgames/users
AuthGroupFile /web/ngames/groups
<Limit GET>
require group coders
</Limit>
```

Si la configuración anterior apareciese dentro de un archivo .htaccess de un directorio, el único grupo que podría acceder a dicha carpeta sería coders y podría recuperar los archivos allí almacenados a través del método HTTP GET. Para que funcionase correctamente, la directriz requerida ha de estar acompañada por AuthName y por AuthType, además de por AuthUserFile o AuthGroupFile.

Satisfy

```
Sintaxis: Satisfy 'any' | 'all'
Predeterminado: Satisfy all
Contexto: directorio, .htaccess
Compatibilidad: únicamente disponible a partir de Apache 1.2
```

Si ha creado una configuración de una autenticación HTTP básica en la que permite el uso de directrices require, puede utilizar Satisfy para indicarle a Apache que es completamente necesario que se cumplan todas las condiciones requeridas. El valor de esta directriz puede ser all o any. Si es all (todo) la autenticación sólo tendrá éxito cuando se cumplan allow (permitido) y require (requerido). Si es any (cualquiera) bastará con que se cumpla cualquiera de los dos casos.

Esta directriz únicamente será de alguna utilidad si el acceso a cierta zona está restringido tanto por el nombre/contraseña y la dirección del cliente del servidor. En este caso, el comportamiento predeterminado (all) requiere que el cliente pase la restricción de la dirección e introduzca un nombre y una contraseña válidos. Con la opción any el cliente tendrá acceso si pasa la restricción de la dirección o introduce correctamente el nombre y la contraseña. Esta directriz se puede utilizar para limitar el acceso a una zona a través de contraseñas y, a la vez, a todos los clientes procedentes de un mismo conjunto de direcciones IP se les permite el acceso sin necesidad de presentar una contraseña.

IdentityCheck

```
Sintaxis: IdentityCheck boolean
Predeterminado: IdentityCheck off
Contexto: configuración servidor, servidor virtual, directorio,
.htaccess
```

Esta directriz le ordena al servidor Apache que registre todos los nombres de los usuarios remotos interactuando con sus demonios de identificación (identd) o con algo similar y con la compilación RFC1413. La verdad es que esta directriz no se suele utilizar porque no funciona con todos los sistemas. La mayoría no ejecutan los procesos identd.

Si opta por utilizarla, tenga en cuenta que únicamente se accederá a la información registrada para hacer una auditoría. Esta directriz también puede causar problemas de eficacia porque el servidor tiene que comprobar todas las solicitudes recibidas. Además, cuando un usuario remoto no suministra su identidad o se encuentra al otro lado de un firewall o proxy, el proceso de comprobación ha de tener fecha de caducidad.

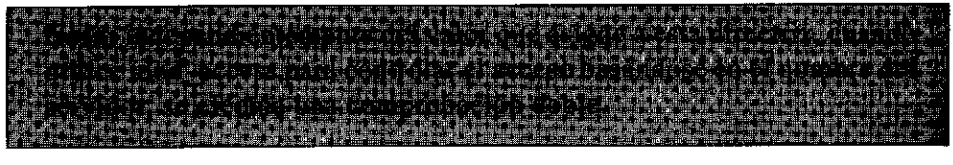
HostNameLookups

```
Sintaxis:HostNameLookups on | off | double
Predeterminado: HostNameLookups off
Contexto: configuración servidor, servidor virtual, directorio,
.htaccess
Compatibilidad: la opción double únicamente disponible a partir de
Apache 1.3
```

Esta directriz le indica a Apache que active o desactive la mejora DNS en cada petición. Cuando se tiene activada, Apache guarda el nombre del servidor del cliente en la variable de entorno REMOTE_HOST de cada proceso CGI o SSI que ejecuta.

Los valores on y off son algo obvio. Double hace referencia a una comprobación DNS doble, invertida. Es decir, que en primer lugar se hace en un sentido y a continuación en el otro.

Por lo menos una de las direcciones IP comprobadas ha de coincidir con la original. De todas formas, sepa que los procesos CGI y SSI no se llevan muy bien con la opción double.



Mi consejo es que deje el valor predeterminado. Aliviará a Internet de bastante tráfico por otra parte innecesario. Si quiere activarlo para que en sus archivos de registros aparezcan los nombres IP en vez de las direcciones IP, es posible que tenga que considerar otra opción, como ejecutar una utilidad logresolve que se encargue de convertir las direcciones IP en nombres IP.

<Limit>

```
Sintaxis: <Limit método método ... > ... </Limit>
Contexto: cualquiera
```

Este contenedor se utiliza para englobar un grupo de directrices de control y aplicarlas a uno de los métodos HTTP especificados. Los nombres de los métodos pueden ser cualquiera de los siguientes (incluso más de uno): GET, POST, PUT, DELETE, CONNECT y OPTIONS. Si se utiliza GET también se restringirán las solicitudes HEAD. Si quiere limitar todos los métodos, no escriba ninguna directriz dentro de <LIMIT>

Observe que este contenedor no se puede anidar, ni que se puede utilizar dentro de él el contenedor <Directory>.

5 Módulos de Apache

En el capítulo anterior nos hemos centrado en las directrices del núcleo de Apache. Este servidor tiene otras muchas directrices a las que se puede acceder a través de los módulos que se distribuyen con la versión estándar. Gracias a dichas directrices se aumenta considerablemente la funcionalidad de Apache.

En este capítulo hablaremos de estos módulos y sus directrices. Para que su lectura resulte más sencilla aparecen por orden alfabético.

mod_access

Su construcción es la predeterminada. Tiene una serie de directrices especializadas en el control de acceso al servidor, que se verán en el capítulo 10.

mod_actions

Para su compilación se utilizan los valores predeterminados. Le permite ejecutar un script CGI basado en el tipo MIME o en un método de solicitud HTTP. Dispone de las siguientes directrices.

Action

```
Sintaxis: Action tipo_MIME script_CGI
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
Estado: Base
```

Esta directriz le permite asociar una acción con un tipo MIME específico. La acción suele ser un script CGI que procesa el archivo cada vez que se le pide. Por ejemplo:

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

Con esta sentencia se consigue que Apache ejecute un script cada vez que se solicita un archivo HTML. El script recibe el URL y la dirección del documento solicitado a través de las variables de entorno CGI estándar PATH_INFO y PATH_TRANSLATED. Pueden ser útiles para desarrollar script de filtrado. En esta sección veremos uno de ellos.

Cuando se solicite un archivo de texto (.txt) desde la Red, aparecerá en el explorador en un formato poco deseable porque estos programas de navegación no traducen los saltos de línea. De hecho, la mayoría muestra los párrafos en una sola línea muy larga. Por medio de la directriz Action puede desarrollar una solución. Para demostrárselo le expongo a continuación el listado de un script que no sólo se limita a traducir los saltos de línea sino que inserta un mensaje con el Copyright al final del texto.

Para que funcione tendrá que añadir la siguiente directriz en el archivo de configuración de Apache (srm.conf):

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

A continuación desarrollar un script en Perl que le muestre el texto de la forma deseada. Lo puede ver en el listado 5.1.

```
#!/usr/local/bin/perl
#
# Script: textfilter.pl
#
# Propósito: script de filtrado que convierte los archivos de texto
#            plano en documentos HTML pero conserva la disposición
#            original.
#
# $Author$
# $Revision$
# $Id$
```

```
# $Status
#

# El archivo en el que se encuentra el mensaje del copyright
# está guardado en el directorio root de documentos y su
# nombre es copyright.html.
#
my $copyright_file = $ENV{DOCUMENT_ROOT} . "/copyright.html";

# Consigue el path del documento solicitado
my $path_translated = $ENV{PATH_TRANSLATED};

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

# Guarda la información sobre el path y el nombre del archivo en una
# matriz
@filename = split(/\//,$path_translated);

# Como se van a utilizar etiquetas HTML para visualizar el archivo
# de texto, habrá que imprimir el contenido de la cabecera text/html.
print "Content-type: text/html\n\n";

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

# Ahora imprime las siguientes etiquetas del documento HTML.
# Las enviará antes del contenido del documento
#
print <<HEAD;
<HTML>
<HEAD>
<TITLE>$filename[-1] </TITLE>
</HEAD>
<BODY BGCOLOR="white">

<BLOCKQUOTE>

<PRE>

<FONT FACE="Arial">

HEAD

# Ahora imprime cada línea almacenada en la variable @text
# (por ejemplo, el contenido del archivo solicitado)
#
foreach $line (@text){ print $line; }

# Ahora se lee el archivo del copyright y se guarda su contenido
# en la variable @html
#
@html = &readFile($copyright_file);
```

```
# Imprime cada una de las líneas almacenadas en la variable @html
# (por ejemplo, el contenido del fichero del copyright)
#
foreach $line (@html){ print $line; }

# Sale del filtro
exit 0;

sub readFile{
#
# Subrutina: readFile
# Propósito: lee el archivo si existiese o imprime
# un mensaje de error al salir del script
#

# Consigue el nombre del archivo que se lo ha entregado
# y lo guarda en la variable $file
 my $file = shift;

# Búfer local variable
 my @buffer;

# Si el archivo existiese, lo abre y lee todas las líneas que se
# encuentran en la variable @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 búfer.
 return (@buffer);
 }
```

Listado 5.1. textfilter.pl

Este script lee el archivo de texto solicitado e imprime su contenido dentro de etiquetas HTML gracias a las cuales se puede ver correctamente. Este truco se debe al uso de la etiqueta <PRE>. Una vez que se muestra el contenido del archivo solicitado, al final del todo se muestra el mensaje del Copyright. Así se puede imprimir este mensaje en todos los archivos de texto que utilicen este filtro. La figura 5.1 nos muestra el resultado obtenido al utilizar el filtro con un archivo de texto.

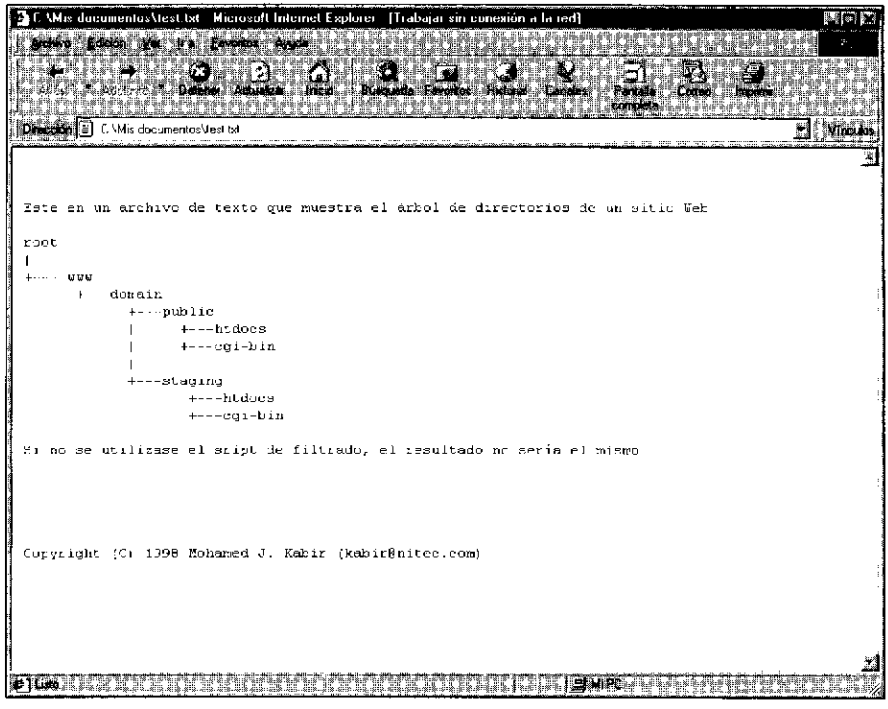


Figura 5.1. Salida del archivo test.txt

Como puede apreciar, el nombre del archivo aparece como título. El documento es un bloque entrecomillado y se imprime un mensaje de Copyright personalizado. El archivo que contiene el Copyright se encuentra en el archivo root. El que se ha utilizado en este ejemplo es:

```
<BLOCKQUOTE>
<CENTER>
<HR>
Copyright (c) 1998 Mohammed J. Kabir () ({Hyperlink
Mail to:kabir@nitec.com})
</CENTER>
</BODY>
</HTML>
```

Script

```
Sintaxis: Script método cgi-script
Contexto: configuración servidor, servidor virtual, directorio
```

Esta directriz es muy parecida a Action, pero en vez de asociar una acción a un tipo MIME la asocia con una petición HTTP como GET, POST, PUT o

DELETE. Este script CGI recibe el URL y el path del archivo solicitado a través de las variables de entorno CGI estándar PATH_INFO y PATH_TRANSLATED. Con esta directriz se define una acción predeterminada, en otras palabras, si ha definido de antemano:

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

en uno de los archivos de configuración (como srm.conf), entonces cualquier petición efectuada a través del método HTTP POST se procesará como siempre a menos que haya que utilizar la acción predeterminada. Por ejemplo, el siguiente formulario HTML no especifica un script CGI para esta acción:

```
<FORM METHOD="POST">
Introduzca el Nombre: <INPUT TYPE="TEXT NAME="nombre" SIZE=25>
<INPUT TYPE=SUBMIT VALUE="Haga clic aquí">
</FORM>
```

Si el usuario introduce un nombre a través de este formulario, se encontrará con que no hay ninguna acción diseñada para que procese esta información, en cuyo caso POST ejecutaría el script /cgi-bin/default_post.pl. De todas formas si efectuamos el siguiente cambio:

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

cada vez que se envíe el formulario, se llamará al script /cgi-bin/form_processor.pl como predeterminado. Lo que se tenga que hacer a continuación depende enteramente del programador. En la configuración de un proveedor de Internet debería imprimir algún mensaje para que el usuario tenga alguna pista que le indique si lo que está haciendo está bien o mal.

Nota: en el caso de encontrarse con una petición GET, sólo se efectuará la acción predeterminada si la solicitud va acompañada de los datos oportunos. Por ejemplo, el archivo www.susitio.com/fichero.html se procesará como siempre, pero si se solicita el archivo www.susitio.com/fichero.html?algun=dato, entonces se ejecutará la acción predeterminada para GET.

mod_alias

Este módulo se compila con la versión predeterminada de Apache. Tiene una serie de directrices que se pueden utilizar para convertir una parte de los

archivos de sistema del servidor en otra o utilizar incluso para redirigir servicios URL.

Alias

```
Sintaxis: Alias URL-path path
Contexto: configuración servidor, servidor virtual
```

Esta directriz le permite transformar cualquier path en un archivo del sistema. Por ejemplo:

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

Con esta sentencia se transforma /data/ en /web/data; por lo tanto, cuando se recibe una petición del tipo www.susitio.com/data/datafile.csv, se muestra el archivo /web/data/datafile.csv.

El path simulado no tiene que encontrarse dentro del árbol de directorios root de documentos, así que tenga cuidado cuando trabaje con los alias, ya que accidentalmente puede llegar a revelar alguna parte de su sistema a todo el mundo.

Obsérvese que cuando utiliza una barra inclinada (/) en la definición de alias, todas aquellas peticiones que se dirijan a dicho alias tendrán que incluir dicho carácter. Por ejemplo:

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

solamente funcionará con www.susitio.com/data/ y no con www.susitio.com/data. En el último caso Apache busca un directorio de datos que se encuentre dentro del árbol de directorio y no dentro de /web/data.

AliasMatch

```
Sintaxis: AliasMatch regex path
Contexto: configuración servidor, servidor virtual
```

Es muy parecido a la directriz Alias, sólo que aquí se pueden usar expresiones regulares. Por ejemplo:

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

Con esta sentencia convierte www.susitio.com/data/index.html en web/data/index.html.

Redirect

```
Sintaxis: Redirect [estado] antiguo-URL nuevo-URL
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
```

La directriz redirige una petición a un URL a otro nuevo. Por ejemplo:

```
Redirect /data www.su-nuevo-sitio.com/data
```

De esta forma se redirigen todas las solicitudes que contengan el directorio /data a un URL nuevo. Por lo tanto, las solicitudes www.susitio.com/data/archivo.txt se dirigen a www.su-nuevo-sitio.com/data/archivo.txt.

La directriz Redirect tiene una gran precedencia sobre Alias y ScriptAlias. Por defecto, el código de estado que se envía al cliente es Temp (código de estado HTTP 302). Si desea especificar otro código de estado, utilice lo siguiente:

- Permanent: le indica al cliente que la redirección es permanente. Se muestra el código de estado HTTP 301.
- Temp: devuelve un estado de redirección temporal (302). Es la opción predeterminada.
- See other: muestra el estado See other (vea otro) cuyo código es el 303, con lo que se indica que se ha sustituido el recurso.
- Gone: devuelve el estado Gone (410) con lo que se indica que el recurso se ha eliminado definitivamente. Cuando se utiliza este estado, se tiene que omitir el argumento URL.

Si lo desea, puede utilizar el formato numérico para indicar el código de estado HTTP. Si este número está comprendido entre 300 y 399, indicará que hay un URL nuevo. En caso contrario se tendrá que omitir. Se puede preguntar por el uso de diferentes códigos de estado para distintas situaciones con sentido.

Por ejemplo, si un servidor proxy recibe un código de redirección permanente, podrá guardar dicha información en memoria caché para utilizarla cada vez que reciba una petición nueva.

RedirectMatch

```
Sintaxis: RedirectMatch [estado] regex URL
Contexto: configuración servidor, servidor virtual
```

Esta directriz es igual a Redirect sólo que acepta expresiones regulares en vez del URL antiguo. Por ejemplo:

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

Con esta sentencia se redirigen todas las peticiones que terminen en .htm a la versión .html del documento solicitado. Por ejemplo, la petición:

```
www.sustio.com/agún/archivo/antiguo/de/dos/index.htm
```

se redirigirá a:

```
www.sustio.com/agún/archivo/antiguo/de/dos/index.html
```

RedirectTemp

```
Sintaxis: RedirectTemp antiguo-URL nuevo-URL
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
```

Similar a la directriz Redirect. Permite que el cliente sepa que la redirección tan solo es temporal. Observe que la directriz Redirect, por defecto, también produce un estado temporal.

RedirectPermanent

```
Sintaxis: RedirectPermanent antiguo-URL nuevo-URL
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
```

Similar a la directriz Redirect. Permite que el cliente sepa que la redirección es permanente. Observe que la directriz Redirect, por defecto, produce un estado temporal pero que, para la misma finalidad, se puede utilizar el código 301 o la cadena de caracteres Permanent.

ScriptAlias

```
Sintaxis: ScriptAlias URL-path path
Contexto: configuración servidor, servidor virtual
```

Esta directriz crea un alias (URL-path) para un path. Además, cualquier archivo que se entregue en la petición será considerado como un script CGI y el servidor tratará de ejecutarlo. Por ejemplo:

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

Se puede utilizar para procesar una petición como www.nitec.com/cgi-bin/script.pl. El servidor tratará de ejecutarlo si comprueba que todos los permisos son correctos. Obsérvese que no se puede navegar por el directorio ScriptAlias.

ScriptAliasMatch

```
Sintaxis: ScriptAliasMatch regex directorio-nombre_archivo
Contexto: configuración servidor, servidor virtual
Estado: base
```

Esta directriz equivale a ScriptAlias sólo que admite el uso de expresiones regulares. Por ejemplo:

```
ScriptAliasMatch ^/cgi-bin(.*)      /www/nitec/pub/cgi-bin$1
```

hará exactamente lo mismo que:

```
ScriptAliasMatch /cgi-bin/    /www/nitec/pub/cgi-bin/
```

mod_asis

Este módulo se compila con la versión predeterminada de Apache. Le permite enviar un documento tal cual, en otras palabras, el documento se envía sin las cabeceras HTTP. Puede resultar útil para redireccionar clientes sin necesitar la ayuda de ningún script. Para enviar un archivo tal cual, tendrá que asegurarse de que uno de los archivos de configuración de Apache (por ejemplo, srm.conf) contiene una entrada como ésta:

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

Esta sentencia asigna el tipo MIME httpd/send-as-is a la extensión .asis. Si crea un archivo llamado foobar.asis y un cliente lo solicita, se le enviará el fichero sin las cabeceras HTTP. El usuario se tendrá que encargar de incluir las cabeceras adecuadas. Por ejemplo, si desea un mecanismo de redirección a través de los archivos .asis, tendrá que crear ficheros con cabeceras como:

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

Por ejemplo, el listado 5.2 muestra un archivo llamado redirect.asis que envía todas las peticiones de los clientes a otra dirección.

```
Status: 301 Nos hemos movido.
Location: http://www.our-new-site/
Content-type: text/html

<H1>Aviso a los visitantes</H1>
Por favor, actualice su bookmark a <A HREF=" www.our-new site/ ">
www.our-new site/ </A><BR>
<BR>
Gracias.
```

Listado 5.2. redirect.asis

Cuando un cliente solicita este archivo, aparece el mensaje de estado 301 que le indica al cliente que utilice la información de ubicación para redirigir su petición. No hay que añadir las cabeceras Date: ni Server:. El servidor se encargará de hacerlo automáticamente. De todas formas, el servidor no toca la cabecera Last-Modified.

mod_auth

Véase el capítulo 10.

mod_auth_anon

Véase el capítulo 10.

mod_auth_db

Véase el capítulo 10.

mod_auth_dbm

Véase el capítulo 10.

auth_external

Véase el capítulo 10.

mod_autoindex

Este módulo se compila con la versión predeterminada de Apache. Cuando se recibe una petición relacionada con un directorio, Apache busca los archivos en los que se encuentran los índices que se especifican en la directriz DirectoryIndex. Generalmente estos archivos serán index.html o index.htm. De todas formas, cuando no se encuentra ninguno de estos archivos, el servidor puede generar un índice sobre la marcha. Este módulo le permite controlar la forma en que Apache generará dicho listado.

Puede generar dos tipos de índices. En la figura 5.2 se muestra uno de ellos.

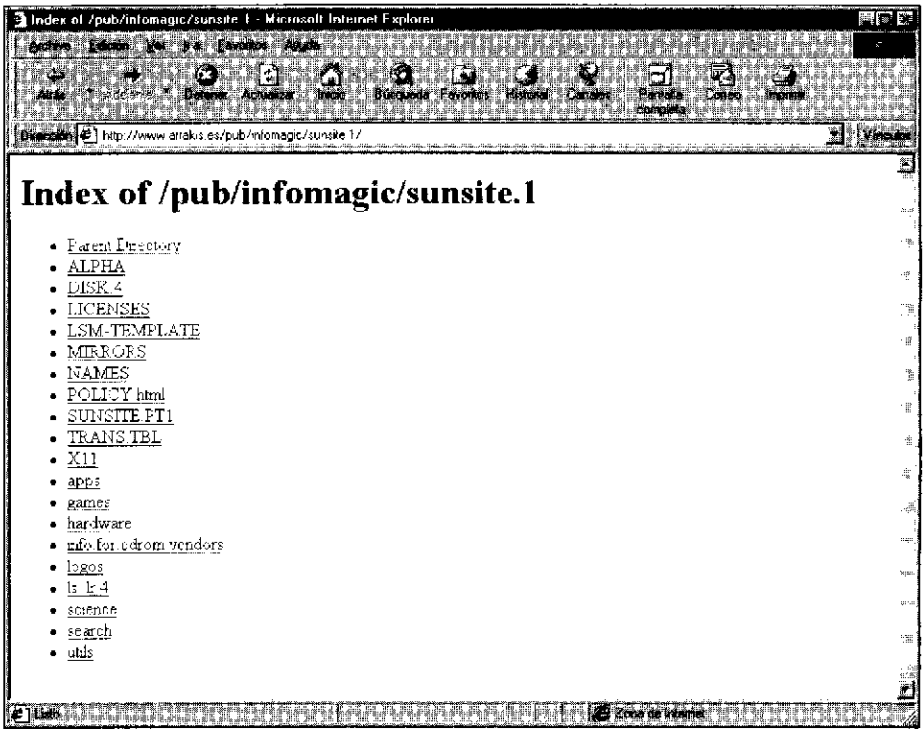


Figura 5.2. Indice generado por Apache

Obsérvese la simplicidad de dicho índice. El otro sistema es algo más atractivo, ya que utiliza una serie de iconos gráficos para diferenciar archivos de directorios. Para generar estos índices tendrá que utilizar las directrices FancyIndexing o IndexOptions que se encuentran en este módulo.

AddAlt

```
Sintaxis: AddAlt "texto" archivo archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Cuando se utiliza FancyIndexing, con esta directriz se determina el texto que aparecerá cuando no es posible asignar ningún icono. Caso que se suele dar cuando el usuario utiliza un explorador que no pueda trabajar con modo gráfico (como Lynx). Por ejemplo:

```
AddAlt "Imágenes" gif jpeg jpg bmp
```

De esta forma Apache mostrará la palabra Imágenes en vez del icono que utiliza para determinar a los ficheros gráficos. Por otro lado, si trabaja con Netscape Navigator o Microsoft Internet Explorer en una plataforma Windows, el texto aparecerá junto al icono como mera ayuda. En este tipo de sistemas, el usuario tiene que colocar el cursor sobre el icono para que aparezca el texto.

AddAltByEncoding

```
Sintaxis: AddAltByEncoding "texto" codificación-MIME codificación MIME
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Si no le gusta asignar texto alternativo a los nombres de archivo o a sus extensiones a través de la directriz AddAlt, puede utilizar AddAltByEncoding para asignárselo a las codificaciones MIME. Al igual que AddAlt, esta directriz sólo se utilizará cuando FancyIndexing esté activada. Por ejemplo:

```
AddAltByEncoding "Archivo comprimido" x-compress
```

AddAltByType

```
Sintaxis: AddAltByType "texto" tipo-MIME tipo-MIME ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Al igual que AddAltByEncoding, esta directriz le permite asignar un texto alternativo para que aparezca en vez del icono que muestra FancyIndexing.

Pero ahora la asignación se basa en el tipo MIME en vez de en la codificación. Por ejemplo:

```
AddAltByType "Archivo HTML" text/html
```

Con esta sentencia aparecerá el texto "Archivo HTML" en lugar del icono correspondiente cuando se trabaje con exploradores Web en modo texto. En el caso de los exploradores en modo gráfico este texto únicamente aparecerá como ayuda.

AddDescription

```
Sintaxis: AddDescription "texto" archivo archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz describe una descripción de texto para el nombre completo, parcial o basado en comodines de uno de los archivos que se encuentren en el sistema. Para ello hay que tener activada la función FancyIndexing. Por ejemplo:

```
AddDescription "Archivos gráficos" *.gif *.jpeg *.jpg *.bmp
```

Con esta sentencia aparecerá una descripción con todos los archivos GIF, JPEG, JPG y BMP.

AddIcon

```
Sintaxis: AddIcon icono nombre nombre ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz le permite asignar iconos a los nombres de archivos y directorios que se muestran con FancyIndexing. Por ejemplo:

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

De esta forma se le está indicando a Apache que muestre el icono /icons/picture.gif junto a todos los archivos que una de las extensiones ahí definidas. Si en vez del icono desea presentar un texto alternativo, tendrá que utilizar un formato similar a este:

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

Aquí, el texto alternativo al icono es IMG, que será lo que aparezca cuando se trabaje con un explorador de modo texto. Si desea mostrar un icono para un directorio determinado, tendrá que escribir lo siguiente:

```
AddIcon /path/al/icono/del/directorio          ^^DIRECTORY^^
```

Del mismo modo, si quiere que aparezca un icono para cada línea en blanco del índice generado con FancyIndexing, puede utilizar:

```
AddIcon /path/al/icono/de/línea/en/blanco      ^^BLANKICON^^
```

AddIconByEncoding

```
Sintaxis: AddIconByEncoding icono codificación-MIME codificación-MIME
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz le permite asignar iconos basándose en la codificación MIME. Por ejemplo:

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

AddIconByType

```
Sintaxis: AddIconByType icono tipo-MIME tipo-MIME ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Le permite asignar iconos basándose en los tipos MIME. Por ejemplo:

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

DefaultIcon

```
Sintaxis: DefaultIcon url
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Cuando ni AddIcon, AddIconByEncoding o AddIconByType puede determinar el icono que ha de utilizar con cierto archivo, se muestra un icono predeterminado. Esta directriz le permite determinarlo. Por ejemplo:

```
DefaultIcon /icon/nose.gif
```

Con esta sentencia se le ordena al servidor que siempre que se encuentre con un archivo al que no sepa que icono asociar, utilice nose.gif.

FancyIndexing

```
Sintaxis: FancyIndexing on | off
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz le permite activar y desactivar la generación de índices con iconos. Se puede obtener el mismo resultado a partir de IndexOptions.

HeaderName

```
Sintaxis: HeaderName nombre_archivo
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Si utiliza FancyIndexing puede insertar el contenido del archivo en la parte superior de la lista del índice. Esta directriz le permite especificar el nombre del archivo cuyo contenido se insertará. Por ejemplo:

```
HeaderName bienvenido
```

Gracias a esta sentencia se consigue que el servidor busque el archivo bienvenido.html en el directorio del que va a generar el índice. Si lo encuentra, imprimirá su contenido en la parte superior del índice.

IndexIgnore

```
Sintaxis: IndexIgnore archivo archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

A la hora de generar el índice de un directorio conviene ocultar ciertos archivos de la mirada del resto del mundo.

Esta es la directriz que le permite seleccionar los archivos que no aparecerán en el índice. Por ejemplo:

```
IndexIgnore bienvenido bienvenido.html per-directory configuration
(.htaccess)
```

Con esta sentencia se asegura de que Apache no incluye ninguno de los archivos anteriormente expuestos en el índice que genere. El carácter "." (punto) aparecerá automáticamente en la lista IndexIgnore; de esta forma, todos los archivos cuyo nombre comience por un punto no se incluirán en el índice. De todas formas yo prefiero añadir la configuración por directorio (.htaccess) para sentirme algo más seguro.

IndexOptions

```
Sintaxis: IndexOptions opción opción ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

La directriz IndexOptions especifica las opciones que determinarán el comportamiento de la indexación. Son las siguientes:

FancyIndexing	Activa la indexación de directorios con iconos. Obsérvese que las opciones FancyIndexing e IndexOptions se solapan la una a la otra.
IconHeight [=píxeles]	Con esta opción permitirá que Apache incluya el atributo HEIGHT=píxeles en la etiqueta IMG del icono, con lo que se obtiene una mayor velocidad de carga en la mayoría de los exploradores Web. Si no se especifica, se utiliza el valor predeterminado.
IconsAreLinks	Los iconos serán parte del nombre.
IconWidth [=píxeles]	Con esta opción permitirá que Apache incluya el atributo WIDTH=píxeles en la etiqueta IMG del icono, con lo que se obtiene una mayor velocidad de carga en la mayoría de los exploradores Web. Si no se especifica, se utiliza el valor predeterminado.
ScanHTMLTitles	Si desea que Apache lea el título (definido en la sección comprendida entre las etiquetas <TITLE> y </TITLE>) de un documento HTML para mejorar el aspecto del índice, use esta opción. De todas formas, si ya ha especificado una opción con la directriz AddDescription, no hace falta que la use. Tenga en cuenta

	que independientemente del contenido del archivo, la búsqueda de la información relativa al título es una tarea que consume recursos del sistema, por lo que es posible que se reduzca el rendimiento de la máquina. La verdad es que no recomiendo el uso de esta opción.
SuppressColumnSorting	Por defecto, Apache deja que se haga clic en la cabecera de una columna para que se ordene la información que contiene. Con esta opción se desactiva esta propiedad.
SuppressDescription	Si no quiere que aparezca en la pantalla la descripción de los archivos, utilice entonces esta opción.
SuppressHTMLPreamble	Si el directorio contiene algún archivo que se haya especificado en la directriz HeaderName, el módulo incluirá su contenido detrás del preámbulo estándar (<HTML>,<HEAD>, etc.). Con esta opción desactiva esta acción.
SuppressLastModified	Hace que no aparezca en pantalla la última fecha en la que se modificó el contenido de los archivos de la lista.
SuppressSize	Esta opción elimina de la lista el tamaño de los archivos.

ReadmeName

```
Sintaxis: ReadmeName nombre_archivo
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Si desea insertar un archivo al final del índice, tendrá que utilizar esta directriz. Por ejemplo:

```
ReadmeName readme
```

Apache buscará un archivo llamado readme.html o simplemente readme y a continuación lo imprimirá al final de la lista de archivos que compone el directorio.

mod_cern_meta

Este módulo no se compila en la versión predeterminada de Apache. Se encarga de la metainformación. Esta información puede ser un extra para las cabeceras HTML, como por ejemplo:

```
Expires: Sábado, 2-Oct-98 12:00:00 GMT
```

o simplemente otra información como:

```
Foo=Bar
```

La metainformación se almacena en un archivo que aparece junto con la cabecera de la respuesta HTTP.

MetaFiles

```
Sintaxis: MetaFiles on/off
Predeterminado: Metafiles off
Contexto: configuración por directorio (.htaccess)
```

Esta directriz activa o desactiva el proceso del archivo metaheader.

MetaDir

```
Sintaxis: MetaDir nombre del directorio
Predeterminado: MetaDir .web
Contexto: configuración por directorio (.htaccess)
```

Esta directriz especifica el nombre del directorio que se usa para guardar los archivos con la metainformación. Si tiene un directorio llamado /www/miempresa/public/htdocs y quiere guardar los archivos de metainformación para dicho directorio, tendrá que crear un directorio llamado .web (si utiliza el valor predeterminado de esta directriz).

MetaSuffix

```
Sintaxis: MetaSuffix sufijo
Predeterminado: MetaSuffix .meta
Contexto: configuración por directorio (.htaccess)
```

Esta directriz especifica la extensión de los archivos en los que se guarda la metainformación. Por ejemplo, si tiene un archivo HTML que se llame

mipágina.html, tendrá que crear mipágina.html.meta (utilizando el valor predeterminado de la directriz) en el que guarde las metacabeceras. El metarchivo se tendrá que guardar en el directorio especificado en la directriz MetaDir.

Para permitir que Apache envíe metainformación al directorio /www/miempresa/public/htdocs tendrá que seguir estos pasos:

1. Configurar la directriz MetaFiles que se encuentra en el archivo de configuración por directorio (.htaccess) para /www/miempresa/public/htdocs. También puede definir las directrices MetaSuffix y MetaDir dentro de este mismo directorio.
2. Crear el subdirectorio .web (suponiendo que utiliza el valor predeterminado de MetaDir).
3. Crear un archivo de texto con la extensión .meta (suponiendo que utiliza el valor predeterminado de MetaSuffix).
4. Colocar en este archivo todas las cabeceras HTTP que desee utilizar.

Por ejemplo, para suministrar metacabeceras para el archivo /www/miempresa/public/htdocs/mipágina.html tendrá que crear un archivo llamado/www/miempresa/public/htdocs/mipágina.html.meta.

En este fichero se puede incluir la línea:

```
Expires: Sábado, 3-Oct-98 12:00:00 GMT
Anything=Whatever
```

mod_cgi

Véase capítulo 8.

mod_digest

Véase capítulo 10.

mod_dir

Este módulo se compila en la versión predeterminada de Apache. Con él Apache podrá redirigir cualquier solicitud relacionada con una directriz que no incluya la barra inclinada. Por ejemplo, se puede utilizar para redirigir

www.susitio.com/directorio a www.susitio.com/directorio/. Incluye una directriz llamada DirectoryIndex que le ayudará a indexar el contenido de un directorio.

DirectoryIndex

```
Sintaxis: DirectoryIndex URL-local URL-local ...
Predeterminado: DirectoryIndex index.html
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz especifica los nombres de los archivos que ha de buscar Apache antes de crear un directorio dinámico. Los archivos pueden ser cualquier cosa, desde un archivo HTML hasta un script CGI. La configuración predeterminada permite que Apache busque el archivo index.html para atender a cualquier petición que termine con el nombre del directorio. Por ejemplo, www.susitio.com/directorio/ hace que Apache busque el archivo www.susitio.com/directorio/index.html. Si lo encuentra, le mostrará su contenido al cliente. En caso contrario, Apache creará un índice dinámico sobre la marcha.

Se puede especificar uno o más archivos como índices. Por ejemplo:

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

De esta forma le está diciendo a Apache que busque todos estos nombres de archivos en el directorio solicitado. Obsérvese que Apache los buscará en el mismo orden en que aparecen aquí. En otras palabras, si Apache encuentra index.html no buscará ningún otro. También se puede especificar el nombre de un script CGI como nombre de índice predeterminado. Por ejemplo:

```
DirectoryIndex /cgi-bin/mostrar_indice.cgi
```

Cada vez que Apache atienda una solicitud relacionada con este directorio, ejecutará el script /cgi-bin/mostrar_indice.cgi.

mod_env

Este módulo no se compila en la versión predeterminada. Permite que entregue variables de entorno a los script CGI o SSI. Dispone de las siguientes directrices.

PassEnv

```
Sintaxis: PassEnv variable variable ...
Contexto: configuración servidor, servidor virtual
```

Esta directriz le dice al módulo que le pase una o más variables de entorno propias del servidor a los script CGI o SSI. Por ejemplo:

```
PassEnv HOSTTYPE PATH
```

SetEnv

```
Sintaxis: SetEnv variable variable ...
Contexto: configuración servidor, servidor virtual
```

Esta directriz determina una variable de entorno que será la que se le entregue a los script CGI o SSI. Por ejemplo:

```
SetEnv CAPITAL_CITY     SACRAMENTO
```

UnsetEnv

```
Sintaxis: UnsetEnv variable variable ...
Contexto: configuración servidor, servidor virtual
```

Esta directriz elimina una o más variables de las variables de entorno entregadas a los script CGI o SSI. Por ejemplo:

```
UnsetEnv PATH
```

mod_expires

Este módulo no se compila con la versión predeterminada de Apache. Le permite terminar con todos los pactos a través de las cabeceras HTTP Expires que se incluyen en la respuesta del servidor. De esta forma tendrá una vía para indicarle al cliente la cantidad de tiempo de que dispone antes de que su solicitud deje de tener validez. Es de gran utilidad cuando el documento al que accede el cliente lo demandan otros usuarios del sistema. Los clientes más inteligentes determinarán la caducidad de un documento que se encuentre en la memoria caché investigando el valor que se encuentra en las cabeceras HTTP. Con este módulo podrá controlar la configuración de estas cabeceras.

ExpiresActive

```
Sintaxis: ExpiresActive on | off
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz le permite activar o desactivar la generación de cabeceras Expires. No garantiza que se genere dicha cabecera. De hecho, si no se cumplen las condiciones requeridas, no se enviará la cabecera.

ExpiresByType

```
Sintaxis1: ExpiresByType tipo MIME M<segundos> | A<segundos>
Sintaxis2: ExpiresByType tipo-MIME "<tiempo base> [plus] <num>
<years|months|weeks|days|hours minutes|seconds>"
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz especifica el valor de la cabecera HTTP Expires de los documentos de un tipo MIME determinado. El tiempo de caducidad se especifica en segundos. Puede definirlo de dos formas. Si opta por utilizar el formato M<segundos>, se usará la fecha en la que se realizó la última modificación como punto de referencia. En otras palabras, M3600 indicará que el archivo caducará una hora después de su última modificación. Por otro lado, si se utiliza el formato A<segundos>, entonces se utilizará como base el tiempo de acceso del usuario. A continuación le mostramos unos cuantos ejemplos.

La siguiente directriz hace que caduquen todos los archivos de texto plano después de llevar una hora en la memoria caché del cliente:

```
ExpiresByType text/plain A3600
```

A continuación se puede ver cómo caducarán los archivos GIF una semana después de la última fecha de modificación:

```
ExpiresByType image/gif M604800
```

Si quiere utilizar la segunda sintaxis para especificar tiempos de caducidad, tendrá que determinar el valor apropiado de <tiempo base> usando una de las opciones siguientes:

- Access: hora en la que el cliente accedió al archivo.
- Now: hora actual. Es la misma que la de acceso.
- Modification: hora de la última modificación.

Por ejemplo:

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

ExpiresDefault

```
Sintaxis1: ExpiresDefault M<segundos> | A<segundos>
Sintaxis2: ExpiresDefault "<tiempo base> [plus] <num>
<years months|weeks|days|hours|minutes|seconds>"
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: Indexes
```

Esta directriz determina el tiempo de caducidad de todos los documentos que se encuentren en el contexto especificado. Por ejemplo, si esta directriz se especificase en el contexto del servidor virtual, sólo se aplicaría a los documentos que se pudiesen acceder a través del servidor virtual. De forma similar, se puede especificar esta directriz en el contexto por directorio, para que todos los documentos que se encuentren en el directorio expiren una vez que se cumpla el tiempo de caducidad. Si desea más detalles sobre esta sintaxis, consulte ExpiresByType. Aquí tiene algunos ejemplos:

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

El primer ejemplo determina la caducidad una hora después de la última modificación de los documentos. En el segundo la caducidad tendrá lugar dos días después del último acceso del cliente.

mod_headers

Este módulo no se compila con la versión predeterminada de Apache. Le permite manipular las cabeceras de las respuestas HTTP. Dispone de una única directriz, Header.

Header

```
Sintaxis1: Header <acción> <cabecera> <valor>
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
```

Esta directriz le permite manipular las cabeceras de las respuestas HTTP. Las acciones que le permite desarrollar son las siguientes:

- Set: define una cabecera. Si existe otra cabecera anterior con el mismo nombre, su valor se actualizará por el más reciente.
- Add: añade una cabecera. Puede hacer que existan varias cabeceras con el mismo nombre.
- Append: asigna un valor a la cabecera existente.
- Unset: elimina la cabecera.

Por ejemplo:

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

Con esta sentencia se añade la cabecera que contiene el nombre del autor "Mohammed J. Kabir". La siguiente sentencia elimina la cabecera:

```
Header unset Author
```

mod_imap

Este módulo se compila con la versión predeterminada de Apache. Suministra un soporte para los mapas de imágenes que se utilizan con un programa CGI llamado imagemap. Se puede utilizar la directriz AddHandler para especificar el controlador de los mapas de imágenes (construidos dentro de este módulo) para cualquier extensión de ficheros. Por ejemplo:

```
AddHandler imap-file map
```

De esta forma Apache tratará todos los archivos que tengan la extensión .map como mapas de imágenes y para procesarlos utilizará este módulo. Obsérvese que todavía se puede utilizar el formato antiguo:

```
AddType application/x-httpd-imap map
```

La verdad es que su uso no es muy recomendado. Las líneas que se encuentran en un archivo imagemap pueden tener cualquiera de estos formatos:

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

Las directrices que se pueden utilizar son las siguientes:

- base: relativo a los URL que se utilizan con los archivos de conversión que hacen referencia al valor de esta directriz. Obsérvese que se sobreescribe la configuración de la directriz Imapbase en el momento en que se encuentra el archivo de conversión. Su valor predeterminado http://nombre_servidor/.base_uri es el sinónimo de base.
- predeterminado: especifica la acción que se tomará cuando las coordenadas no coincidan con un polígono, círculo o recta y no se facilite ninguna directriz de punto. El valor predeterminado es nocontent que le indica al cliente que mantenga en pantalla la misma página.
- poly: define un polígono usando, por lo menos, tres puntos de un máximo de 100. Si el usuario utiliza unas coordenadas que se encuentren dentro del polígono, se activará.
- circle: define un círculo utilizando el centro de las coordenadas y uno de los puntos de la circunferencia. Si el usuario utiliza unas coordenadas que se encuentren dentro del círculo, se activará.
- rect: define un rectángulo utilizando las coordenadas de dos vértices opuestos. Si el usuario utiliza unas coordenadas que se encuentren dentro del rectángulo, se activará.
- point: define las coordenadas de un punto. Esta directriz se activa cuando el usuario se encuentra en unas coordenadas cercanas a las del punto definido y no se cumple ninguna otra directriz.

El valor es un URI absoluto o relativo, o bien uno de los valores especiales que aparece en la siguiente lista. Las coordenadas (x,y) se separan por caracteres en blanco. El texto que se encuentra entre dobles comillas se utiliza como texto de un hipervínculo si se ha generado un menú de mapas de imágenes. Cualquier línea que comience por el carácter # se considerará comentario y se ignorará.

Los valores especiales de la directriz del archivo del mapa de imágenes son los siguientes:

- URL: una URL absoluta o relativa. Las URL relativas se apoyan en la base.
- Map: igual que el URL de un archivo imagemap. A menos que el valor de ImapMenu sea None, se creará un menú.
- Menu: igual que Map.

- Refer: igual que URL pero del documento al que se hace referencia. Si la cabecera Refer: no está presente se usará http://nombre_servidor/.
- noconten: código de estado que toma el valor 204 para indicarle al cliente que mantenga la misma página en la pantalla. Este no es el valor base.
- Error: se envía el código de estado 500 para informarle al cliente de la existencia de un error en el servidor.

Las coordenadas estarán escritas en formato x,y, en donde cada una de ellas aparecerá separada por un espacio en blanco. El texto que se encuentra entre dobles comillas se utiliza como texto de un hipervínculo si se ha generado un menú de mapas de imágenes. Cuando no aparezca dicha cadena, el URL se enlazará tal y como se muestra en el ejemplo siguiente:

```
# Aquí aparecerían los comentarios
# Versión  1.0.0

base http://www.susitio.com/some/dir
rect este_archivo.html "Customer info" 0,0 100,200
circle http://download.susitio.com/index.html 295,0 100,22
```

Si este archivo imagemap se llamase imagemap.map se le podrá invocar desde otro archivo HTML, como:

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

ImapMenu

```
Sintaxis: ImapMenu {none, formatted, semi-formatted, unformatted}
Contexto: configuración servidor, servidor virtual, .htaccess
Override: Indexes
```

Esta directriz determina la acción que se tomará al recibir una solicitud de un archivo imagemap sin que las coordenadas especificadas sean válidas. Puede efectuar las acciones siguientes:

- none: no se genera ningún menú y se emprende la acción predeterminada.
- Formatted: se genera el menú más sencillo. Se ignorarán los comentarios. En primer lugar se imprime una cabecera, a continuación una hrule y luego los vínculos, cada uno en una línea separada.

- Semiformatted: en el menú semi-formateado se imprimen los comentarios. Las líneas en blanco se convierten en saltos HTML. No se imprime ninguna hrule ni cabecera.
- Unformatted: con este menú se imprimen los comentarios. Se ignoran las líneas en blanco.

ImapDefault

```
Sintaxis: ImapDefault {error, nocontent, map, referer, URL}
Contexto: configuración servidor, servidor virtual, .htaccess
Override: Indexes
```

Define la acción predeterminada de los mapas de imágenes. Por medio de la directriz predeterminada se puede sobreescribir esta acción predefinida.

ImapBase

```
Sintaxis: ImapBase {map, referer, URL}
Contexto: configuración servidor, servidor virtual, .htaccess
Override: Indexes
```

Esta directriz define la base predeterminada que se utiliza con los archivos imagemap. La configuración de la base se puede sustituir por la directriz base del mapa de imágenes. Si dicha directriz no está presente, se utilizará el valor predeterminado, http://nombre_servidop/.

mod_include

Véase capítulo 7.

mod_info

Véase capítulo 11.

mod_log_agent

Véase capítulo 11.

mod_log_config

Véase capítulo 11.

mod_log_referer

Véase capítulo 11.

mod_mime

Este módulo se compila con la versión predeterminada de Apache. Se utiliza para entregarle a los clientes metainformación sobre los documentos. Se puede definir un controlador para un documento para determinar cómo lo controlará Apache.

AddEncoding

```
Sintaxis: AddEncoding codificación-MIME extensión-archivo extensión-
archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
```

Esta directriz convierte una o más extensiones de ficheros en una codificación MIME.

Por ejemplo:

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

De esta forma, el archivo backup.gz se tratará como un archivo codificado con x-gzip y tarball.tar se le considerará como si su codificación se correspondiese con x-tar.

AddHandler

```
Sintaxis: AddHandler nombre-controlador extensión-archivo extensión-
archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
```

Esta directriz se utiliza para definir un controlador para una o más extensiones. Por ejemplo:

```
AddHandler server-parsed  .shtml
```

Aquí se indica que los archivos .shtml los ha de procesar un controlador llamado server-parsed, que se encuentra en el módulo mod_include.

AddLanguage

```
Sintaxis: AddLanguage lenguage-MIME extensión-archivo extensión-
archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
```

Esta directriz convierte una lista de extensiones en un lenguaje MIME. Por ejemplo:

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

Convierte todos los archivos con la extensión .en o .english al lenguaje inglés. Resulta interesante para todos aquellos casos en el que el servidor puede mostrarle un documento u otro al cliente basándose en su idioma. Por ejemplo:

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

Si el cliente prefiere el documento en inglés y el servidor tiene dos versiones del mismo, una en inglés (documento.en.html) y otra en francés (documento.fr.html) le mostrará el primero.

AddType

```
Sintaxis: AddType tipo-MIME extensión-archivo extensión-archivo ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
```

Esta directriz convierte una lista de extensiones de archivos en tipo-MIME. Por ejemplo:

```
AddType text/html htm html. HTML MHTL
```

ForceType

```
Sintaxis: ForceType tipo-MIME
Contexto: directorio, configuración por directorio (.htaccess)
```

Esta directriz se usa para forzar ciertos tipos MIME para todos los archivos de un directorio.

La carpeta se especificará en el contenedor <Directorio> o <Location>. Por ejemplo:

```
<Directory> /www/nitec/public/htdocs/archivos/sin/extension>
ForceType text/html
</Directory>
```

Con esta sentencia se obliga a que todos los archivos que se encuentren en dicho directorio se traten como si fuesen del tipo MIME text/html, independientemente de sus extensiones.

SetHandler

```
Sintaxis: SetHandler nombre-controlador
Contexto: directorio, configuración por directorio (.htaccess)
```

Esta directriz se usa para definir un controlador para un directorio o URL. Por ejemplo:

```
<Location /ssi>
SetHandler server-parsed
</Location>
```

Fuerza a todos los archivos que se encuentran en el directorio /ssi a que se traten como SSI, de lo que se encarga el controlador server-persed.

TypesConfig

```
Sintaxis: TypesConfig nombre_archivo
Predeterminado: TypesConfig conf/mime.types
Contexto: configuración servidor
```

Esta directriz especifica el archivo de configuración MIME predeterminado. El valor por omisión debería valer para la mayoría de las instalaciones de Apache.

Si desea añadir sus propios tipos MIME, antes de modificar este archivo, conviene que utilice la directriz AddType.

Truco: si tiene que trabajar con tipos MIME extra, es posible que desee echar un vistazo al módulo mod_mime_magic. Realmente, para la mayoría de instalaciones de Apache, no es necesario, por lo que no se verá en esta obra.

mod_negotiation

Este módulo se compila en la versión predeterminada de Apache. Se hace cargo del contenido de las negociaciones. En un caso típico, el cliente suministra información sobre el tipo de contenido con el que puede trabajar y el servidor trata de suministrárselo. Para ello el servidor se apoya en los mapas de tipos y en la búsqueda MultiViews.

Un mapa de tipos es aquel en el que aparece una descripción de los documentos. Cada una contiene una o más cabeceras. También puede contener comentarios, que son las líneas que comienzan por #. Las descripciones de los documentos están separadas por líneas en blanco. Las cabeceras son las siguientes:

- Content-Encoding: esta especifica el tipo de codificación del archivo. De momento únicamente se admiten las codificaciones x-compress y x-gzip.
- Content-Language: idioma del documento.
- Content-Lenght: tamaño, en bytes, del documento.
- Content-Type: tipo MIME del documento. Se pueden utilizar parámetros del tipo clave=valor. Los dos parámetros admitidos con level y ps. Con el primero se especifica el número de la versión (como entero) del tipo MIME, mientras que con el segundo se indica la calidad (como un número de coma flotante) del documento.
- URI: path del documento relativo al archivo de mapas.

La búsqueda MultiViews trata de determinar qué documento se aproxima más al solicitado (si no se encuentra éste directamente) para lo que se utiliza la información del cliente.

Cuando activa la opción MultiViews en la directriz Options, el servidor podrá efectuar este tipo de búsquedas cuando no localice el archivo que es solicitado.

Mod_negotiation tiene dos directrices.

CacheNegotiatedDocs

```
Sintaxis: CacheNegotiatedDocs
Contexto: configuración servidor
```

Esta directriz permite que los servidores proxy introduzcan los documentos cuyo contenido se ha negociado en la memoria caché. La especificación HTTP/1.1 ofrece un mayor control para este tipo de acciones, con lo que CacheNegotiatedDocs no tiene ningún efecto con las solicitudes HTTP/1.1. Esta directriz se encamina a su extinción, ya que HTTP/1.1 se utiliza cada vez más. Por eso no se recomienda su utilización.

LanguagePriority

```
Sintaxis: LanguagePriority idioma-MIME idioma-MIME ...
Contexto: configuración servidor, servidor virtual, directorio,
configuración por directorio (.htaccess)
Override: FileInfo
```

Esta directriz especifica qué preferencia idiomática ha de utilizar el servidor en una búsqueda MultiViews, cuando el cliente no indica en qué idioma desea que se le presente la respuesta. Por ejemplo:

```
LanguagePriority en fr de
```

Con esta sentencia se le indica al servidor que si el cliente no indica el idioma con el que quiere trabajar, intentará mostrarle la respuesta en inglés. Si no puede, lo intentará en francés, etc. Al igual que la directriz CacheNegotiatedDocs, no tiene ningún efecto con las solicitudes HTTP/1.1.

mod_rewrite

Véase capítulo 18.

mod_setenvif

Este módulo se compila con la versión predeterminada de Apache. Le permite crear variables de entorno personalizadas para que le ayuden a optimizar sus decisiones.

BrowserMatch

```
Sintaxis: BrowserMatch regex variable[=valor] [...]
Predeterminado: nada
Contexto: configuración servidor
Override: nada
```

Esta directriz se utiliza para definir y eliminar variables de entorno personalizadas cuando el modelo coincide con una expresión regular. Por ejemplo:

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

De esta forma se configura una variable llamada vbscript, que tomará el valor no si la cabecera de la petición User-Agent HTTP contiene la palabra Mozilla y a una variable de entorno llamada javascript se le asigna el valor 1 porque no se le ha especificado ningún otro valor. Vamos a ver otro ejemplo:

```
BrowserMatch IE vbscript !javascript
```

Aquí, la variable javascript se ha eliminado y a vbscript se le asigna el valor 1 si se encuentra la palabra IE en la cabecera de la solicitud HTTP.

BrowserMatchNoCase

```
Sintaxis: BrowserMatchNoCase regex variable[=valor] [...]
Predeterminado: nada
Contexto: configuración servidor
Override: nada
```

Esta directriz es la misma que BrowserMatch, sólo que distingue entre mayúsculas y minúsculas en las expresiones regulares. Por ejemplo:

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

Buscaría MSIE, msie, Msie, etc.

SetEnvIf

```
Sintaxis: SetEnvIf atributo regex variable[=valor] [...]
Predeterminado: nada
Contexto: configuración servidor
Override: nada
```

Al igual que BrowserMatch y BrowserMatchNoCase, esta directriz le permite definir y eliminar variables de entorno personalizadas. Realmente,

BrowserMatch y BrowserMatchNoCase son dos versiones de SetEnvIf. Sólo pueden trabajar con las expresiones regulares que aparezcan en el campo User-Agent de las cabeceras de las solicitudes HTTP, mientras que SetEnvIf puede trabajar con todos los campos de las cabeceras. Por ejemplo:

```
SetEnvIf Remove_Host "subdominio\.com" local_user=true
```

Entre las variables que puede utilizar tenemos Remote_Addr, Remote_User, Request_Method, Request_URI y refer, siempre y cuando se encuentren disponibles.

SetEnvIfNoCase

```
Sintaxis: SetEnvIfNoCase atributo regex variable[=valor] [...]
Predeterminado: nada
Contexto: configuración servidor
Override: nada
```

Igual que SetEnvIf pero distingue entre mayúsculas y minúsculas.

mod_spelling

Este módulo no se compila con la versión predeterminada de Apache. Le permite controlar las solicitudes URL mal escritas. Compara el nombre del documento solicitado por el cliente con los nombres de los archivos que se encuentran dentro del directorio solicitado.

En caso de que encuentre alguno parecido, el módulo permitirá errores de un carácter de diferencia, es decir, que se haya escrito un carácter de más, se haya suprimido o equivocado. Si el nombre está bien escrito, pero se han utilizado incorrectamente las mayúsculas y las minúsculas, el módulo efectúa una comparación de nombres. En cualquier caso, si el módulo localiza un documento que parezca que coincide con lo solicitado por el cliente, se lo envía como respuesta. Si hay más de uno que pueda coincidir con el deseado, se los muestra para que sea el cliente quien escoja.

Este módulo tan sólo tiene una directriz, CheckSpelling.

CheckSpelling

```
Sintaxis: CheckSpelling on | off
Predeterminado: CheckSpelling off
Contexto: configuración servidor, servidor virtual
```

Esta directriz permite activar o desactivar el módulo. Cuando la corrección está activada, el servidor puede sufrir una pérdida de efectividad debido a las búsquedas extras que tiene que realizar. Recuerde que este módulo solamente trabaja con los nombres de archivos y directorios.

mod_status

Véase capítulo 11.

mod_unique_id

Módulo que se asegura de que cada petición es única, para lo que se sirve de una serie de condiciones muy específicas. El identificador tendrá que ser único a través de todas las máquinas que formen parte de una misma agrupación. La variable de entorno UNIQUE_ID es la que se encarga de identificar cada petición.

mod_usertrack

Véase capítulo 11.

Parte II
Administrar su sitio Web

6 Albergar sitios virtuales

En capítulos anteriores hemos aprendido a configurar Apache para que albergue un sitio Web. Según se va familiarizando con la Red y su negocio va cambiando, es posible que necesite algo más que un sitio Web para exponer sus productos, servicios, etc. En estos casos necesitará más de un servidor Apache en los que ubicar esos sitios extra. Es posible que un administrador que lleve poco tiempo en el oficio pensase que lo mejor sería dedicar varias máquinas a este fin, pero aquéllos que ya llevan algún tiempo en el sector verían una solución mejor. Optará por lo que se llama *sitio Web virtual*. En este capítulo aprenderá a configurar sitios Web virtuales con Apache.

¿Necesita un sitio virtual?

Antes de entrar en los detalles relativos a la implementación de un sitio Web virtual en un servidor Apache, es importante determinar si lo necesita o no. En la tabla 6.1 puede ver las ventajas y desventajas de un sitio Web virtual.

Si piensa que elegir un sitio Web virtual es principalmente una decisión económica, la verdad es que no está muy equivocado. De todas formas, hay otros puntos a tener en cuenta (que se suelen aprender con la experiencia). La

mayoría de los administradores Web con experiencia saben que lo normal es que un sitio Web no necesite demasiada CPU ni coma excesivos recursos de sistema (cuando hablo de recursos me refiero a disco o memoria). Por lo tanto, cuando se encuentran con un grupo de sitios típicos, prefieren usar servidores virtuales antes que dedicados. De todas formas, estos últimos se utilizan cuando se espera un gran volumen de tráfico en el sitio Web (incluso se puede llegar a utilizar varios servidores por cada sitio Web).

Tabla 6.1. Ventajas e inconvenientes de un sitio Web virtual

Ventajas	Inconvenientes
Los sitios Web virtuales son más fáciles de manejar en un sistema compuesto por varios ordenadores en los que se ejecuten sitios Web independientes.	Dedicar un servidor por sitio puede aumentar el rendimiento de la máquina, porque cada servidor sólo atenderá las llamadas pertenecientes a un sitio Web determinado. De todas formas, esta mejora en el rendimiento únicamente será efectiva con los sitios Web que estén sometidos a una gran cantidad de tráfico. Si se utiliza un servidor dedicado a un sitio Web con poca densidad de tráfico, no se puede asegurar que se alcance el rendimiento máximo.
Un único conjunto de archivos de configuración. Tan sólo hay que administrar cuatro ficheros diferentes. Lo normal es que no tenga que modificar el archivo de tipos mime, por lo que sólo tendrá que encargarse de tres ficheros: httpd.conf, srm.conf y access.conf.	Hay que administrar los archivos de configuración para cada servidor. Lo normal es que no se tenga que tocar el fichero de tipos mime, pero de todas formas conviene saber que tiene que estar disponible en todos los servidores.
Siempre que se efectúe una actualización de software o de hardware en un servidor en el que se encuentren varios sitios Web virtuales, todos se beneficiarán de las modificaciones.	Cuando se actualiza el sistema de un servidor, únicamente aprovecharán los cambios los sitios Web virtuales que se encuentren dentro de él.
Cuando se utiliza el servidor de una empresa (es decir, un sistema compuesto por varios servidores)	Cuando falla el servidor en el que se alberga un sitio Web, éste se queda completamente inutilizado porque el

Ventajas	Inconvenientes
para albergar uno o más sitios Web virtuales, se acepta cierta tolerancia de errores. Lo normal es que en este tipo de configuración se incluya un sistema DNS y otro NFS.	resto de servidores están dedicados a otros sitios.

La mayoría de los sitios Web son virtuales. Es posible que se haya encontrado numerosos anuncios tratando de vederle un espacio virtual. Muchas organizaciones utilizan este tipo de tratos para aparecer en la Red. De todas formas, antes de que existiesen los sitios Web virtuales, los proveedores (ISP) ofrecían soluciones que no tenían nada de profesional. Por ejemplo, un proveedor que careciese de sitios Web virtuales le podría ofrecer a un cliente que quisiese uno cualquiera de estas tres opciones:

- www.isp.net/cliente o www.isp.net/~cliente, que quería decir que la dirección del sitio Web del ISP formaba parte de la dirección del cliente, con lo que el resultado no era nada elegante ni apropiado.
- http://<dominio del cliente>:<puerto no estándar>/ era otra solución. Por ejemplo, www.glunchy.com:8080/ podría ser el sitio Web de Glunchy, Inc. De todas formas conviene resaltar que se puede utilizar cualquier puerto que no sea el 80 (que es el puerto HTTP por omisión).
- La solución final podría ser el uso de un servidor Apache dedicado para el dominio del cliente.

Se preguntará por qué estoy dedicando tanto tiempo en explicar estas soluciones. Bueno, se lo indico a continuación.

Es posible que algún lector necesite trabajar con una de las soluciones no virtuales que acabamos de ver. Conviene que, antes de decidirse por ninguna, las conozca todas. Por ejemplo, si planea recoger el sitio Web de varios clientes y no tiene nombres personalizados para ellos, puede utilizar http://su.sitio.Web/~cliente como dirección de la página principal de cada uno de ellos. De hecho, éste es el método estándar. La segunda de las opciones que aparecen arriba es buena si desea utilizar el mismo alias IP para otro sitio. Por ejemplo, si tiene que desarrollar un sitio en el que pueda probar nuevas tecnologías que más tarde aplicará a su sitio principal, le conviene esta opción. Y la última opción es la más conveniente para aquellos sitios Web que tengan una gran afluencia de tráfico.

Ahora bien, si su idea es crear un sitio Web virtual que le resulte económico, el resto del capítulo le ayudará a conseguirlo.

Registro del nombre de dominio

El Servicio de Nombre de Dominio (DNS) es el que se encarga de la resolución de los dominios de Internet. Por ejemplo, cuando escribe una dirección como www.idgbooks.com/ en el campo URL de su explorador Web, un servidor DNS (normalmente el servidor DNS de su proveedor de Internet) se pone en contacto con su ordenador para convertir el alias IP www.idgbooks.com en una dirección IP numérica, como 206.80.15.140. Para su sitio Web virtual tendrá que hacer la misma transformación. Esto es lo que veremos en esta sección.

Vamos a intentar aclarar este concepto tan complicado con un ejemplo. Tanto en esta sección como en la siguiente hablaremos de la creación de un sitio Web virtual llamado www.milkyweb.com que se encontrará en un dominio llamado nitec.com. Este dominio tiene varios servidores DNS especializados en resolver todos los nombres de los sitios Web que alberga.

Por ejemplo, si escribe www.nitec.com/ en su servidor Web accedería a las páginas Web de Nitec. Es decir, que el explorador es capaz de ponerse en contacto con su servidor DNS para resolver la dirección www.nitec.com. En pocas palabras, nitec.com está configurado de tal forma que puede albergar otros sitios Web en su servidor www.nitec.com.

El dilema del nombre de dominio

Cada vez es más complicado encontrar un buen nombre de dominio. Parece que todas las palabras en inglés están registradas. Los ejemplos que estamos viendo en estas secciones (nitec.com y milkyweb.com) pertenecen a dos organizaciones reales. La verdad es que cuando me puse a escribir estas líneas me tomé la molestia de buscar un nombre de dominio que no estuviese registrado por nadie y, aunque me llevó bastante tiempo, encontré que milkyweb estaba libre. Pero cuando terminé de escribir este libro me encontré con que ¡ya lo habían registrado! Por eso me tuve que poner en contacto con ellos para pedirles permiso para utilizar su nombre en este ejemplo. Parece que cualquier cosa que se pueda pensar es registrable como nombre de dominio. Lo que ocurre es que mientras lo piensa, alguien se adelanta y lo registra.
Tenga cuidado y escoja un buen nombre de dominio.

Si no va a trabajar con un DNS es posible que prefiera pasar a la sección en la que hablamos de los servidores IP virtuales. Lo normal es que sea el proveedor de servicios de Internet (ISP) el que se encargue de la conversión DNS. El siguiente ejemplo describe qué ocurre cuando no es el caso.

Si milkyweb.com es un dominio que se va a crear, todo lo que tiene que hacer es pasarse por las páginas Web de InterNIC (http://rs.internic.net/), buscar el formulario de registro de dominios y rellenarlo.

Pero para hacerlo tiene que saber cuáles son los nombres del servidor principal y del secundario de nitec.com., ya que serán ellos los que negociarán el nombre del dominio con el servidor DNS. Localizar el dominio de servidores de nombre es algo bastante sencillo. Todo lo que hay que hacer es ejecutar una utilidad TCP/IP llamada whois. En la figura 6.1 se puede ver el resultado obtenido al utilizar el comando whois nitec.com. Whois es un programa que se conecta con la base de datos del dominio InterNIC y solicita la información disponible sobre un dominio.

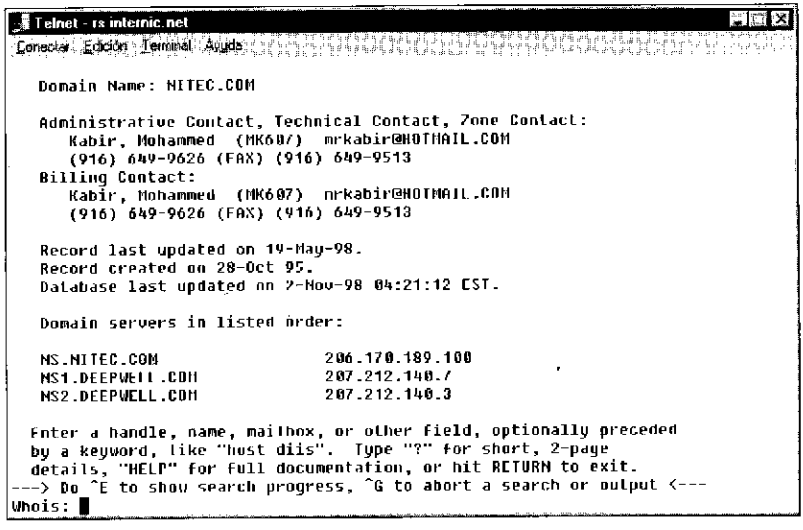


Figura 6.1. Resultado del comando whois nitec.com

Como puede ver, los servidores DNS (ns.nitec.com, macha.s-cc.com, news.s-cc.com) de nitec.com son los que aparecen en la pantalla de salida del comando whois. En la aplicación de registro de dominio tendrá que escribir ns.nitec.com como servidor DNS principal y mach1.s-cc.com como el secundario de milkyweb.com. Una vez que ha completado la aplicación tendrá que configurar el nombre de dominio de Nitec para contestar las preguntas relacionadas con milkyweb.com.

Nota: para crear un dominio nuevo hay que pagar a InterNIC.

Configuración DNS para su sitio Web virtual

Como el servidor www.milkyweb.com tiene una dirección de dominio milkyweb.com (y no nitec.com), tendrá que indicarle a InterNIC (la autoridad del servicio de nombres de dominio) que mande las peticiones dirigidas a milkyweb.com a nitec.com. Se hace en dos pasos: en primer lugar se crea el registro para el dominio en la base de datos correspondiente y, a continuación, se escoge entre una dirección IP o un nombre.

Creación de los registros para el dominio en la base de datos

El primer paso que se ha de dar en la creación de un servidor virtual es construir las bases de datos para el dominio. Puede construir bases de datos nuevas o modificar las ya existentes.

Crear nuevas bases de datos para el dominio

Para que los servidores de nitec.com puedan trabajar con milkyweb.com, nitec.com tendrá que conocer todos los detalles del sitio Web. Mediante dos archivos de texto, milkyweb.db (listado 6.1) y milkyweb.rev (listado 6.2) se pueden crear los registros que necesita el dominio milkyweb.com. Obsérvese que estos archivos usan el nombre BIND del formato del archivo del servidor.

```
@ IN SOA milkyweb.com.   hostmaster.milkyweb.com.(
   19981110001 ; Serial AAAAMMDDXXX
   7200 ; refresh
   3600 ; (1 hour) retry
   1728000 ; (20 days) expire
   3600) ; (1 hr) minimum ttl

; Name Servers
IN NS     ns.nitec.com.
IN NS     machl.s-cc.com.

; A Records
www IN A  206.171.50.50
```

Listado 6.1. milkyweb.db

Aunque un análisis completo de todos los registros DNS es algo que se sale del alcance de esta obra, me gustaría explicar por lo menos los conceptos básicos. El primero de todos es un registro SOA, que se especifica cuando el servidor remoto tiene que actualizar la información con la que trabaja, volver a recogerla o simplemente caducarla. Esta acción permite que el proveedor DNS actualice la información que propaga por Internet de acuerdo con ciertos modelos. En este registro también se guarda la dirección electrónica, pero sin el carácter @ (algo así como hostmaster.milkyweb.com) y un número de serie. La dirección electrónica se guarda para que el administrador del sistema tenga acceso a ella, mientras que el número se registra para clasificar el registro. Yo prefiero usar un formato AAAAMMDDXXX, en el que aparece el Año, el Mes, el Día y dejo tres dígitos para el número de serie. Esta información es la que consultan los servidores DNS para ver si tienen que proceder con la actualización de los datos que tienen guardada en la memoria caché. De todas formas, siempre que se actualiza un registro DNS, se incrementa este número.

El siguiente conjunto de registros es el NS, que informa al servidor DNS de los nombres de los servidores disponibles para un dominio. En este caso, los servidores DNS para milkyweb.com son los mismos que para nitec.com.

Truco: uno de los errores típicos que suelen cometer los administradores DNS es olvidar colocar un punto al final del nombre del servidor (como ns.nitec.com).

El último de todos es un registro A, o un registro de una dirección. Especifica la dirección de cada servidor. Se puede utilizar un formato en el que www llama a www.milkyweb.com, por lo que se necesita toda la configuración previa que le indica al mundo que www.milkyweb.com es un servidor del dominio milkyweb.com, que a su vez pertenece a nitec.com. El nombre del servicio primario es ns.nitec.com.

El procedimiento de configuración de los parámetros DNS no acaba aquí. Aún tendrá que idear una forma para que el resto de los ordenadores pueda invertir el sistema de búsqueda de nombres. Para eso se utiliza el archivo milkyweb.rev que aparece en el listado 6.2.

```
@ IN SOA milkyweb.com.    hostmaster.milkyweb.com.(
   1998110001 ; Serial AAAAMMDDXXX
   7200 ; refresh
   3600 ;(1 hour) retry
   1728000 ;(20 days) expire
  3600) ; (1 hr) minimum TTL
```

```
; Name Servers
IN NS      ns.nitec.com.
; PTR Records
50 IN PTR www.milkyweb.com.
```

Listado 6.2. milkyweb.rev

Este archivo tiene los mismos registros SOA y NS que el que aparece en el listado 6.1. La única novedad es el registro PTR que dirige una dirección IP a un alias IP. En este caso, en el registro PTR sólo se utiliza la parte del servidor (el último octeto de una red de clase C) de la dirección IP (206.171.50.50) para dirigirse a www.milkyweb.com.

Ahora bien, si un servidor de Internet tiene interés en conocer si realmente 206.171.50.50 es www.milkyweb.com, puede invertir la búsqueda del nombre para confirmarlo. Ahora tendrá que usar estos dos archivos en el campo del nombre del dominio del servidor que se encuentra en el fichero de configuración (se guarda en /etc/named.boot). Las líneas que tiene que agregar en el archivo de configuración son las siguientes:

```
primary     milkyweb.com                206.171.in-addr.arpa     milkyweb.db
primary     50.171.206.in-addr.arpa     milkyweb.rev
```

En la primera línea se indica que el servidor DNS (ns.nitec.com) también es un servidor DNS primario para milkyweb.com y que los registros se guardan en el fichero milkyweb.db. En la segunda línea se indica que las direcciones inversas principales están almacenadas en el archivo milkyweb.rev. Obsérvese que las direcciones DNS inversas suelen depender del ISP.

Una vez que se han añadido estas líneas al archivo /etc/named.boot, se tiene que reiniciar el proceso del nombre del servidor para que se puedan leer dichos archivos. Para ello se utiliza el comando:

```
/usr/sbin/named.restart
```

Generalmente, para efectuar esto que acabamos de ver hay que ser un usuario root. En realidad no es otra cosa que una llamada a un script que ejecuta el comando /usr/sbin/ndc con un argumento llamado restart (reinicio). El programa ndc es una interfaz para el servidor de nombres. El software de este servidor puede tener un programa aparte o una opción para reiniciarlo. Consulte las páginas de ayuda. Una vez que se ha reiniciado el servidor, habrá que probarlo. En la figura 6.2 puede ver que escribí:

```
ping www.milkyweb.com
```

en la línea de comandos y obtuve la siguiente respuesta del servidor.

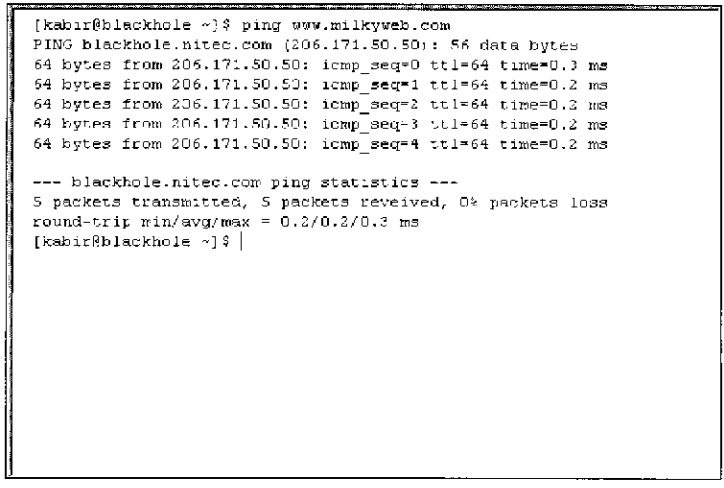


Figura 6.2. Ping a www.milkyweb.com

La prueba podría ser suficiente para convencerme de que la configuración DNS funciona. De todas formas, habría que ver una prueba más robusta porque, en caso de tener un error de configuración, el comando ping se limitaría a fallar sin proporcionar ningún tipo de información. Por eso vamos a ver una utilidad del servidor llamada nslookup.

Esta utilidad se encarga de buscar al servidor de nombres. En la figura 6.3 se puede ver una sesión nslookup que establecí para ver si la configuración DNS anterior era correcta.

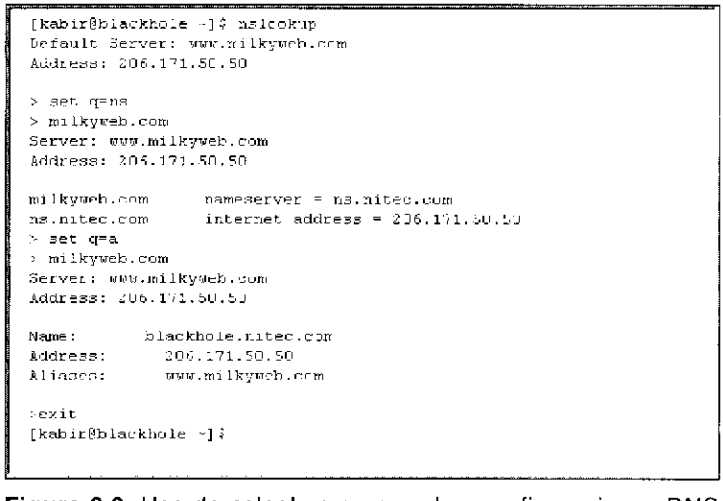


Figura 6.3. Uso de nslookup para probar configuraciones DNS

Para ejecutar esta comprobación hay que escribir en la línea de comandos lo siguiente:

```
nslookup
```

Así abrirá una sesión con el prompt > en pantalla. También se mostrará el servidor DNS que se está utilizando para efectuar peticiones. Si dicho servidor no es aquel en el que se determina la nueva configuración, habrá que utilizar un comando como el que aparece a continuación para indicarle a nslookup que le indique al servidor cuál es su configuración:

```
server <nombre_servidor>
```

En el caso que nos ocupa el nombre del servidor es ns.nitec.com, para el que ya se han modificado los parámetros DNS. Por eso mismo no me ha hecho falta escribir el comando anterior. Si observa la figura 6.3 verá que el primer grupo de comandos que he escrito es:

```
set q=ns
milkyweb.com
```

El primero le indica a nslookup que registre el resultado de la petición en un registro NS (name server, nombre del servidor). La siguiente se limita a decirle que busque milkyweb.com como registro del servidor de nombres. En la salida de dicho comando se podrá ver que ns.nitec.com informa que el servidor DNS de milkyweb.com es ns.nitec.com. Ahora vamos a fijarnos en la segunda línea de comandos:

```
set q=a
www.milkyweb.com
```

El primer comando le dice al programa que escriba la petición en un registro A (address, dirección) y que le solicite a ns.nitec.com la dirección de www.milkyweb.com. En la salida se puede ver que la dirección del alias IP es 206.171.50.50.

Por último, se sale del programa utilizando el comando:

```
exit
```

Llegados a este punto nos encontramos con que la configuración del servidor DNS está completa. De todas formas aún hay que configurar el servidor DNS secundario. Para ello se puede añadir la siguiente línea en los archivos /etc/named.boot de cada servidor:

```
secondary          milkyweb.com                  milkyweb.db
secondary          50.171.206.in-addr.arpa       milkyweb.rev
```

Obsérvese que en vez de utilizar el siguiente registro A:

```
www        IN     A          206.171.50.50
```

puede utilizar el registro CNAME:

```
www        IN     CNAME      blackhole.nitec.com
```

¿Dónde está la diferencia? La verdad es que no hay mucha, puesto que 206.171.50.50 es la verdadera dirección de blackhole.nitec.com. De todas formas, si no se utiliza un registro A se conseguiría que el DNS de milkyweb.com fuese más manejable porque eliminaría la necesidad de actualizar el registro DNS correspondiente cada vez que se modifica la dirección IP del servidor de blackhole.nitec.com. También es verdad que el registro A se utiliza con los sitios Web virtuales basados en una dirección IP, mientras que CNAME se utiliza con los que se basan en un nombre IP. Ya los veremos con más detalle al final del capítulo.

Modificación de las bases de datos ya existentes

De momento, hemos hablado de crear un nombre de un servidor virtual para un grupo de dominios (www.milkyweb.com). Si ya dispone de una red del tipo www.suempresa.com y quiere crear un sitio virtual que se llame intranet.suempresa.com, todo lo que tendrá que hacer es crear un alias en su servidor DNS para www.suempresa.com utilizando el registro CNAME:

```
intranet   IN     CNAME      www
```

En su base de datos de direcciones DNS inversas querrá duplicar la entrada para www.suempresa.com y sustituir la parte www por **intranet**.

Por ejemplo, si escribo lo siguiente:

```
apache     IN     CNAME      blackhole          ; in nitec.db file

50         IN     PTR        apache.nitec.com   ; in nitec.rev file
```

Ahora, cuando utilizo nslookup, como se puede apreciar en la figura 6.4, para ver qué dirección tiene el servidor apache.nitec.com, me encuentro con que muestra la dirección IP de la máquina blackhole.nitec.com.

Cuando se utiliza un servidor virtual, hay que usar la configuración anterior, pero si desea trabajar con un servidor virtual que se base en la dirección

IP tendrá que utilizar un registro A. Por ejemplo, para llegar al mismo final puedo configurar el mismo servidor apache.nitec.com:

```
apache      IN    A          206.171.50.50     ; in nitec.db file
50          IN    PTR        apache.nitec.com  ; in nitec.rev file
```

De todas formas se puede utilizar esta versión con los servidores virtuales que se basen en direcciones IP.

```
[Kabir@blackhole ~]$ nslookup
Default Server: www.milkyweb.com
Address: 206.171.50.50

> set q=a
> apache.nitec.com
Server:  www.milkyweb.com
Address:  206.171.50.50

Name:      blackhole.nitec.com
Address:   206.171.50.50
Aliases:   apache.nitec.com

>
```

Figura 6.4. Uso de nslookup para comprobar el registro de modificaciones del dominio

Decidir entre basarse en una dirección IP o un nombre IP

El siguiente paso que hay que dar para crear un sitio Web virtual es decidir si se desea trabajar con un servidor basado en una dirección IP o en un nombre IP. Lo veremos en esta sección.

Servidores virtuales basados en una dirección IP

Un servidor virtual basado en una dirección IP implica que le asigna una dirección IP a cada servidor. En la figura 6.5 se puede ver el esquema de esta configuración, donde cada servidor virtual tiene su propia dirección IP y estas direcciones se envían a una sola máquina.

Utilizar una única dirección IP para sitio Web virtual implica que se ha de crear un registro para cada sistema. Por ejemplo, la configuración que apare-

ce en la figura 6.5 podría ser el resultado de una configuración DNS similar a ésta:

```
; La siguiente entrada se dirige a la base de datos DNS de nitec.com
(ref hyperlink http:www.nitec.com)          IN   A 207.212.140.213
; La siguiente entrada se dirige a la base de datos DNS de
client1.com
(ref hyperlink http:www.client1.com         IN   A 207.212.140.214
; esta entrada se dirige a la base de datos DNS de client2.com
(ref hyperlink http:www.client2.com         IN   A 207.212.140.215
```

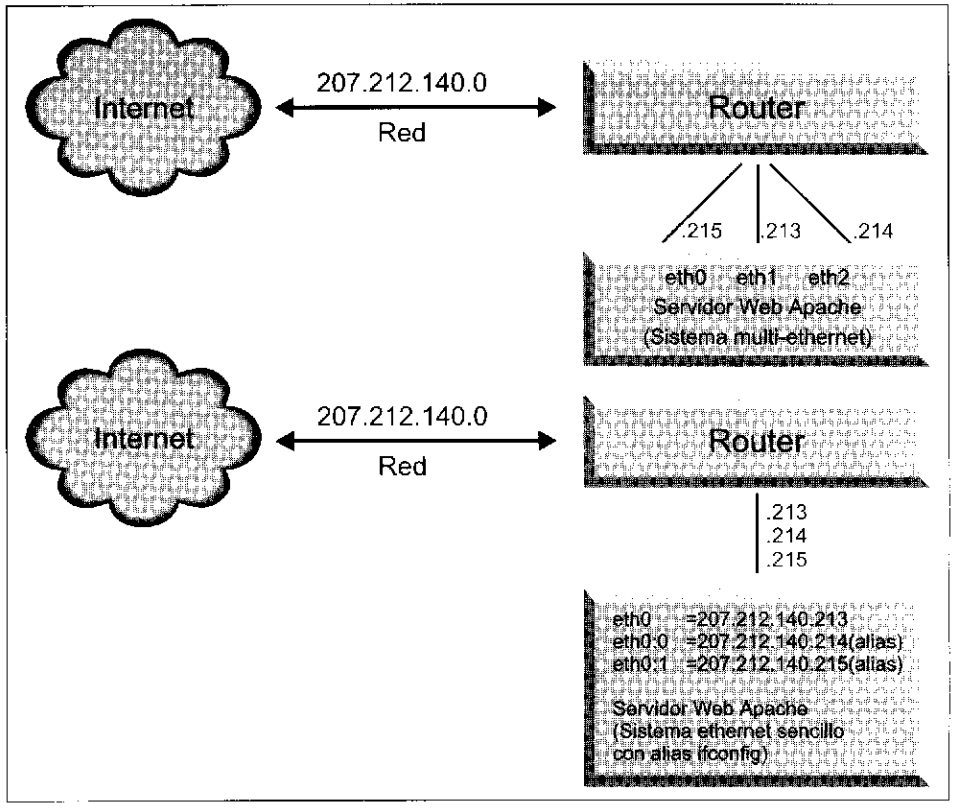


Figura 6.5. Uso de una única dirección IP para cada sitio virtual

También tiene que configurar correctamente los routers para que se asocien adecuadamente todas las direcciones IP que quiera usar con los servidores Web (virtuales y principal). Dependiendo del sistema operativo que utilice en su servidor Web principal, podrá escoger entre crear alias IP para cada interfaz Ethernet o disponer de distintas interfaces para cada dirección IP (generalmente se añaden tarjetas Ethernet adicionales al sistema).

En el siguiente ejemplo veremos cómo se pueden utilizar los alias para enlazar varias direcciones IP a un único dispositivo Ethernet. Se usa el sistema operativo Linux y la utilidad ifconfig, que puede trabajar con alias. Si utiliza un sistema operativo diferente, consulte el manual de ifconfig para ver cuáles son las opciones de alias.

Truco: los sistemas operativos más populares, como Linux, BSDI, Digital UNIX, FreeBSD, HPUX 10.x y AIX 4.1, disponen de opciones de alias para trabajar con ifconfig. De todas formas, es posible que se encuentre con sistemas operativos a los que tendrá que añadir un parche para que puedan trabajar con esta utilidad.

En la figura 6.6 se muestra la salida que se obtiene al ejecutar el comando ifconfig desde la cuenta del usuario root.

```
[kabir@blackhole kabir]# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING MTU:3584 Metric:1
          RX packtes:58684 errors:0 dropped:0 overruns:0
          TX packets:58684 errors:0 dropped:0 overruns:0

eth0      Link encap:10Mps Ethernet  HWaddr 00:C0:F6:98:37:37
          inet addr:206.171.50.5C  Bcast:206.171.50.63  Mask:255.255.255.240
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packtes:353979 errors:0 dropped:0 overruns:0
          TX packets:353379 errors:_ dropped:0 overruns:0
          Interrupt:5 Base address:0x340

[kabir@blackhole kabir]#
```

Figura 6.6. Salida de ifconfig antes de utilizar los alias

Como se puede ver en la figura se han configurado dos dispositivos para que utilicen ifconfig: el encargado de volver atrás y el Ethernet predeterminado eth0. Ahora, para añadir un alias IP para el dispositivo eth0 tiene que utilizar el comando ifconfig de la siguiente manera:

```
ifconfig device:N IP_address
```

Aquí, N es 0, 1, 2, 3, etc. Por ejemplo, para crear un alias para eth0 utilizando la dirección IP 206.171.50.50 habría que escribir el siguiente comando:

```
ifconfig eth0:0 206.171.50.50
```

Al ejecutar el comando ifconfig nos encontraríamos con una salida similar a la que aparece en la figura 6.7.

```
[kabir@blackhole kabir]# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          RX packets:58685 errors:0 dropped:0 overruns:0
          TX packets:58685 errors:0 dropped:0 overruns:0

eth0      Link encap:10Mps Ethernet  HWaddr 00:C0:F6:98:37:37
          inet addr:206.171.50.50  Bcast:206.171.50.63  Mask:255.255.255.240
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:354063 errors:0 dropped:0 overruns:0
          TX packets:354063 errors:0 dropped:0 overruns:0
          Interrupt:5 Base address:0x340


eth0:0    Link encap:10Mps Ethernet  HWaddr 00:C0:F6:98:37:37
          inet addr:206.171.50.52  Bcast:206.171.50.255  Mask:255.255.255.0
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
          TX packets:0 errors:0 dropped:0 overruns:0

[kabir@blackhole kabir]#
```

Figura 6.7. Salida de ifconfig antes de utilizar los alias

También hace falta un router para el alias de la dirección IP:

```
router add -host 206.171.50.50 dev eth0:0
```

Al añadir estos dos comandos al archivo rc.local que se encuentra en el directorio /etc/rc.d se asegura de que este alias esté correctamente configurado para cuando se inicia el servidor.

Una dirección IP es un recurso muy apreciado en Internet porque no hay muchas disponibles. Hasta que los nuevos sistemas se utilicen con total normalidad (IPNG o IP Next generation), cada vez va a ser más complicado conseguir una dirección IP. InterNIC, la autoridad en lo que a direcciones IP se refiere, no entrega ninguna dirección a nadie. De hecho, recomienda que se soliciten directamente al proveedor de servicios de Internet, con lo que se tendrá que abonar una suma de dinero para que se asigne una serie de direcciones IP a un sistema como puede ser su servidor Web.

Compatibilidad con los exploradores anteriores a HTTP/1.1

Estas acciones que acabamos de ver implican el que se utilice la directriz ServerPath en la configuración del primer servidor virtual. Por ejemplo:

```
NameVirtualHost 111.222.333.444

<VirtualHost 111.222.333.444>
# primary virtual host
DocumentRoot /www/subdomain
...
</VirtualHost>

<VirtualHost 111.222.333.444>
DocumentRoot /www/subdomain/client1
ServerName www.client1.com
ServerPath /client1/
...
</VirtualHost>

<VirtualHost 111.222.333.444>
DocumentRoot /www/subdomain/client2
ServerName www.client2.com
ServerPath /client2/
...
</VirtualHost>
```

Es posible que, en la página en la cual aparece el índice del primer servidor virtual (/www/subdomain/index.html), tenga que añadir un prefijo URL al nombre de los servidores virtuales. También tendrá que localizar el nombre de los servidores virtuales que se encuentran en el directorio raíz del primer servidor virtual, cada uno con su propio subdirectorio. En el ejemplo anterior, el directorio raíz para el documento del primer servidor virtual es /www/subdomian/. www.client1.com y www.client2.com tienen sus documentos guardados en los subdirectorios /www/subdomain/client1 y /www/subdomain/client2 respectivamente. Lo último que tiene que añadir a este sitio Web virtual es una lista con todos los enlaces relativos (por ejemplo, archivo.html o /icons/image.gif) o bien de los enlaces que tengan el prefijo /dominio/ (como por ejemplo, www.dominio.tld/dominio/misc/archivo.html o bien /dominio/misc/archivo.html).

La directriz ServerPath envía una petición al URL www.client1.com/client1/ para que sea este servidor el que le atienda. Las peticiones enviadas a www.client1.com únicamente las atenderá el servidor client1 si se envía la cabecera Host: correcta utilizando HTTP/1.1. En caso contrario el cliente recibirá una página de información procedente del primer servidor virtual.

Esta solución puede parecer algo engorrosa y difícil de manejar, pero la verdad es que de esta forma se asegurará de que sus páginas funcionen con la mayoría de exploradores Web, tanto nuevos como antiguos.

Servidores virtuales basados en un nombre IP

Debido a la gran aceptación que ha tenido el protocolo HTTP/1.1, ahora se pueden utilizar otros métodos para crear servidores Web virtuales utilizando sistemas IP. Así se puede utilizar la misma dirección IP para varios nombres de servidores.

Antes de que apareciese HTTP/1.1 un servidor Web no tenía ninguna forma de conocer el nombre del servidor virtual que utilizaba el explorador Web con el servicio de peticiones. La nueva especificación de HTTP/1.1 añade esta propiedad por la que el explorador le tiene que decir al servidor el nombre del host con el que está trabajando. Para eso se utiliza la cabecera Host:. Si alguien utiliza un explorador que no sea compatible con HTTP/1.1, no enviará la información relacionada sobre el host, con lo que el servidor no será capaz de localizar el sitio Web apropiado.

Para poder trabajar con los servidores (hosts) virtuales que no admiten direcciones IP, todo lo que tiene que hacer es disponer los registros CNAME apropiados en las bases de datos DNS correspondientes gracias a los cuales se apuntará al host que se indique en un registro A. Por ejemplo, si me quiero dirigir desde los servidores www.client1.com y www.client2.com a www.nitec.com, tendrá que modificar las entradas de las bases de datos DNS de client1.com y client2.com y añadir lo siguiente:

```
; La siguiente entrada se dirige a la base de datos DNS de client1.com
{ref hyperlink http:www.client1.com}  IN  CNAME  webserver.nitec.com.
; La siguiente entrada se dirige a la base de datos DNS de client2.com
{ref hyperlink http:www.client2.com}  IN  CNAME  webserver.nitec.com.
```

En esta sección hemos visto dos formas de configurar el DNS de un sitio Web virtual. Basándose en lo que se ha visto aquí ya debería ser capaz de decidir si prefiere trabajar con sitios Web virtuales basados en una dirección IP o en un nombre IP.

Ahora vamos a configurar Apache para que ofrezca servicios con la Red a los servidores virtuales.

Configurar Apache para que trabaje con servidores virtuales

Puede configurar Apache de dos formas distintas para que pueda trabajar con varios servidores (el principal y los virtuales). Una opción es ejecutar varios demonios de tal modo que cada servidor tenga su propio demonio httpd.

La otra es ejecutar uno solo que se encargará de todos los servidores (del principal y de los virtuales). En la figura 6.8 se pueden ver ambas opciones.

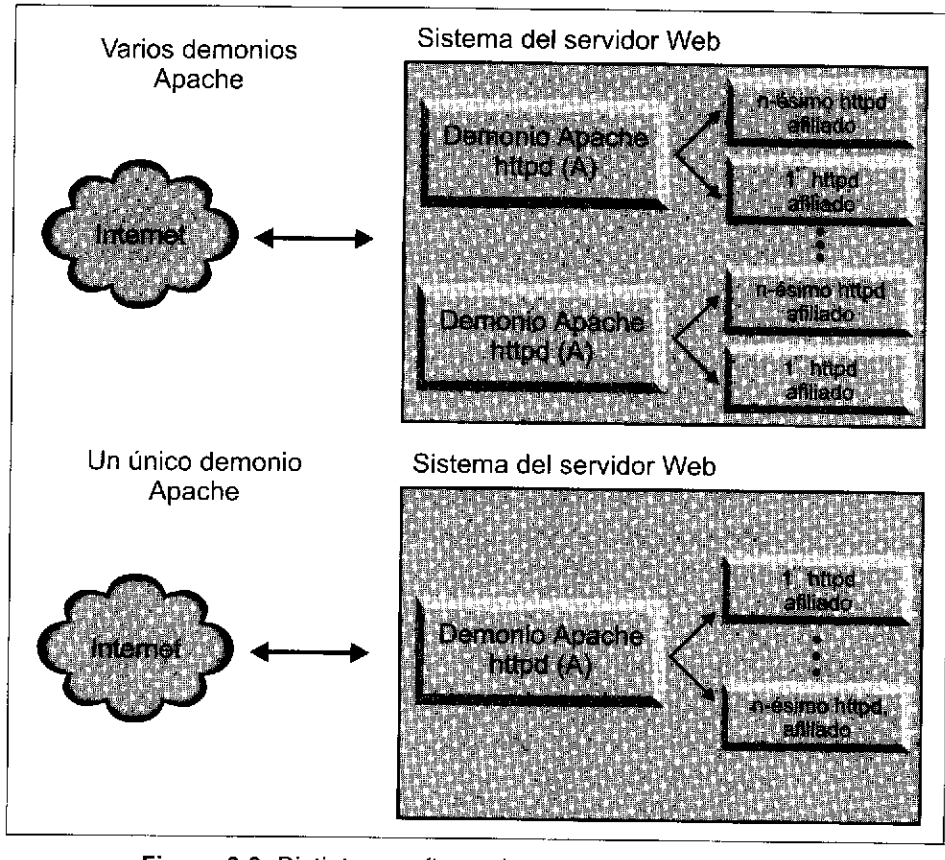


Figura 6.8. Distintas configuraciones de demonios Apache

En la figura se puede ver que en la primera configuración se trabaja con varios servidores primarios, mientras que en la segunda tan solo hay uno. En ambos métodos, todos los procesos afiliados abren el servicio de peticiones del servidor principal. Por ejemplo, si una petición proviene del servidor A que se encuentra en la configuración de varios servidores, generará un proceso afiliado y atenderá a la petición. En la configuración de un único servidor primario, éste generará un proceso afiliado y se encargará del servicio de peticiones. La diferencia entre ambas configuraciones es que con la segunda se utiliza un mismo servidor para que se haga cargo de todo el trabajo.

Para las necesidades típicas es posible que se prefiera utilizar un único demonio Apache primario. De todas formas, si no se puede compartir la confi-

guración httpd entre servidores por diferencias en el tipo de servidor (ServerType), usuario, grupo, tipo de configuración (TypesConfig), ServerRoot, etc., tendrá que ejecutar varios procesos Apache. Es posible que para efectuar esta separación de los procesos httpd haya que utilizar archivos de configuración independientes.

Vamos a ver cómo se puede configurar cada una de las distintas opciones: configuración de varios demonios y de uno sólo.

Configurar varios demonios Apache

Para configurar varios demonios Apache lo primero que tiene que hacer es crear una instalación httpd independiente para cada servidor virtual. Básicamente, hay que crear una serie de archivos de configuración para cada demonio de proceso. Utilice la directriz Listen en el fichero de configuración para seleccionar las direcciones IP (o los servidores virtuales) de los servicios de los distintos demonios.

Hay dos formas de indicarle a Apache las direcciones o los puertos a los que tiene que prestar atención: la directriz BindAddress, con la que se especifica una única dirección o un puerto; la directriz Listen, con la que se puede especificar un conjunto de direcciones y puertos. Por ejemplo, si se ejecuta el servidor principal en la dirección IP 206.171.50.50 y en el puerto 80, y el virtual en 206.171.50.54, puerto 8080, tendría que escribir lo siguiente:

```
Listen 206.171.50.50:80
Listen 206.171.50.54:8080
```

Truco: en la directriz Listen es preferible utilizar direcciones IP en vez de nombres IP, ya que de esta forma se reduce la necesidad de convertir dichos nombres en direcciones IP durante el proceso de arranque del servidor.

Obsérvese que al utilizar varios servidores principales con los sitios Web virtuales se carga considerablemente el tráfico del sistema. Por esto mismo no es conveniente utilizarlo con sitios en los el tráfico ya sea denso de por sí.

Configurar un único demonio Apache

Es la configuración predeterminada de los servidores Apache. Un único demonio httpd se encarga de abrir los procesos afiliados para que se hagan

cargo del servicio de peticiones del sitio Web principal y todos los virtuales que tenga asociados. Por defecto, Apache escucha todas las direcciones IP que llegan a través del puerto 80 a la máquina local. A menudo es insuficiente. En el momento en que se tiene algún requisito algo más complicado, como por ejemplo, tener que escuchar a través de varios puertos o centrarse únicamente en una serie de direcciones IP, habrá que usar las directrices BindAddress y Listen.

Vamos a ver cómo se puede configurar el servidor Apache para que utilice un único demonio de proceso para trabajar con todos los sitios Web virtuales.

Apache utiliza un contenedor especial <VirtualHost>, que se encuentra en el archivo httpd.conf, para controlar las configuraciones propias del servidor. Un ejemplo de una configuración mínima para un servidor virtual podría ser algo así:

```
<VirtualHost 206.171.50.50>
DocumentRoot /www/apachebible/public/htdocs
ServerName  www.apachebible.com
</VirtualHost>
```

La primera línea marca el inicio de la configuración del servidor virtual (www.idgbooks.com). La dirección IP determinada tiene que ser una de las validadas para www.idgbooks.com. Si hubiese que utilizar un puerto no estándar (es decir, otro distinto al 80), habría que escribir lo siguiente:

```
<VirtualHost IP:port>
```

Por ejemplo:

```
<VirtualHost 206.171.50.50:8080>
```

Advertencia: es posible que se pregunte si puede utilizar nombres IP en vez de direcciones IP dentro del contenedor <VirtualHost>. Sí, sí que puede, pero no lo haga hasta que los desarrolladores de Apache le den el visto bueno. Si utiliza el nombre del servidor en vez de su dirección IP, Apache tendrá que hacer una conversión DNS para determinar la dirección IP del servidor. Si por alguna razón fallase esa comprobación, el servidor virtual no estaría operativo hasta que no se reiniciase Apache. Al igual que a mucha otra gente, no me gusta guardar direcciones IP dentro de mis archivos de configuración, pero hasta que los desarrolladores de Apache no nos mejoren las cosas, habrá que seguir utilizándolas.

Cualquier directriz que se encuentre dentro del contenedor <VirtualHost> sólo se aplicará al servidor virtual. Cualquier directriz que se utilice fuera del contenedor <VirtualHost> formará parte de la configuración principal del servidor. Cada servidor virtual toma la configuración del servidor principal a no ser que surja algún conflicto. En caso de que se utilice la misma directriz en la configuración del servidor principal y en la del virtual (es decir, dentro del contenedor <VirtualHost>), la directriz del servidor virtual tiene preferencia sobre la del servidor principal, pero sólo en la parte de la configuración perteneciente al servidor virtual. Por ejemplo, si en la configuración del servidor principal el valor de la directriz ServerName es www.suempresa.com (es decir, fuera del contenedor <VirtualHost>) y dentro de <VirtualHost> tiene otra directriz ServerName configurado a www.sucliente.com, entonces parece lógico pensar que se desea que el servidor virtual responda a www.sucliente.com. Y es exactamente esto lo que ocurre. La directriz que se encuentra en la sección del servidor virtual anula la directriz correspondiente que se encuentra dentro de la sección del servidor virtual. Es más fácil de comprender si se mira la configuración del servidor principal como la configuración global de todos los servidores que se crean utilizando <VirtualHost>. Por eso, al configurar los servidores virtuales, tiene que decidir qué cambios hay que aplicar en la configuración virtual de cada uno de ellos.

Las directrices sobrescriben la configuración del servidor principal o la de un suplemento, dependiendo de la directriz utilizada. Por ejemplo, DocumentRoot es una directriz que si, aparece en la sección <VirtualHost>, anula la que se encuentra en la configuración del servidor principal, mientras que AddType complementa los tipos MIME definidos en la configuración del servidor principal.

Ahora bien, cuando se recibe una petición, Apache utiliza las direcciones IP y el puerto por el que se ha recibido para localizar la configuración de la máquina virtual. Si ningún servidor coincide con la dirección ni el puerto, se utiliza la configuración del servidor principal. Si coincide con la dirección de un servidor virtual, Apache utiliza la configuración de dicho servidor para hacerse cargo de la petición.

En el ejemplo anterior, la configuración del servidor virtual que se utiliza es la misma que la del servidor principal, exceptuando que DocumentRoot está configurada a /www/apachebible/public/htdocs y que ServerName es www.apachebible.com. Las directrices que suelen aparecer en los contenedores <VirtualHost> son DocumentRoot, ServerName, ErrorLog y TransferLog.

La verdad es que se puede poner cualquier directriz dentro de VirtualHost, con la salvedad de ServerType, StartServers, MaxSpareServers, MinSpareServers, MaxRequestPerChild, BinAddress, Listen, PidFile, TypesConfig,

ServerRoot y NameVirtualHost. User y Group se pueden utilizar dentro de los contenedores virtuales, pero sólo si se está usando suEXSC.

Puede tener tantos contenedores <VirtualHost> como quiera. También puede hacer que el servidor principal se encargue de controlar a los servidores virtuales o dejar que un <VirtualHost> disponga de todas las direcciones y puertos y que el servidor principal no controle ninguna petición.

Llegados a este punto, se puede preguntar cómo es posible que Apache sepa cuál es la configuración del servidor virtual que tiene que utilizar si varios comparten la misma dirección IP. Bueno, es lo suficientemente inteligente como para mirar la directriz ServerName para ver si su contenido coincide con el de la cabecera Host: que envían los exploradores Web compiladores de HTTP/1.1. Por medio de esta comparación sabrá a qué servidor virtual se trata de acceder y se lo sirve al cliente.

Tenga en cuenta que la versión 1.3 de Apache (o anteriores) requiere la presencia de una directriz adicional llamada NameVirtualHost antes de <VirtualHost>. En dicha directriz se especifica la configuración del servidor virtual basado en el nombre IP. Por ejemplo:

```
NameVirtualHost 206.171.50.50

<VirtualHost 206.171.50.50>
ServerName www.client1.com
DocumentRoot /www/client1
</VirtualHost>

<VirtualHost 206.171.50.50>
ServerName www.client2.com
DocumentRoot /www/client2
</VirtualHost>

NameVirtualHost 206.171.50.56

<VirtualHost 206.171.50.56>
ServerName www.client3.com
DocumentRoot /www/client3
</VirtualHost>
```

La directriz NameVirtualHost especifica la dirección IP que se tiene que utilizar con el servidor virtual. En el ejemplo anterior hay dos servidores virtuales, www.client1.com y www.client2.com, que usan la misma dirección IP (206.171.50.50). El tercer servidor virtual, www.client3.com, tiene una dirección diferente (206.171.50.56), por lo que la segunda directriz <VirtualHost> se coloca justo antes de la configuración del servidor virtual. Todo lo que necesita es una directriz NameVirtualHost para todas las configuraciones no-IP de servidores virtuales, en caso de agruparlos como aparecen en este ejemplo.

Una vez que se ha creado la configuración del servidor virtual para Apache puede iniciar el servidor para que su configuración tenga efecto. No se olvide de crear una estructura de directorios antes de dar este paso final.

Según hemos mencionado con anterioridad, algunos clientes no envían los datos requeridos, por lo que el servidor Apache no podrá trabajar correctamente. A estos clientes siempre se les envían las páginas del primer servidor virtual que aparece en el archivo de configuración de una dirección IP determinada. En la siguiente sección, veremos alguna de las soluciones que se utilizan para evitar este problema.

Ejemplos de servidores virtuales

Ahora que ya sabe cómo configurar un servidor DNS y un Apache para que trabajen con su sitio virtual, veremos unos cuantos ejemplos que le ayudarán a comprender mejor los distintos aspectos relacionados con los sitios Web virtuales.

Nota: las direcciones (111.222.333.x) que se utilizan en los siguientes ejemplos no son válidas intencionadamente.

Albergar un servidor Web con varias direcciones IP

En este ejemplo vamos a configurar un servidor Web virtual que se basa en una dirección IP, www.micliente.com, y que se encuentra en un host llamado www.miempresa.com. Este host necesitará un servidor llamado www.miempresa.com. En la figura 6.9 se puede ver un diagrama de lo que se verá en este ejemplo. Como se puede apreciar, el servidor Web tiene dos direcciones IP con el siguiente registro DNS:

```
www.mywebserver.com     IN  A       111.222.333.1
www.myclient.com        IN  A       111.222.333.2
www.mycompany.com       IN  CNAME   webserver.mycompany.com.
```

El archivo de configuración httpd.conf (simplificado) tendría este aspecto:

```
...
Port 80
DocumentRoot /www/mycompany
```

```
ServerName www.mycompany.com

<VirtualHost 111.222.333.2>
DocumentRoot /www/myclient
ServerName www.myclient.com
...
</VirtualHost>
```

Sólo se puede acceder al host www.micliente.com a través de la dirección 111.222.333.2, mientras que al servidor www.miempresa.com únicamente se accederá a través de la dirección 111.222.333.1.

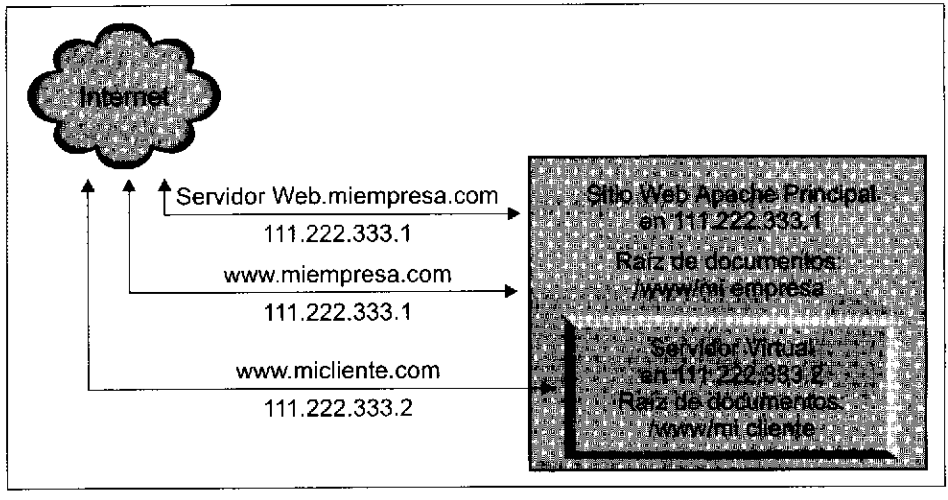


Figura 6.9. Ejemplo de un host virtual basado en una dirección IP en un servidor con varias direcciones IP

Albergar un servidor host basado en una dirección IP sin un servidor principal

Este ejemplo es muy parecido al anterior; la diferencia es que tan sólo los usuarios locales (es decir, los usuarios que ejecutan sus exploradores Web en la máquina www.miempresa.com) pueden ver las páginas del servidor principal. En otras palabras, no tienen un servidor principal dedicado a ellos. En la figura 6.10 se muestra el diagrama lógico para este ejemplo.

El archivo de configuración httpd.conf (simplificado) tendría este aspecto:

```
...
Port 80
ServerName webserver.mycompany.com
DocumentRoot /www/localinfo
```

```
<VirtualHost 111.222.333.1>
DocumentRoot /www/mycompany
ServerName www.mycompany.com
...
</VirtualHost>

<VirtualHost 111.222.333.2>
DocumentRoot /www/myclient
ServerName www.myclient.com
...
</VirtualHost>
```

En esta configuración, el servidor principal tan solo puede atender las peticiones de los usuarios locales que están trabajando con exploradores Web desde la misma máquina. De esta forma, cuando un usuario local accede a http://servidorlocal/, tendría entonces que poder acceder a las páginas del servidor virtual que se encuentran en /www/infolocal. Las llamadas dirigidas a www.miempresa.com/ y www.micliente.com/ las atenderán los respectivos servidores virtuales.

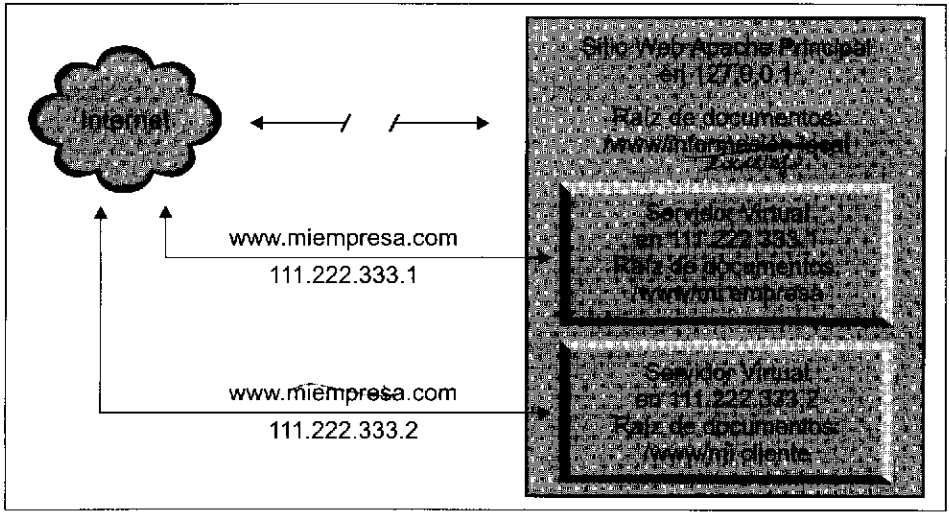


Figura 6.10. Ejemplo de un host virtual basado en una dirección IP sin un servidor principal

Host virtuales basados en una dirección IP y con diferentes puertos

En este ejemplo se utiliza un mismo host visto en los dos casos anteriores, pero esta vez intentaré mostrarle cómo se puede configurar Apache para

disponer de un host virtual que escuche a través de un puerto y una segunda máquina para que haga lo propio con otro puerto distinto. En la figura 6.11 se muestra el diagrama correspondiente a este ejemplo.

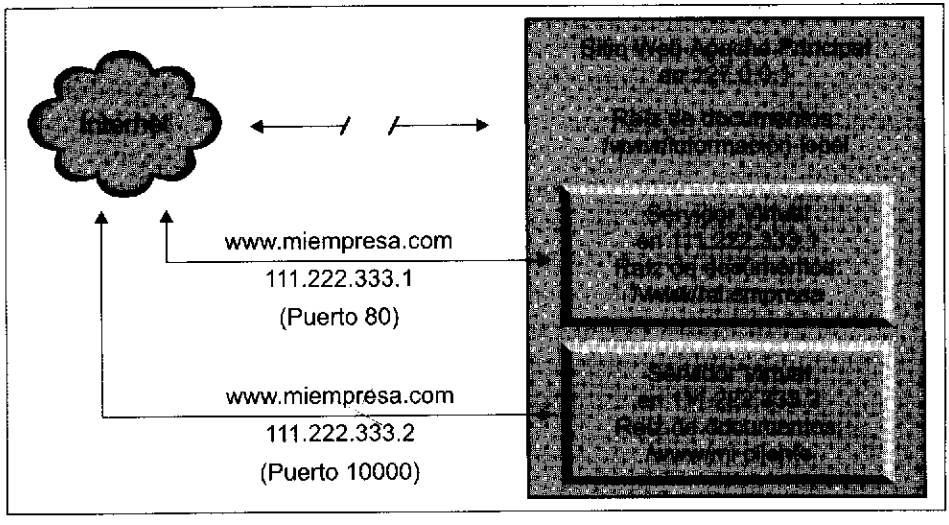


Figura 6.11. Ejemplo de un host virtual basado en una dirección IP con puertos diferentes

El archivo de configuración httpd.conf (simplificado) tendría este aspecto:

```
...
Listen 111.222.333.1:80
Listen 111.222.333.2:10000
ServerName webserver.mycompany.com
DocumentRoot /www/localinfo

<VirtualHost 111.222.333.1>
DocumentRoot /www/mycompany
ServerName www.mycompany.com
...
</VirtualHost>

<VirtualHost 111.222.333.2>
DocumentRoot /www/myclient
ServerName www.myclient.com
...
</VirtualHost>
```

Como en el ejemplo anterior, el servidor principal únicamente se hará cargo de las peticiones efectuadas por los usuarios locales. Se puede acceder al servidor www.miempresa.com a través de un puerto estándar (los URL váli-

dos son www.miempresa.com/ y www.miempresa.com:80/). De todas formas, al sitio www.micliente.com sólo se podrá acceder a través del puerto 10000. Por lo tanto, habrá que utilizar la siguiente URL:

```
http://www.miempresa.com:10000/
```

Vamos a ver la configuración de un ejemplo que se base en el nombre IP.

Servidores basados en un nombre IP

Este ejemplo le muestra cómo configurar un host basado en IP, servidorweb.miempres.com para albergar dos servidores virtuales basados en el nombre. Como puede ver, el sistema del servidor Web tiene una única dirección IP dentro del registro DNS:

```
www.mywebserver.com     IN  A       111.222.333.1
www.mycompany.com       IN  CNAME   webserver.mycompany.com.
www.myclient.com        IN  CNAME   webserver.mycompany.com.
```

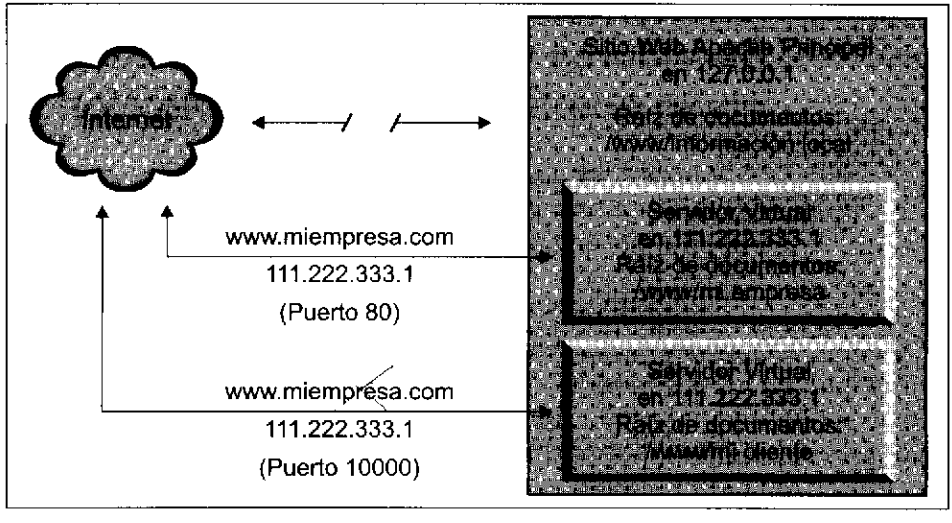


Figura 6.12. Ejemplo de un servidor basado en un nombre

Tanto www.miempresa.com como www.micliente.com son dos alias (registros CNAME) de servidorweb.miempresa.com.

La configuración simplificada de httpd.conf tendría este aspecto:

```
...
Port 80
ServerName webserver.mycompany.com
```

```
NameVirtualHost 111.222.333.1

<VirtualHost 111.222.333.1>
DocumentRoot /www/mycompany
ServerName www.mycompany.com
...
</VirtualHost>

<VirtualHost 111.222.333.1>
DocumentRoot /www/myclient
ServerName www.myclient.com
...
</VirtualHost>
```

El servidor principal tan sólo trabaja con los usuarios locales del sistema. El primer servidor virtual será el que vean los usuarios que tengan exploradores Web que no trabajen con HTTP/1.1. Los que trabajan con exploradores que comprendan dicho protocolo podrán dirigirse al servidor adecuado. Y éste les atenderá de acuerdo con lo que aparezca en su archivo de configuración. Vamos a ver ahora un ejemplo de un servidor virtual (basado en un nombre IP y en una dirección IP).

Servidores virtuales mixtos basados en un nombre IP y en una dirección IP

Vamos a ver un ejemplo en el que los servidores virtuales tienen una configuración mixta, ya que están basados en el nombre IP y en la dirección IP. Por ejemplo, he utilizado los siguientes registros DNS:

```
www.mywebserver.com     IN  A       111.222.333.1
www.myclient.com        IN  A       111.222.333.2
www.mycompany.com       IN  CNAME   webserver.mycompany.com.
www.mycompany.com       IN  CNAME   webserver.mycompany.com.
```

La configuración simplificada de httpd.conf tendría este aspecto:

```
...
Port 80
ServerName webserver.mycompany.com
NameVirtualHost 111.222.333.1

<VirtualHost 111.222.333.1>
DocumentRoot /www/client1
ServerName www.client1.com
...
</VirtualHost>

<VirtualHost 111.222.333.1>
```

```
DocumentRoot /www/client2
ServerName www.client2.com
...
</VirtualHost>

<VirtualHost 111.222.333.2>
DocumentRoot /www/mycompany
ServerName www.mycompany.com
...
</VirtualHost>
```

Cuando se combinan servidores virtuales basados en nombres y en direcciones IP, no se olvide de utilizar la directriz NameVirtualHost ante aquellos servidores basados en nombre que tengan la misma dirección IP. Si desea que sus archivos de configuración sean legibles, introduzca una directriz NameVirtualHost por cada IP y numere todos los servidores virtuales indicando su IP, tal y como se ha hecho en el ejemplo anterior.

Uso de _default_

Este ejemplo le muestra cómo hacerse con todas las peticiones de cualquier dirección IP sin especificar, es decir, de una combinación dirección-puerto que no se utiliza con ningún servidor virtual.

El archivo de configuración httpd.conf simplificado tendría este aspecto:

```
...
<VirtualHost _default_:*>
DocumentRoot /www/default
...
</VirtualHost>
```

Al utilizar este servidor virtual con un comodín (*) en lugar del puerto, se evita que ninguna petición se escape al servidor principal. Un servidor virtual nunca atiende una petición enviada a una dirección/puerto que utilice un servidor virtual basado en un nombre. Si la petición contiene una cabecera Host: (desconocida o no), siempre lo atenderá el servidor principal de los basados en nombres (apareciendo en primer lugar la dirección/puerto en el archivo de configuración).

De todas formas, si quiere utilizar otro valor predeterminado para el puerto, también puede hacerlo. Observe el ejemplo siguiente:

El archivo de configuración httpd.conf simplificado tendría este aspecto:

```
...
<VirtualHost _default_:80>
DocumentRoot /www/default80
```

```
...
</VirtualHost>

<VirtualHost _default_:*>
DocumentRoot /www/default
...
</VirtualHost>
```

El puerto virtual predeterminado es 80 (tiene que aparecer antes que cualquier servidor cuyo valor del puerto sea un comodín) y se hace con todas las peticiones que se envían sin especificar una dirección IP. El servidor principal nunca se utiliza para atender una petición. Si desea usar un servidor virtual exclusivo para el puerto 80, no utilice el asterisco como puerto de otros servidores.

Paso de un servidor basado en un nombre a uno basado en una dirección IP

En este ejemplo vamos a suponer que al servidor virtual www.micliente1.com se le ha asignado la dirección 111.222.333.3. Para reflejar los cambios en la configuración de Apache tendrá que modificar la configuración del servidor virtual www.micliente1.com de la manera que se describe a continuación:

```
...
...
NameVirtualHost 111.222.333.1

<VirtualHost 111.222.333.1 111.222.333.3>
DocumentRoot /www/client1
ServerName www.myclient1.com
...
</VirtualHost>
```

Obsérvese que ahora la directriz VirtualHost tiene la antigua dirección IP 111.222.333.1 y la nueva 111.222.333.3. De esta forma atenderá tanto a las peticiones enviadas al servidor basado en el nombre, como a las basadas en la dirección IP. No es recomendable eliminar la antigua dirección porque es posible que muchos servidores y proxies la sigan utilizando hasta que renueven los datos DNS.

Ahora se podrá acceder al servidor virtual www.micliente.com a través de la nueva dirección (como si se tratase de un servidor basado en una dirección IP) o bien de la antigua dirección (como si fuese un servidor basado en un nombre).

Factor límite para servidores virtuales

Vamos a ver un caso que puede afectar a los servidores que alberguen una gran cantidad de host virtuales.

Los sistemas operativos UNIX limitan el número de descriptores de archivos (también conocidos como controladores de ficheros) que puede llegar a utilizar un proceso. Generalmente este valor límite es 64, aunque es cierto que se puede aumentar. Si se hace, es cierto que Apache se puede encontrar sin controladores disponibles.

Apache utiliza un descriptor de archivos para cada registro de ficheros que abre, y entre 10 y 20 descriptores adicionales para uso interno. Si Apache se aproxima al límite configurado en el sistema, se puede tratar de distribuir más controladores de ficheros.

De todas formas, la llamada puede fallar si el sistema no utiliza la directriz setrlimit( ) para las llamadas, o si la llamada no funciona con dicho sistema, o si el número de descriptores excede el máximo permitido en el sistema o bien si el sistema tiene otros límites que afecten al número de descriptores de archivos.

Si se encuentra con una de estas situaciones puede tratar de resolverlo de dos formas:

- Una de ellas consiste en reducir el número de archivos de registro que utiliza Apache. Es posible que tenga que revisar las entradas de VirtualHost y eliminar todas las propiedades de registro propias de cada servidor virtual.
- En la mayoría de los sistemas UNIX puede tratar de ajustar los límites del descriptor de archivos a mano antes de ejecutar Apache. Para ello ha de utilizar el siguiente comando:

  ```
  ulimit -S -n 100
  ```

  para ajustar el nuevo límite a 100 descriptores. Según las necesidades de cada cliente se puede utilizar un número u otro.

Hasta ahora hemos visto diferentes aspectos de la configuración de Apache, gracias a los cuales podrá ajustarlo a las necesidades propias de su sitio Web virtual.

Ahora nos vamos a centrar en otros puntos de los servidores virtuales que no están directamente relacionados con Apache, pero que son realmente importantes para los sistemas con varios servidores virtuales.

Configuración de sendmail para un sitio virtual

Tal y como he mencionado con anterioridad, los sitios Web virtuales son una de las ofertas más económicas de las que disponen los ISP. En vez de tener un servidor dedicado a Internet, una organización (o incluso un individuo) puede tener un sitio Web virtual en el ISP cuya apariencia sea exactamente la misma que uno grande. Un sitio Web virtual es el inicio de la presencia profesional de su empresa en Internet. Para parecer realmente profesionales, la mayoría de las empresas desean que aparezca el nombre de sus dominios en su dirección de correo electrónico. Por ejemplo, si tiene que averiguar la dirección de un programador de Microsoft, lo normal es que su dirección termine en @microsoft.com, ¿verdad? Pues lo mismo ocurre con el correo de cualquier empresa moderna que cuente con acceso a Internet. Así, hay una gran necesidad de soportar un dominio virtual de Internet que tenga un sitio Web virtual y un servidor de correo electrónico. Como estos requisitos son básicos para cualquier dominio virtual de Internet, nos centraremos en los cambios que hay que efectuar para crear un servidor SMTP de correo electrónico virtual.

Como sendmail es el servidor SMTP más popular, es el que utilizaré en todos los ejemplos de esta sección.

Para que lo visto aquí sea lo más sencillo posible, supondremos que queremos configurar un servidor SMTP virtual para el nombre de dominio milkyweb.com que se encuentra en nitec.com. También asumiremos que el servidor de correo de nitec.com es mail.nitec.com y será el que se utilice con milkyweb.com.

Configuración DNS para el servidor SMTP virtual

Antes de que pueda configurar el servidor de correo (sendmail) tendrá que asegurarse de que se dispone de los parámetros DNS apropiados. Los registros DNS que dictan cómo llega el correo se llaman registros MX.

En el archivo de la base de datos DNS del dominio virtual (milkyweb.com) tendrá que escribir la siguiente línea:

```
IN      MX    10 mail.nitec.com.
```

Esta línea aparece a continuación de las del registro NS. Especifica que el servidor SMTP de milkyweb.com es mail.nitec.com y que su prioridad es 10.

nitec.com tiene varios servidores de correo y todos ellos están configurados para que reciban el correo de milkyweb.com, por lo que es posible añadir varios registros MX con diferente número de prioridad y servidor de correo. Por ejemplo, si tiene otro servidor de correo llamado postoffice.nitec.com en nitec.com que esté preparado para recibir el correo de milkyweb.com, tendría que añadir:

```
IN      MX      10 mail.nitec.com.
IN      MX      20 postoffice.nitec.com.
```

Observe que los números de prioridad son 10 y 20. Se han cogido al azar pero el significado es claro: cuanto más bajo es el valor, mayor es la prioridad. Así que, en el ejemplo anterior, un servidor de mail de Internet que esté esperando enviar correo a milkyweb.com intentará contactar en primer lugar con mail.nitec.com. Si, por alguna razón, mail.nitec.com no estuviese disponible, tratará de conseguirlo a través del servidor postoffice.nitec.com.

A través de la utilidad nslookup se puede asegurar de que la configuración de los registros MX es la correcta. El comando que le muestro a continuación es el encargado de hacerlo:

```
set q=mx
milkyweb.com
```

Una vez que se ha completado la configuración DNS, el siguiente paso a dar en la configuración virtual del correo es editar el archivo /etc/sendmail.cw que se encuentra en el sistema mail.nitec.com. Como la configuración de sendmail puede variar de una versión a otra, conviene que antes de nada consulte la documentación que se encuentra en el siguiente URL antes de saltar a la siguiente sección:

```
www.sendmail.org/
```

La configuración siguiente se ha probado con sendmail 8.8.5/8.8.5.

Configurar /etc/sendmail.cw

El archivo /etc/sendmail.cw contiene los nombres de los dominios a los que atiende el demonio sendmail. En este caso, tendrá que añadir el dominio milkyweb.com:

```
nitec.com
milkyweb.com
```

A continuación creará un usuario virtual para la tabla de conversión de usuarios físicos.

Crear una base de datos para la tabla de usuarios virtuales

Una base de datos de la tabla de usuarios virtuales es la encargada de convertir las direcciones virtuales en direcciones reales. Se crea un archivo de texto donde en cada línea aparezca un par clave/valor, separado por una tabulación. Por ejemplo:

```
info@milkyweb.com              jgunchy
webmaster@milkyweb.com         kabir
@milkyweb.com                  jenny.gunchy@aol.com
```

En este ejemplo, la dirección info@milkyweb.com se convertirá a la dirección del usuario jgunchy, webmaster@milkyweb.com a la del usuario kabir y cualquier otra cosa que llegue a milkyweb.com se le enviará al usuario remoto jenny.gunchy@aol.com. Si milkyweb.com tuviese usuarios reales en otro dominio (como hotmail.com), habría que sustituir la última línea por:

```
@milkyweb.com          %1@hotmail.com
```

Con ella se convierte a todos los usuarios de milkyweb.com a los usuarios reales que se encuentran en hotmail.com. La idea es convertir a los usuarios virtuales en cuentas reales que se encuentren en un servidor remoto. Una vez que se ha creado este archivo de texto, lo podrá utilizar para crear una base de datos a través de la utilidad makemap que se distribuye con la versión estándar de sendmail.

```
makemap <database type> /etc/virtusertable <  <archivo de texto que
contiene la conversión del usuario virtual>
```

Si el archivo de texto anterior en el que se encuentra la tabla de usuarios virtuales se llamase /etc/virtusertable.txt y se estuviese trabajando con una base de datos del tipo dbm, puede utilizar el comando makemap de la siguiente manera:

```
makemap dbm /etc/virtusertable < /etc/virtusertable.txt
```

De esta forma se crean más de un archivo no de texto (generalmente /etc/virtusertable.dir y /etc/virtusertable.pag o /etc/virtusertable.db) pero en nin-

gún caso se modifica el contenido del fichero de texto /etc/virtusertable.txt. En los sistemas en los que no se pueda trabajar con el formato dbm, se podría intentar la opción hash:

```
makemap hash /etc/virtusertable < /etc/virtusertable.txt
```

El siguiente paso será configurar el archivo /etc/sendmail.cf.

Configurar /etc/sendmail.cf

Localice las siguientes líneas que se encuentran en el archivo /etc/sendmail.cf:

```
# Virtual user table (maps incoming users)
#Kvirtuser dbm /etc/virtusertable
```

Deje la segunda línea sin comentar (eliminando el signo #); sustituya dbm por **hash -o** si va a trabajar con el método hash. Para este caso el aspecto de la línea sería el siguiente:

```
Kvirtuser hash  -o /etc/virtusertable
```

y para el método dbm:

```
Kvirtuser dbm /etc/virtusertable
```

Ahora busque la regla de configuración S98 y quite el símbolo de comentario para que su aspecto sea el siguiente:

```
S98
R$* < @ $+ .REDIRECT. >              $: $1 < @ $2 . REDIRECT . >
< ${opMode} >
R$* < @ $+ .REDIRECT. > <i>          $: $1 < @ $2 . REDIRECT. >
R$* < @ $+ .REDIRECT. > < $- >       $# error $@ 5.1.1 $: "551 User has
moved; please try " <$1@$2>
```

Obsérvese que las reglas pueden variar entre las distintas versiones de sendmail, por lo que convendrá que les quite el símbolo de comentario antes de modificarlas.

Probar un servicio de correo electrónico virtual

Una vez que se han completado los pasos anteriores tendrá que reiniciar el servidor sendmail y probar los cambios.

Nota: cuando se modifica un usuario virtual (o las tablas de conversión inversa) no hay que reiniciar el servidor, únicamente cuando modifique /etc/sendmail.cf o los archivos de clase, como /etc/sendmail.cw.

Sendmail tiene un método que permite comprobar las reglas de los archivos de configuración.

Al utilizar el comando:

```
/usr/sbin/sendmail -bt
```

entrará en un modo de comprobación de reglas interactivo. En la figura 6.13 se puede ver el resultado obtenido al efectuar una de estas comprobaciones con la dirección webmaster@milkyweb.com. Obsérvese que sendmail comprueba todas las reglas de transformación hasta que finalmente decide enviar el correo a la cuenta del usuario kabir que se encuentra en el sistema local. Para comprobar una dirección todo lo que tiene que hacer es introducir el siguiente comando en modo de prueba interactiva:

```
0 usuario@host
```

```
#sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> 0 webmaster@milkyweb.com
rewrite: ruleset   0    input: webmaster @ milkyweb . com
rewrite: ruleset  98    input: webmaster @ milkyweb . com
rewrite: ruleset  96  returns: webmaster @ milkyweb . com
rewrite: ruleset  97    input: webmaster @ milkyweb . com
rewrite: ruleset   3    input: webmaster @ milkyweb . com
rewrite: ruleset  96    input: webmaster < @ milkyweb . com >
rewrite: ruleset  96  returns: webmaster < @ milkyweb . com >
rewrite: ruleset   3    input: webmaster < @ milkyweb . com >
rewrite: ruleset   0    input: webmaster < @ milkyweb . com >
rewrite: ruleset  96    input: webmaster < @ milkyweb . com >
rewrite: ruleset  98  returns: webmaster < @ milkyweb . com >
rewrite: ruleset   3    input: kabir
rewrite: ruleset  96    input: kabir
rewrite: ruleset  96  returns: kabir
rewrite: ruleset   0    input: kabir
rewrite: ruleset  98    input: kabir
rewrite: ruleset  98  returns: kabir
rewrite: ruleset   0  returns: $# local $: kabir
rewrite: ruleset  97  returns: $# local $: kabir
rewrite: ruleset   0  returns: $# local $: kabir
rewrite: ruleset  97  returns: $# local $: kabir
rewrite: ruleset   0  returns: $# local $: kabir
>
```

Figura 6.13. Comprobación de las reglas de sendmail para un dominio virtual

Si en la prueba apareciese algún problema, asegúrese de revisar todos los cambios realizados. También tendrá que verificar los archivos de registro de

sendmail que se encuentran en su servidor para ver si encuentra algún error. Si todo es correcto tendrá que intentar enviar correo desde otro servidor para ver si se recibe sin problemas.

Tenga en cuenta que si efectúa cambios DNS para configurar el servicio, es posible que el resto de los servidores con los que trabaja en Internet tarden algún tiempo en actualizar sus datos.

7 SSI para Apache

Posteriormente aprenderá que el contenido Web dinámico se puede crear a través de programas CGI. Pero hay tareas a las que no se puede en el proceso de creación de programas en CGI, pero que requieren cierta intervención dinámica. Por ejemplo, supongamos que quiere añadir un mensaje de Copyright estándar a sus páginas HTML: ¿cómo lo haría? Tiene dos soluciones: añadir el contenido del mensaje en todas y cada una de las páginas HTML de su sitio Web o escribir un programa CGI que lo haga. Ninguna de estas opciones es elegante. La primera requiere que cada vez que se cambie el contenido del mensaje haya que modificar manualmente todos los archivos. Con la segunda tendrá que ejecutar su programa CGI antes de enviar las páginas al explorador Web de los usuarios. Es decir, que todos los enlaces de sus páginas Web tendrán que llamar al programa CGI para que añada el mensaje del Copyright a la página que se va a abrir. Estas situaciones requieren una solución más sencilla: SSI (Server Side Includes), que es lo vamos a ver en este capítulo.

Qué son los SSI

Una página SSI suele ser una página HTML que incorpora una serie de comandos del servidor Web. Generalmente, el servidor Web no analiza las

páginas HTML antes de entregárselas al explorador Web (o a cualquier cliente Web). Sin embargo, una página HTML SSI sí que es analizada y si se encuentra algún comando SSI en la página, lo ejecutará. La figura 7.1 muestra el ciclo de distribución de una página HTML y de otra SSI que parten del mismo servidor Web.

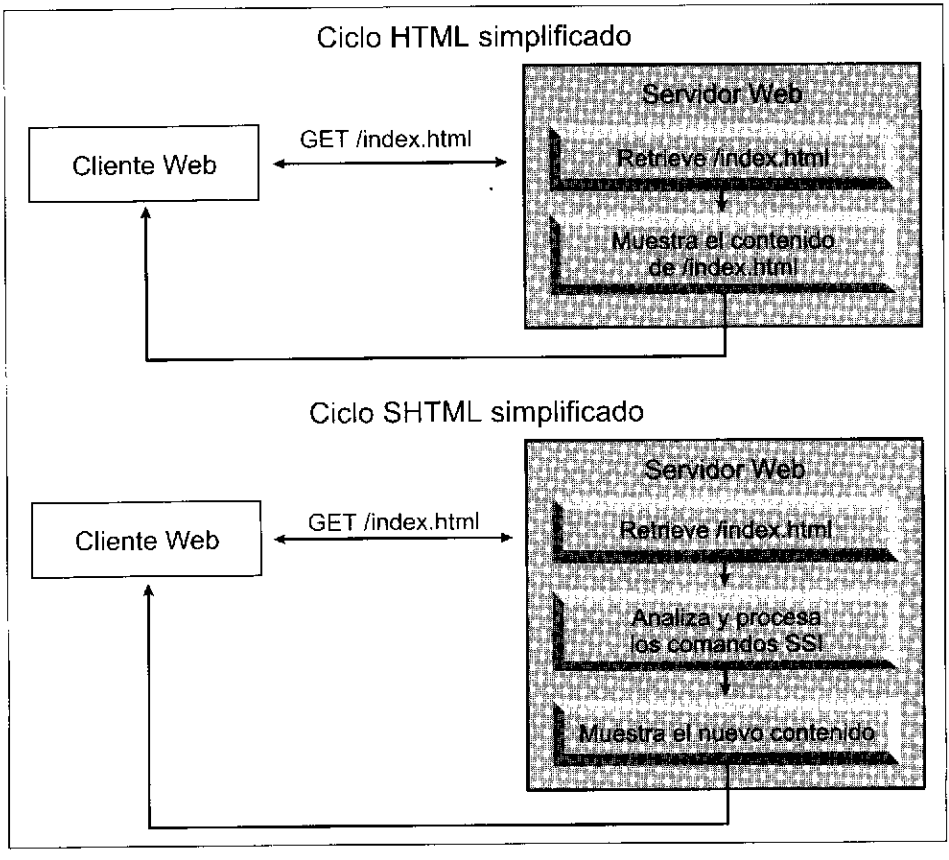


Figura 7.1. Diagrama simplificado de la distribución de una página HTML y de otra SSI que parten del mismo servidor Web

Como se puede ver, en la versión SSI de la página HTML lo primero que se hace es analizar los comandos SSI. Y antes de entregar la página al servidor Web, se ejecutan.

Apache lleva mucho tiempo trabajando con SSI. El módulo responsable de SSI es mod_include, que se incluye en la distribución estándar de Apache. Durante algún tiempo existió un módulo adicional llamado XSSI (eXtended Server Side Includes), que incluía una serie de características mejoradas. El

paquete XSSI, fue escrito por Howard Fear y se incluye dentro del módulo mod_include de la distribución estándar de Apache, por lo que ahora es capaz de hacerse cargo de los comandos SSI y XSSI.

Truco: si aún trabaja con una versión antigua de Apache (es decir, anterior a 1.2.x), tendrá que instalar el módulo XSSI, que se encuentra en www.pageplus.com/~hsf/xssi/.

Gracias al código de Fear se dispone de una serie de funcionalidades propias de los lenguajes de programación, como variables o control de la configuración de los comandos SSI del servidor Apache. De todas formas, antes de poder utilizar los comandos SSI en sus páginas HTML tendrá que configurar el servidor Apache.

Configurar Apache para SSI

Antes de configurar Apache para SSI tendrá que verificar el ejecutable de Apache (httpd) para asegurarse de que se incluye el módulo mod_include. Para ello tendrá que ejecutar el comando:

```
httpd -l
```

De esta forma accederá a una lista en la que se muestran todos los módulos que forman parte del ejecutable de Apache. Por defecto, tendría que tener compilado este módulo. Si no es así, habrá de añadirlo al archivo de configuración utilizando la siguiente línea:

```
AddModule modules/estándar/mod_include.o
```

A continuación ejecutará las utilidades Configure y make, respectivamente.

Aunque el mod_include se compila por defecto en la distribución estándar de Apache, el análisis de las páginas HTML no se hace por defecto. Para que Apache pueda trabajar con SSI (versión 1.2 o posteriores) tendrá que dar los siguientes pasos:

1. Agregar un nuevo controlador para las páginas HTML SSI/IXSSI.
2. Agregar una nueva extensión de archivo para las páginas HTML SSI/IXSSI.
3. Permitir el análisis SSI para un directorio.

Cuando se completen dichos pasos, con un directorio que se llame "chapter 07" que se encuentre en el servidor virtual apache.nitec.com, nos encontraremos con la siguiente configuración:

```
<VirtualHost 206.171.50.50>

ServerName apache.nitec.com
DocumentRoot /data/web/apache/public/htdocs
ScriptAlias /data/web/apache/public/cgi-bin

<Directory /data/web/apache/public/htdocs/chapter_07>
AddHandler server-parsed .shtml
AddType text/html .shtml
Options +Includes
</Directory>

</VirtualHost>
```

Advertencia: el uso de la directriz Options activa todos los comandos SSI. Si tiene previsto desactivar la ejecución de programas externos a través de los comandos SSI, tendrá que utilizar IncludesNOEXEC. Pero tenga en cuenta que con ella se desactiva la carga de programas externos a través de comandos SSI.

Vamos a ver más detenidamente cada uno de los pasos que hay que dar para activar los comandos SSI.

Agregar un nuevo controlador para las páginas HTML SSI

Por defecto, Apache no analiza los archivos HTML que se le solicitan. En caso contrario analizaría todas las páginas HTML, incluso aquellas en las que no hubiese ningún comando SSI, con lo que reduciría la velocidad de distribución de páginas HTML. La solución a este problema sería identificar las páginas que tienen comandos SSI, para lo cual se utilizaría una extensión diferente al resto de páginas HTML (.html o .htm). La extensión acordada para determinar a las páginas HTML con comandos SSI es .shtml. Hay que indicarle a Apache que tiene que considerar a todos los archivos con la extensión .shtml como páginas HTML que pueden tener comandos SSI. Para ello se utiliza AddHandler:

```
AddHandler server-parsed .shtml
```

Esta directriz le dice a Apache que los archivos .shtml se deben tratar con un controlador que se encuentra en el módulo mod_include.

Si, por alguna razón, tuviese que utilizar la extensión .htm o .html como SSI, no use:

```
AddHandler server-parsed .html
AddType text/html .html

AddHandler server-parsed .htm
AddType text/html .htm
```

De esta forma se degradaría el rendimiento del servidor. Apache procesaría todos los archivos .html y .htm, es decir, que también se analizarían los archivos que no tuviesen comandos SSI, con lo que se retrasa el tiempo de entrega del archivo.

De todas formas, si tiene que utilizarlos, le recomiendo que use la directriz XbitHack que se encuentra en el módulo mod_include.

La directriz XbitHack

La directriz XbitHack controla el análisis que efectúa el servidor a los archivos asociados con el tipo MIME text/html.

```
Sintaxis: XBitHack on | off | full
Predeterminado: XBitHack off
Contexto: configuración del servidor, servidor virtual, directorio,
archivo de control de acceso por directorio (.htaccess)
Override: Opciones
Módulo: mod_include
```

Generalmente, los únicos archivos asociados con text/html son .htm y .html. El valor predeterminado "off" le indica al servidor que no los analice. Cuando se utiliza "on", cualquier archivo HTML que tenga permiso de ejecución de su propietario se considerará SSI y se analizará. Cuando la directriz tiene el valor "full", el servidor comprueba quién es el propietario y el grupo de bits ejecutables de los parámetros de permiso. Si se ha configurado el grupo de bits ejecutables, Apache toma la fecha de modificación del último archivo devuelto. En caso contrario no se enviará la fecha de la última modificación. Gracias a este conjunto de bits los clientes y proxies pueden almacenar en memoria caché los resultados de sus peticiones. De todas formas no se recomienda el uso de full con las páginas SSI ya que produce una salida diferente a la estándar cuando se analiza y procesa.

Nota: no olvide que al utilizar esta directriz aún tendrá que dar el tercer paso de los que hemos hablado con anterioridad.

Agregar una nueva extensión para las páginas HTML SSI

Aunque Apache ahora ya sabe cómo controlar el archivo .shtml, necesitará que se le indique qué le ha de decir al explorador Web sobre estos ficheros. Los servidores Web envían cabeceras de información con cada petición a través de las cuales se le indica a los exploradores Web el tipo del contenido que se les envía en la respuesta a su petición. Por lo tanto, tendrá que instruir a Apache para que cuando responda una petición de un archivo .shtml le diga al cliente servidor que la información enviada sigue siendo un documento HTML. Así, el explorador mostrará el contenido del documento en la pantalla igual que ha hecho siempre. El tipo MIME para un contenido HTML es text/html. La siguiente línea le muestra cómo se le puede decir a Apache que genere una cabecera para el tipo de contenido text/html al transmitir una página .shtml:

```
AddType text/html       .shtml
```

Nota: para conservar una compatibilidad con las versiones anteriores, los documentos del tipo MIME text/x-server-parsed-html o bien text/x-server-parsed-html3 también se analizarán (y el resultado se mostrará utilizando el tipo text/html).

Activar el análisis SSI para un directorio

Tanto Apache como los exploradores Web saben cómo tratar los archivos .shtml. De todas formas, Apache aún no está listo para analizar las páginas .shtml.

Por medio de la directriz Options le indicará a Apache que quiere que soporte Includes. Antes de nada, habrá que determinar dónde se colocará dicha directriz.

Si quiere que activar el soporte SSI en todo el sitio Web (principal), añada la siguiente directriz en uno de los archivos de configuración globales (por ejemplo, en access.conf):

```
Options +Includes
```

Si quiere activar el soporte SSI para un sitio Web virtual, tendrá que colocar la directriz anterior dentro de un contenedor <VirtualHost ...>. O, si desea controlar esta opción de directorio a directorio, tendrá que colocarla dentro del contenedor <Directory ...> que se encuentra en el archivo de control de acceso por directorios (.htaccess).

Si utiliza este archivo para activar el soporte SSI, asegúrese de que utiliza la directriz AllowOverride con el sitio al que pertenece el directorio. La directriz AllowOverride ha de permitir que se sobreescriba la opción Includes. Por ejemplo, si para este sitio el valor de AllowOverride es None, no se efectuará ningún análisis SSI.

Nota: si no utiliza el signo + en la línea Options del ejemplo anterior, se desactivarán todas las opciones a excepción de Includes.

Ahora que sabe cómo activar el soporte SSI en Apache, pasaremos a ver los comandos SSI con mayor detenimiento.

Los comandos SSI

Los comandos SSI se incluyen dentro de las páginas HTML en forma de comentarios. La estructura base de los comandos sería la siguiente:

```
<!-#comando argumento1=valor argumento2=valor argumento3 valor ...>
```

A menudo se encierra el valor entre comillas. Muchos comandos sólo pueden trabajar con un único par atributo-valor. Obsérvese que el terminador (→) ha de ir precedido de un espacio en blanco para asegurarse de que no se considera parte del comando SSI.

Ahora vamos a ver los comandos SSI con los que se puede trabajar.

config

El comando config le permite configurar los mensajes de error de análisis y el formato que se utilizará para mostrar la información sobre la fecha y tamaño del archivo. Para ello se utilizan las siguientes líneas de código:

```
config errmsg="error message"
config sizefmt=["bytes"   "abbrev"]
config timefmt=format string
```

La primera sintaxis muestra cómo se crea un mensaje de error:

```
config errmsg="error message"
```

Este mensaje de error aparecerá siempre que aparezca un problema de análisis. Por ejemplo, el listado 7.1 le muestra el contenido de un archivo llamado config_errmsg.shtml.

```
<HTML>
<HEAD><TITLE> Biblia del Servidor Apache </TITLE></HEAD>
<BODY BGCOLOR="white">

<FONT SIZE=+1 FACE="Arial"> Simple SSI Example #1</FONT>
<HR SIZE=1>

<P> Example of the SSI <STRONG>config errmsg</STRONG> command: </P>
<P> Embedded commands: <BR><BR>

<CODE>
       &lt;!-#config errmsg="Mensaje de error SSI personalizado."
 &gt; <BR>
       &lt;! #config errmsg_type  "Mensaje de error SSI
personalizado."  &gt;
</CODE>

</P>

<P> Result: <BR>

<!-#config errmsg="Mensaje de error SSI personalizado." -->
<BR>
<! #config errmsg_type="Mensaje de error SSI personalizado." -->

</P>

</BODY>
</HTML>
```

Listado 7.1. config errmsg.shtml

En este archivo ejemplo tenemos dos comandos SSI:

```
<!-#config errmsg="Mensaje de error SSI personalizado."    >
```

y

```
<!-#config errmsg type "Mensaje de error SSI personalizado." -->
```

El primero es la configuración del comando errmsg que especifica que el mensaje de error será "Mensaje de error SSI personalizado". El segundo es un

comando SSI no válido que he escrito intencionadamente para que veamos qué ocurre.

Este comando causa un error de análisis, por lo que aparecerá el mensaje que se acaba de configurar. Se pueden introducir etiquetas HTML o insertar scripts en la cadena de mensajes de error. Por ejemplo, la línea que aparece a continuación muestra una ventana de aviso en JavaScript con un mensaje de error:

```
<!  config errmsg="<SCRIPT LANGUAGE=JavaScript> alert("Ha ocurrido un
error. \n Por favor, informe a webmaster@nitec.com");</SCRIPT>"    >
```

La segunda sintaxis le permite escoger el formato de salida para el tamaño del archivo:

```
config sizefmt=["bytes" | "abbrev"]
```

Los valores aceptables para el formato son "bytes" o bien "abbrev". Por ejemplo:

```
<!- config sizefmt="bytes"    >
```

Este comando nos muestra el tamaño de los archivos en bytes. Para mostrar los ficheros en kilobytes o megabytes, utilice:

```
<!- config sizefmt="abbrev" -->
```

La sintaxis final le permitirá elegir el formato para la fecha:

```
config timefmt=format string
```

Los valores más utilizados para la cadena de formato (format string) son los siguientes:

%a	La abreviatura del día de la semana en el idioma local.
%A	El nombre completo del día de la semana en el idioma local.
%b	La abreviatura del mes en el idioma local.
%B	El nombre completo del mes en el idioma local.
%c	Representación para fecha y hora de acuerdo con la norma local.
%d	Día del mes en formato decimal (del 01 al 31)
%H	Hora en formato decimal usando el reloj de 24 horas (del 00 al 23)

%I La hora como número decimal utilizando el reloj de 12 horas (del 01 al 12)

%j Día del año en formato decimal (del 001 al 366)

%m Mes en formato decimal (del 01 al 12)

%M Minuto en el formato decimal (del 01 al 24)

%p Formato a.m. o p.m., de acuerdo con el valor local.

%S Segundo en formato decimal.

%w Día de la semana en formato decimal, empezando por el Domingo, al que le corresponde el 0.

%x Representación de la fecha de acuerdo con la norma local, pero sin mostrar la hora.

%X Representación de la hora de acuerdo con la norma local, pero sin mostrar la fecha.

%y Año en formato decimal sin el siglo (del 00 al 99)

%Y Año en formato decimal con el siglo.

%Z Nombre o abreviatura de la hora local.

%% Un carácter %.

Considere este ejemplo:

```
<!-#config timefmt="%c" -->
```

nos muestra la fecha como Mar May 20 00:54:07 1998.

Y el ejemplo siguiente:

```
<!-#config timefmt-"%m %d %Y" -->
```

nos mostrará la fecha como 05/20/1998.

echo

El comando echo imprime una de las variables Include (que se definirá más tarde) o cualquiera de las variables de entorno CGI.

La sintaxis es:

```
echo var="nombre variable"
```

Si no se puede acceder al valor de la variable, se imprime "(none)". Cualquier fecha impresa estará sujeta a la configuración timefmt. Por ejemplo:

```
<!-#config timefmt="%m/%d/%Y" -->
<!-#echo var-"DATE_LOCAL"    >
```

Con estas dos líneas se imprimirá la fecha como 05/20/1998 debido al formato definido en timefmt.

exec

El comando exec le permite ejecutar programas externos. Este puede ser un programa en CGI o cualquier otro ejecutable, ya sea en formato binario o en script. La sintaxis para los programas CGI es:

```
exec cgi="path/to/cgi/program"
```

La sintaxis para cualquier otro programa es:

```
exec cmd="path/to/other/programs"icon-note
```

Nota: si se utiliza el valor IncludesNOEXEC en la directiz Options, se desactivará este comando.

Vamos a ver sus distintas opciones.

cgi

El valor cgi especifica un URL (codificado-%) relativo al path del script CGI. Si el path no comienza con una barra inclinada (/), entonces se referenciará respecto al documento actual. Un script será quien invoque al documento al que se hace referencia en este path, aun en caso de que el servidor no lo reconozca como tal. De todas formas, el directorio que contiene el script ha de estar disponible para todos los script CGI (por medio de ScriptAlias o ExecCGI Option).

El script CGI se obtiene de PATH_INFO y de la cadena solicitada (QUERY_STRING) por el cliente. No se puede especificar a través del URL. Las variables Include, así como las CGI estándar, estarán a disposición del script.

El script devuelve una cabecera Location: como salida, que se traduce a una etiqueta HTML. Por ejemplo, el siguiente código muestra un script CGI en Perl que imprime una cabecera Location:.

```
#!/usr/local/bin/perl
print "Location: http://apache.nitec.com/\\n\\n\";
exit 0;
```

Cuando un explorador Web solicita el archivo exec_cgi.shtml, que es el que aparece en el listado 7.2, el servidor devuelve la cabecera Location: dentro de una etiqueta HTML, en vez enviar al explorador a la dirección http://apache.nitec.com.

```
<HTML>
<HEAD> <TITLE> Biblia del Servidor Apache </TITLE></HEAD>

<BODY BGCOLOR="white">
<FONT SIZE=+1 FACE="Arial"> Simple SSI Example #2</FONT>
<HR SIZE=1>

<P> Example of the SSI <STRONG>exec cgi</STRONG> command: </P>
<P> Embedded commands: <BR><BR>

<CODE> &lt;!-#exec cgi="/cgi-bin/loc.cgi" -&gt; <BR> </CODE>
</P>
<P> Result: <BR>  <!-#exec cgi="/cgi-bin/loc.cgi" --> </P>

</BODY>
</HTML>
```

Listado 7.2. exec_cgi.shtml

Como se puede observar en el listado, la única llamada SSI del archivo es:

```
<!-#exec cgi="/cgi-bin/loc.cgi" -->
```

La salida de este archivo será una etiqueta HTML.

cmd

Cuando se invoca a un programa que no sea CGI, tendrá que utilizar la versión cmd de exec. En la mayoría de los sistemas UNIX el servidor ejecuta la cadena utilizando el shell sh (/bin/sh). Este comando puede disponer de las variables Include. Por ejemplo, el listado 7.3 nos muestra un archivo llamado exec_cmd.shtml.

```
<HTML>
<HEAD> <TITLE> Biblia del Servidor Apache </TITLE></HEAD>
<BODY BGCOLOR="white">

<FONT SIZE +1 FACE="Arial"> Simple SSI Example #3</FONT>
<HR SIZE=1>
```

```
<P> Example of the SSI <STRONG>exec cmd</STRONG> command: </P>
<P> Embedded commands: <BR><BR>

<CODE>
&lt;!-#exec cmd="/bin/date +%m/%d/%y" -&gt; <BR>
&lt;!-#exec cmd="/bin/ls -l ./" -&gt; <BR>
</CODE>

</P>
<P> Result: <BR>

<!-#exec cmd="/bin/date +%m/%d/%y" --> <BR>

<PRE>

<!-#exec cmd="/bin/ls -l ./*.html"   > <BR>

</PRE>

</P>
</BODY>
</HTML>
```

Listado 7.3. exec_cmd.shtml

Este archivo tiene dos llamadas cmd:

```
<!-#exec cmd="/bin/date +%m/%d/%y" -->
<!-#exec cmd="/bin/ls -l ./*.html" -->
```

La primera invoca a la utilidad de UNIX /bin/date con el argumento +%m/%d/%y, mientras que la segunda llama a la utilidad ls de UNIX, en donde ./*.html es el argumento.

El formato de la salida de ls se especifica con el par de etiquetas <PRE> y </PRE>. Si desea mostrar algo que utilice nuevas líneas, habrá de usar las etiquetas <PRE> para que dicha salida sea legible.

fsize

Este comando imprime el tamaño de los archivos especificados. La sintaxis utilizada dependerá de si el path del directorio es relativo o virtual:

- `fsize file="path"`
- `fsize virutal="URL"`

Cuando se utiliza la primera sintaxis, se supone que el path es relativo al directorio que contiene el documento SSI que se está analizando. No utilice

../ en el path, ya que en ese caso trabajará con el path absoluto. Aunque con este sistema no se puede acceder a un script CGI, sí se puede alcanzar otro documento analizado. Por ejemplo:

```
<!-#fsize file="download.zip">
```

Si se utiliza la segunda sintaxis, se supone que se trabaja con un URL de un path virtual (codificado %). Si no comienza con una barra inclinada (/), se supone que el path está referido al documento en cuestión. Con este método se puede acceder a los archivos normales, pero no a los script CGI. De todas formas, puede acceder a otros documentos analizados. Por ejemplo:

```
<!-#fsize virtual="/download/free_software.zip">
```

Nota: el formato de salida está sujeto a lo especificado en sizefmt.

flastmod

El comando flastmod imprime la última fecha de modificación del archivo especificado. De nuevo nos encontramos con dos posibles sintaxis, dependiendo del path del directorio:

- `flastmod file="path"`
- `flastmod virtual="URL"`

El formato de la salida depende de lo especificado en timefmt. Por ejemplo:

```
<!-#flastmod file="free_software.zip">
<!-#flastmod virtual="/download/free_software.zip">
```

Nota: si aún no tiene muy clara la diferencia que hay entre las sintaxis, consulte el comando fsize como ejemplo. Para controlar cómo se imprime la fecha de la última modificación, consulte el comando config.

include

La directriz include inserta el texto de un fichero dentro de un documento SSI en el momento en que se procesa.

La sintaxis depende del path del directorio:

- `include file="path"`
- `include virtual="URL"`

Nota: si desea más información sobre las diferencias entre los modos file y virtual consulte el comando fsize.

Cualquiera de los archivos que se incluya estará sujeto al control de acceso usual. Si el directorio que contiene el archivo analizado tiene el valor de la directriz Option configurada a IncludesNOEXEC y la inclusión el archivo puede hacer que se ejecute el programa, entonces no se añadirá. De esta forma se evita la ejecución de script CGI. Para llamar a los script CGI se utiliza el sistema tradicional, es decir el URL completo a través de un comando dado, incluyendo la cadena de la petición.

Por ejemplo:

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

Con esta orden si se incluye el archivo copyrights.html en el documento actual. Este comando es especialmente útil para los casos en los que se añade código repetitivo HTML en los archivos. Muchos sitios Web usan una barra de menús estándar en cada página. Si dicha barra se encuentra en el archivo HTML menu.html, se podrá invocar desde las páginas SSI utilizando una llamada similar a la que hemos visto en el ejemplo anterior. En el futuro, cuando haya que efectuar algún cambio al menú, el administrador del sistema tan sólo tendrá que modificar la página menu.html. De esta forma se ahorra una gran cantidad de trabajo.

Cuando se detectan inclusiones recursivas, los mensajes de error se generan según se completa el primer pase. Por ejemplo, si el archivo a.shtml tiene una llamada SSI del tipo:

```
<!-#include file="b.shtml" -->
```

y b.shtml tiene otra:

```
<!-#include file="a.shtml" -->
```

entonces Apache procede con los registros y después nos muestra un mensaje de error.

printenv

El comando printenv imprime una lista de todas las variables existentes y de sus valores. Su sintaxis es la siguiente:

```
printenv
```

Por ejemplo:

```
<!-#printenv  ->
```

imprimirá todas las variables de entorno CGI y las Include. Para que la salida generada sea más legible, le aconsejo que utilice el par de etiquetas <PRE>.

set

El comando set ajusta el valor de una variable definida por el usuario. La sintaxis es:

```
set var="nombre variable" value="valor de la variable"
```

Por ejemplo:

```
<! #set var="home" value="index.shtml" -->
```

Variables SSI

El módulo SSI tiene un conjunto de variables. Además de las variables de entorno CGI (que veremos en capítulos posteriores), dispone de todos los archivos SSI. A estas variables se las conoce como "variables include". Se pueden utilizar con comandos SSI (echo, if, elif, etc.) y con cualquier programa que invoque un comando SSI. Son las siguientes:

DATE_GMT	Fecha actual según el meridiano de Greenwich.
DATE_LOCAL	Fecha actual según la zona horaria.
DOCUMENT_NAME	Nombre del archivo SSI.
DOCUMENT_URI	El URL (codificado-%) del path del documento.
LAST_MODIFIED	La fecha de la última modificación del archivo. Está sujeta al formato configurado con timefmt.

Las variables include y las CGI se pueden utilizar en cualquier momento. Todas ellas se pueden usar como argumentos de otros comandos. La sintaxis para utilizar una variable ya definida es:

```
<!-#command argumento1="$variable1" argumento2="$variable2" ...-->
```

Como se puede apreciar, al nombre de la variable le precede el signo $. Aquí tiene otro ejemplo:

```
<!-#config errmsg="Hubo un error en $DOCUMENT_NAME." -->
```

Nota: al usar variables en var="variable", el signo $ no seguirá siendo necesario. Por ejemplo:

```
<!-#echo var="DOCUMENT_NAME" -->
```

Si tiene que insertar un signo $ dentro del valor de una variable, tendrá que hacerlo anteponiendo una barra invertida. Por ejemplo:

```
<!-#set var="password" value="\$cheese" -->
<!-#echo var="password"   >
```

De esta forma se imprimirá $cheese como valor de "password".

Si tiene que hacer referencia a una variable en mitad de una secuencia de caracteres, coloque el nombre de la misma entre paréntesis. Por ejemplo:

```
<!-#set var="uniqueid" value="${DATE_LOCAL}_${REMOTE_HOST}" -->
```

Comandos de control de flujo

Al igual que los lenguajes de programación, el módulo SSI también puede controlar el flujo. Gracias a sus comandos, puede crear diferentes tipos de salidas. Lo más sencillo es:

```
<!-#if expr "test_expression" -->
<!-#endif -->
```

Aquí se evalúa test_expression. Si el resultado es "true" (verdadero), entonces se incluye el comando endif en la salida. Test_expresion puede ser:

```
string
```

que será verdadero siempre que la cadena no esté vacía, o

```
string1 comparison_operator string2
```

Aquí, comparsion_operator puede ser =, !=, <, >, <= o >=.

Nota: si string2 tiene la forma /cadena/, a la hora de efectuar la comparación se tomará como una expresión regular.

Vamos a ver un ejemplo con string:

```
<!-#if expr="foobar" -->
Esta prueba tiene éxito.
<!-#endif -->
```

Esta sintaxis siempre imprimirá "Esta prueba tiene éxito." porque la expresión es verdadera cuando test_expression no es una cadena vacía. Pero si expr="foobar" cambia por expr="" o expr="", entonces el texto que se encuentra dentro del bloque if-endif nunca formará parte de la salida impresa.

Vamos a ver un ejemplo de una cadena de una prueba de igualdad:

```
<!-#set var="quicksearch" value="yes" -->

<!-#if expr="$quicksearch = yes" -->
Solicitada búsqueda rápida.
<!-#endif -->
```

Aquí, a la variable llamada quicksearch se le asigna "yes" para más tarde compararla con ese mismo valor. Como el modelo con el que se compara y el valor de la variable son iguales, aparecerá en la salida que se genere la cadena de texto "Solicitada búsqueda rápida.".

Al usar operadores lógicos como !, && y || se pueden crear test_expressions más complejas. Por ejemplo:

```
<!-#if expr="${REMOTE_HOST} - /206\.171\.50/ &&
${DOCUMENT_NAME} = /${DATE_LOCAL}/" -->
<! #exec cmd "/usr/local/build/timesheet/timecalc.pl">
<!-#endif -->
```

En este caso, test_expression se compone de dos expresiones. Se valora la primera subexpresión, ${REMOTE_HOST} = /206\.171\.50/, para ver si la variable definida por el servidor REMOTE_HOST coincide con la dirección de red 206.171.50. Obsérvese que dicha expresión se ha escrito utilizando un

formato regular /206\.171\.50, donde cada punto va precedido de una barra invertida. Dicha barra es necesaria ya que gracias a ella se anula el significado especial del punto.

La segunda subexpresión, ${DOCUMENT_NAME} = /${DATE_LOCAL}, se evalúa para ver si el archivo SSI que se está procesando tiene un nombre que coincida con la fecha actual (obviamente depende de la configuración que se establezca con timefmt). Y, por último, el operador && (equivale a AND) requiere que ambas expresiones sean verdaderas para que toda la expresión también lo sea. Si la expresión final es verdadera, entonces se ejecutará el script timecalc.pl por medio del comando exec cmd.

Otras operaciones lógicas que se pueden hacer con test_expresion son:

```
<!-#if expr="! test_expression"   >
Esta frase se imprimirá únicamente cuando test_expression sea falsa.
<! #endif  ->
```

y

```
<!-#if expr="test_expression1  | test_expression2" -->
Esta frase se imprimirá cuando por lo menos una de las
test_expressions sea verdadera.
<!-#endif  ->
```

Los signos = (igual) y != (no igual) tienen mayor preferencia que los operadores && (y) y || (o). La prioridad del operador ! (no) es la más alta. Si desea aumentar la prioridad, puede utilizar un par de paréntesis.

```
<!-#if expr="($win = yes && $loss   false) != ($profit = yes)" -->
```

Aquí, ($win = yes && $loss = false) se evalúa antes que el operador !=.

Nota: cualquier cosa que no se reconozca como un operador o una variable se considerará una cadena. Estas se pueden colocar entre comillas: 'cadena'. Las cadenas que no vayan entre paréntesis no pueden tener espacios en blanco (ni tabuladores) porque se utilizan para separar elementos como variables. Si en una misma fila aparecen varias cadenas, se podrán concatenar utilizando espacios en blanco.

Si necesita construcciones más complejas puede utilizar lo siguiente:

```
<!-#if expr="test_condition1" -->

<!-#elif expr="test_condition2" -->
```

```
<!-#else -->

<!-#endif -->
```

Elif le permite crear condiciones else-if. Por ejemplo:

```
<!-#if expr="${HTTP_USER_AGENT} = /MSIE/" -->

      <!-#set var="browser" value="IE" -->
      <!-#include file="vbscript.html" -->

<!-#else -->

      <!-#set var="browser" value="Navigator" -->
      <!-#include file="javascript.html" -->

<!-#endif -->
```

Aquí, se comprueba la variable HTTP_USER_AGENT para ver si contiene la cadena MSIE (cadena que se utiliza con el explorador Microsoft Internet Explorer). Si es así, la variable browser tomará el valor "IE" y el archivo vbscript.html se insertará en el documento actual. Por otro lado, si la variable HTTP_USER_AGENT no contiene la cadena MSIE se supone que se utiliza otro navegador (Netscape Navigator) y el archivo que se insertará en el documento será javascript.html. Gracias a la construcción if-then-else, este ejemplo define distintos valores para la misma variable y carga distintos archivos.

8 Configuración CGI

El principal trabajo de un servidor Web consiste en distribuir el contenido de las peticiones que recibe a los clientes, como por ejemplo a un explorador Web.

En la mayoría de los casos, el contenido de la petición se guarda en algún tipo de archivo al que pueda acceder y leer el servidor. Este recupera el archivo y transmite su contenido al sistema del cliente. Pero, ¿qué ocurre cuando no se puede leer el contenido de dicho archivo y se tiene que generar desde una aplicación? El servidor Web tendrá que ejecutar una aplicación y enviar la salida que genere. Aquí es donde entra la interfaz Common Gateway Interface (CGI).

La especificación CGI le indica al servidor Web cómo tiene que interactuar con una aplicación externa. Un servidor Web que ejecute aplicaciones CGI permite que prácticamente cualquiera pueda ejecutar una que se encuentre en un sistema remoto.

Este es uno de los aspectos que más asusta a la gente que no está bien informada sobre CGI.

En el presente capítulo vamos a poder ver las bases de CGI para que consiga tener una visión lo más clara posible y también veremos algunos de los detalles de la configuración de Apache para que pueda trabajar con ejecutables CGI.

¿Qué es CGI?

La verdad es que es muy probable que ya haya utilizado más de una aplicación CGI en sus paseos por la Red. Son muchos los sitios Web que utilizan aplicaciones CGI para dotar de un mayor dinamismo a la Red. Obviamente, cuantas más tecnologías Web surgen, tantas más formas de distribuir contenidos dinámicos por la Red. La mayoría de estas soluciones dependen de un sistema operativo, especificaciones de un lenguaje o software comercial. En la otra cara de la moneda tenemos CGI, una especificación independiente del lenguaje que se puede implementar utilizando cualquiera de las aplicaciones de desarrollo de lenguajes de programación, como C, C++, Perl, lenguajes de shell script o incluso Java.

Vamos a ver cómo trabaja un programa CGI (figura 8.1). La idea básica es que el servidor Web obtiene cierto URL que, mágicamente (de momento), le dice al servidor que tiene que ejecutar una aplicación externa llamada *helloworld.cgi*. El servidor Web abre dicha aplicación, espera a que se complete su ejecución y muestra la salida que ha generado. A continuación transmite la salida al cliente Web que se encuentra en otro sistema.

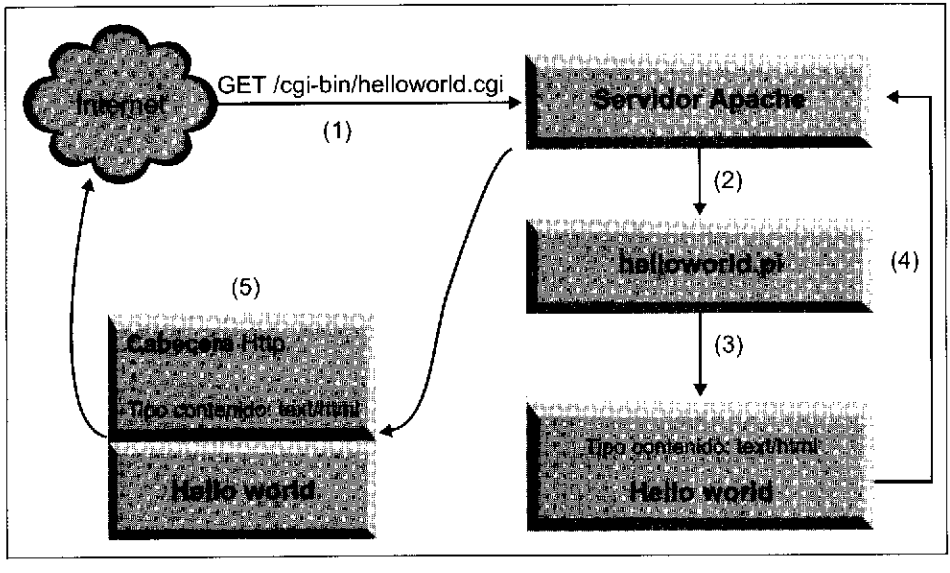


Figura 8.1. Funcionamiento de un programa CGI

¿Qué ocurre cuando desea que el cliente pueda interactuar con la aplicación? Bueno, la entrada de datos que efectúa el cliente tendrá que ser a través

de aplicaciones. De igual forma, cuando una aplicación genera una salida, ¿cómo sabrá el servidor o el cliente de qué tipo es dicha salida? Un programa puede generar un mensaje de texto, un formulario HTML para que se introduzcan más datos, una imagen, etc. Como se puede apreciar, la salida variará bastante de una aplicación a otra. Por lo tanto, tiene que haber una forma de ponerse en contacto con el servidor Web y con el cliente para comentarles el tipo de salida que han generado.

CGI define un grupo de estándares para que el servidor le entregue las entradas del cliente a las aplicaciones externas. También define cómo enviará la aplicación externa la salida que genere. Cualquier aplicación que utilice estos estándares se podrá definir como script/aplicación/programa CGI. Para simplificar usaremos el término *programa CGI* para referirnos a aquello (script en Perl o programa en C) que cumpla la especificación CGI. En la siguiente sección veremos el funcionamiento del proceso de entrada/salida.

Entrada y salida CGI

Son muchas las formas en las que un servidor Web puede recibir información procedente de un cliente (por ejemplo, a través de un explorador Web). El protocolo HTTP define un sistema de intercambio de información entre el servidor Web y el cliente. El método más común para solicitar un intercambio de información es a través de GET y POST.

Petición GET

La petición GET es el método más sencillo. Siempre que se introduzca la dirección de un sitio Web en su explorador Web, se genera una petición GET y se le envía al servidor Web correspondiente. Si escribe:

```
www.idgbooks.com
```

su explorador Web envía:

```
GET /
```

al servidor Web www.idgbooks.com. GET le pide al servidor Web de IDG que le envíe el documento que se encuentra en el nivel superior del árbol de directorios. A este documento se le suele llamar página principal y su contenido es un índice de todo el árbol de directorios. Además, el protocolo HTTP le permite codificar información adicional en una petición GET. Por ejemplo:

www.miempresa.com/cgi-bin/search.cgi?books=cgi?author=kabir

Aquí, la petición GET será así:

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

Se le está diciendo al servidor que ejecute el programa CGI /cgi-bin/search.cgi y que entregue como datos de entrada books=cgi y author=kabir.

Cuando un servidor Web compilador de CGI, como Apache, recibe una petición de este tipo, sigue las especificaciones CGI y le entrega los datos a la aplicación (en este caso, search.cgi, que se encuentra en el directorio cgi-bin). Cuando se solicita un recurso CGI a través del método HTTP GET, Apache hace lo siguiente.

1. Prepara las variables de entorno para el programa CGI. Se guarda el nombre del método HTTP solicitado en la variable REQUEST_METHOD y los datos recibidos del cliente en QUERY_STRING.
2. Ejecuta el programa solicitado.
3. Espera a que el programa finalice su ejecución y envíe de vuelta la salida generada.
4. Si la salida que le ha entregado el programa CGI no tiene una cabecera que le indique lo contrario, procederá a su análisis. (Un programa CGI que genera cabeceras que evitan el análisis de los datos que lleva asociados crea sus propias cabeceras HTTP, por lo que el servidor se ahorra tener que examinar las cabeceras.)
5. Crea las cabeceras HTTP que sean necesarias.
6. Envía las cabeceras y la salida del programa al cliente que lo ha pedido.

La figura 8.2 muestra el proceso.

Vamos a ver ahora qué es lo que tiene que hacer un programa CGI para recuperar la entrada y utilizarla con propósitos internos.

Tal y como se puede observar en la figura 8.3, un programa CGI hace lo siguiente:

1. Lee la variable REQUEST_METHOD.
2. Determina si se utiliza el método GET analizando el contenido de la variable REQUEST_METHOD.
3. Recupera los datos que se encuentran almacenados en la variable de entorno QUERY_STRING (si se utiliza el método GET).
4. Descodifica los datos.

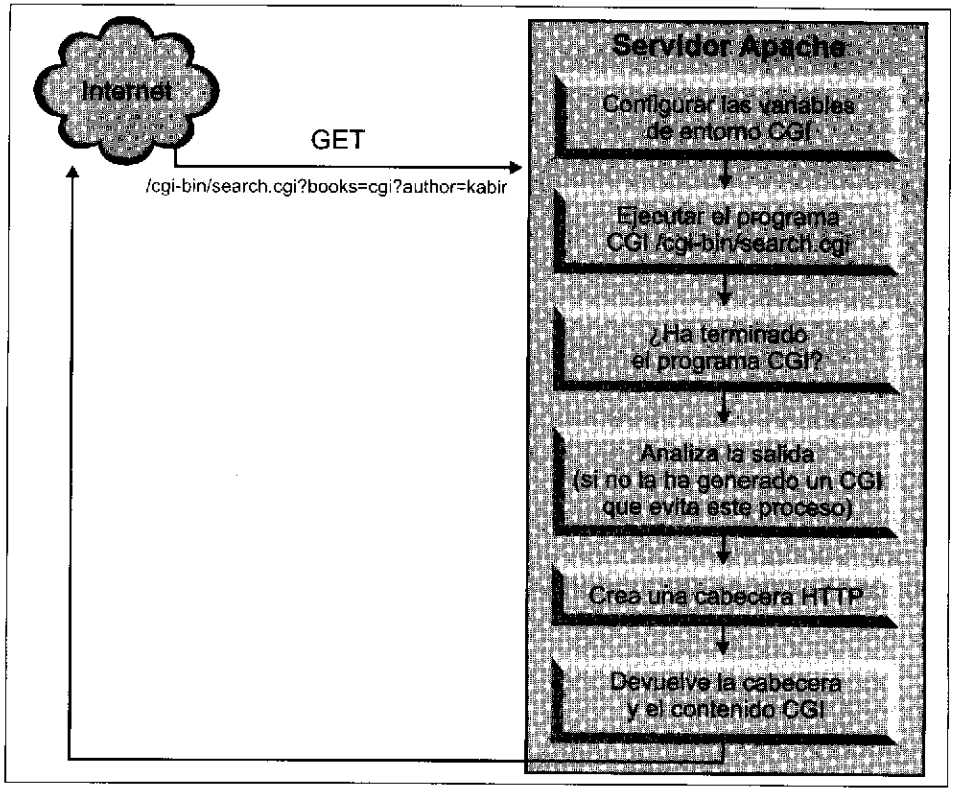


Figura 8.2. Diagrama del proceso de un servidor CGI

5. Llegados a este punto, el programa CGI puede procesar los datos descodificados si lo cree conveniente.
6. Después de que se complete el proceso, escribe entonces el tipo del contenido (Content-Type) de la salida en su dispositivo de salida estándar (STDOUT).
7. Por último, escribe los datos generados en STDOUT y después sale del proceso.

El servidor Web lee los datos STDOUT de la aplicación y los analiza para determinar el Content-Type de la salida. A continuación se le transmite al cliente las cabeceras HTTP apropiadas en las que se especifica el tipo de contenido, seguido de la salida generada. Se sale del programa CGI con lo que se completa la transacción.

En la figura 8.3 tiene un diagrama en el que se ilustra todo el proceso del programa CGI.

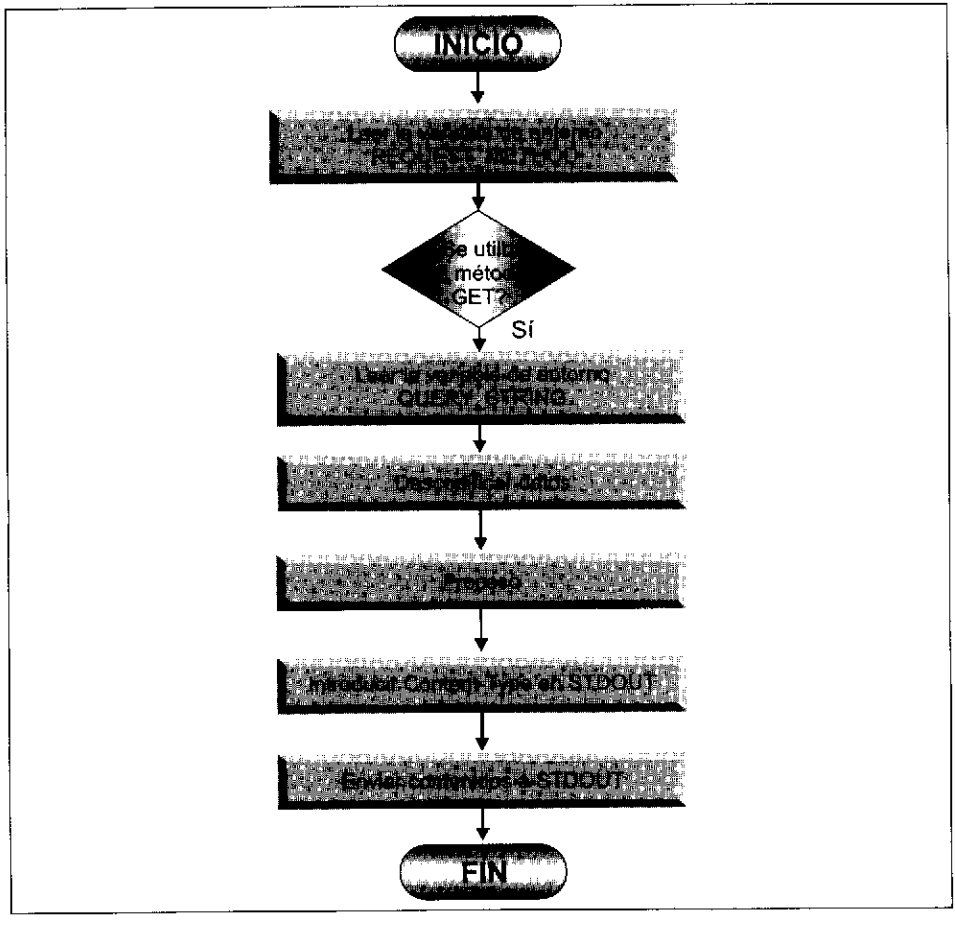


Figura 8.3. Diagrama del proceso de un programa CGI

Obsérvese que si un programa CGI proporciona las cabeceras CGI y la información sobre Content-Type necesarias, a su nombre habrá que aplicarle el prefijo nph (que indica "cabecera sin analizar"). El servidor Web no analizará una salida nph generada por un programa CGI, sino que la enviará directamente al cliente. La mayoría de programas CGI dejan que sea el servidor quien se encargue de escribir las cabeceras HTTP y, por lo tanto, de analizar las cabeceras de los programas.

La verdad es que el uso del método GET para pasarle los datos del cliente al programa CGI es muy limitado:

- El tamaño total de los datos que se pueden transmitir como parte del URL está limitado por la configuración del cliente. Los exploradores

Web más populares tienen serias limitaciones al respecto, por lo que la cantidad de datos que se pueden enviar en un URL está limitada. De todas formas, hay ocasiones en las que es conveniente entregarle datos al programa CGI a través del URL. Por ejemplo, si tiene un formulario HTML que utilice el método GET para enviar datos a un programa CGI, se puede guardar el URL para usarlo más tarde sin tener que pasar por el trámite del formulario. Puede ser una propiedad muy interesante para las aplicaciones tipo base de datos que suelen solicitar bastantes datos.

- La longitud del valor de una única variable de entorno (QUERY_STRING) está limitada. Muchos sistemas operativos(si no todos) tienen ciertas restricciones relacionadas con la cantidad de bytes que puede llegar a contener una variable de entorno. De este modo se limita el número de bytes que se pueden almacenar en los datos de entrada.

Es muy probable que estos límites no afecten aquellos programas CGI que apenas precisan que el cliente introduzca datos. Para los programas que necesitan que el cliente introduzca una gran cantidad de datos hay otro método: POST. Es el que veremos en la siguiente sección.

Peticiones POST

El método de petición HTTP POST es el más utilizado para la entrega de datos a programas CGI. El típico ejemplo del uso de este método lo tenemos en los formularios HTML que se rellenan a través de las páginas Web. El listado 8.1 le muestra uno:

```
<HTML>
        <HEAD>
        <TITLE> Biblia del Servidor Apache </TITLE>
      </HEAD>

<BODY>
<H1>Listado 8-1</H1>
<H2>Un ejemplo en el que se ve cómo utilizar la petición HTTP POST en
un formulario HTTP </H2>
<HR>
<FORM ACTION="/cgi-bin/search.cgi" METHOD="POST">

<PRE>
 Tipo del libro <INPUT TYPE-"TEXT" NAME="book" SIZE-"10"
MAXSIZE="20">
Nombre del Autor <INPUT TYPE="TEXT" NAME="author" SIZE="10"
MAXSIZE="20">
</PRE>
<INPUT TYPE=SUBMIT VALUE="Buscar Ahora">
```

```
</FORM>

</BODY>
</HTML>
```

Listado 8.1. Formulario HTTP que utiliza el método POST

Habrá observado que hay una sección <FORM...> ... </FORM> en el listado anterior. Un formulario HTML suele comenzar con una etiqueta FORM que define la acción (ACTION) que se ha de desarrollar y solicita un método del formulario (METHOD). En el ejemplo anterior, la acción es el programa CGI /cgi-bin/search.cgi y el método POST.

Inmediatamente después de la etiqueta <FORM> inicial suelen aparecer una o varias entradas INPUT. Estas entidades pueden incluir cajas de entrada de texto, menús desplegables y listas. En nuestro ejemplo aparecen tres entidades INPUT. El primero permite que el usuario introduzca el valor que se le asignará a la variable book (libro). El siguiente es muy parecido, sólo que se introduce el nombre del autor. Y el tercero es algo especial, porque permite que se envíe el formulario. En el momento en que se transmite el formulario, el software del cliente envía una petición POST al servidor solicitando un recurso ACTION (es decir, /cgi-bin/search.cgi) y transmite book=<valor introducido por el usuario> y author=<valor introducido por el usuario> en el formato adecuado.

Comparación de los métodos GET y POST

¿Qué diferencia hay entre los métodos GET y POST? Los datos que se envían con POST no se guardan en la variable de entorno QUERY_STRING de ningún programa CGI. En su lugar se guardan en la entrada estándar (STDIN) de dicho programa. La variable REQUEST_METHOD toma el valor POST, mientras que los datos codificados se almacenan en STDIN del programa CGI y se invoca una nueva variable de entorno llamada CONTENT_LENGHT en la que se guarda el número de bytes almacenados en STDIN.

El programa CGI comprueba el valor de la variable REQUEST_METHOD. Si toma el valor POST para las peticiones HTTP POST, el programa tendrá que comenzar por determinar el tamaño de los datos introducidos, información que se encuentra en la variable de entorno CONTENT_LENGTH, y después leer los datos de STDIN. Obsérvese que el servidor Web no es quien inserta la marca EOF (que determina dónde acaba el archivo), por eso se utiliza la variable CONTENT_LENGTH, con cuyo valor se determina la longitud de los datos (en bytes). Así, el programa CGI podrá determinar la cantidad total de bytes de datos con los que tiene que trabajar.

Se puede utilizar GET y POST a la vez. Aquí tenemos un ejemplo de un formulario HTML que oficialmente utiliza el método POST, pero que oficiosamente pasa username=joe como parte de la acción (ACTION) CGI.

```
<FORM ACTION="/cgi-bin/edit.cgi?username=joe" METHOD=POST>
  <INPUT TYPE=TEXT NAME="PhoneNumber">
  </FORM>
```

En este ejemplo, la petición username=joe puede formar parte del URL, pero el otro campo (Phone Number) tendría que formar parte de los datos POST. El resultado: el usuario final puede agregar el URL a su carpeta de favoritos y ejecutar el script edit.cgi siempre que lo desee como usuario joe sin tener que indicar cuál es el valor del resto de campos. Su uso es recomendable con las aplicaciones especializadas en las bases de datos y con los buscadores.

Siempre que se utilice GET o POST (o ambos), se le entregan los datos codificados al programa CGI para que sea él quien los descodifique. En la siguiente sección veremos el proceso relacionado con la descodificación de los datos.

Descodificación de los datos de entrada

Los diseñadores del protocolo HTTP original querían que la implementación de su creación en cualquier sistema fuese muy sencilla. Además simplificaron mucho los sistemas de codificación de datos.

Estos sistemas definen una serie de caracteres como caracteres especiales. Por ejemplo, el signo = se usa en casos como clave=par de valores. el signo + sustituye al espacio en blanco y & separa dos pares de valores.

Es posible que se pregunte cómo se transmiten los datos que contienen caracteres con significado especial. En este caso se utiliza un sistema de codificación basado en tres caracteres, que puede codificar cualquier carácter. Un signo % indica el inicio de una secuencia de caracteres que no se ha de codificar, secuencia que tendrá dos dígitos hexadecimales.

Nota: el sistema hexadecimal se compone de 16 números, donde 0-9 tienen el mismo valor que en el sistema decimal, pero hay una serie de caracteres extra: A(=10), B(=11), C(=12), D(=14), H(=15). Por ejemplo, 20 en sistema hexadecimal equivale a 32 en el decimal. La conversión se efectúa de la siguiente manera:

20 = 2*16 + 0*1

Estos dos dígitos hexadecimales serán el valor que se convierta en la tabla ASCII para hacerse con el carácter. Por ejemplo, %20 (hex) es 23 (decimal) y se convierte en el carácter ASCII del espacio en blanco.

Soporte CGI en Apache

Hay dos formas por las que Apache puede implementar el soporte para CGI. En la distribución estándar de Apache se incluye un módulo CGI que implementa soporte tradicional para CGI. De todas formas, hay un nuevo módulo (FastCGI) que implementa soporte para las aplicaciones CGI de alto rendimiento. En esta sección veremos las propiedades del soporte CGI estándar.

En los apartados anteriores hemos visto que los servidores Web que compilan CGI utilizan variables de entorno, STDIN y STCOUT para intercambiar información con los programas CGI. Apache proporciona una serie de variables de entorno lo suficientemente flexibles para que la utilicen los desarrolladores de programas CGI. Al utilizar estas variables, un programa CGI no se limita a recuperar los datos introducidos, sino que reconoce el tipo de cliente y de servidor con los que trabaja.

En las próximas secciones veremos las variables de entorno disponibles en el módulo CGI que se distribuye con la versión estándar de Apache.

Variables del servidor

Con estas variables el servidor Apache facilita información sobre sí mismo al programa CGI. Por medio de las variables del servidor un programa CGI podrá hacerse con distinta información sobre el servidor, como la versión del software de Apache, la dirección de correo electrónico del administrador, etc.

SERVER_SFOTWARE

La variable SERVER_SOFTWARE la configura el propio servidor Apache y su valor será de la siguiente forma:

```
Apache/Versión
```

En este caso, Apache es el nombre del software del servidor que ejecuta el programa CGI y la versión es el número de la versión de Apache. Por ejemplo, un caso real sería algo así:

```
Apache/1.3
```

Esto es especialmente útil cuando el programa CGI puede aprovecharse de alguna propiedad de las últimas versiones de Apache y aun así, seguir trabajando con versiones antiguas.

GATEWAY_INTERFACE le indica al programa CGI con qué versión de la especificación CGI puede trabajar el servidor. Un ejemplo sería:

```
CGI/1.1
```

Un programa CGI puede determinar el valor de esta variable y utilizar (en casos condicionales) diferentes propiedades según la versión de la especificación CGI con la que se trabaje. Por ejemplo, si el valor es CGI/1.0, el programa no podrá utilizar las propiedades de CGI/1.1 y viceversa.

Nota: el primer entero antes del punto decimal recibe el nombre de *número mayor* y el que se encuentra después del punto, *número menor*. Como a ambos enteros se les trata de forma distinta, CGI 2.2 será una versión más antigua que CGI/2.15.

SERVER_ADMIN

Si usa la directriz ServerAdmin en el archivo httpd.conf para determinar la dirección de correo electrónico del administrador del sistema, esta variable se hará con ella. Si coloca dicha directriz dentro del contenedor de algún servidor virtual, el valor de la variable SERVER_ADMIN cambiará según la parte del servidor a la que acceda el programa CGI.

DOCUMENT_ROOT

El valor de esta variable adopta el de la directriz DocumentRoot del sitio Web al que está accediendo.

Variables que solicita el cliente

Apache crea una serie de variables de entorno a partir de las cabeceras HTTP de las peticiones que recibe del cliente que solicita el programa CGI. Le suministra información al programa CGI a través de las siguientes variables de entorno.

SERVER_NAME

Esta variable le indica al programa CGI a qué servidor está accediendo. Su valor puede ser tanto una dirección IP como el nombre completo de un host:

```
SERVER_NAME = 206.171.50.51
SERVER_NAME = www.nitec.com
```

HTTP_HOST

Véase la variable SERVER_NAME.

HTTP_ACCEPT

Esta variable especifica una lista con los tipos MIME que puede aceptar el cliente:

```
HTTP_ACCEPT = image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
image/png, */*
```

En este caso, el cliente indica que puede trabajar con formatos GIF, JPEG, PNG y más. De esta forma, el programa CGI determinará en qué formato tendrá que generar la salida que le entregue al cliente. Por ejemplo, un programa CGI puede producir indistintamente archivos GIF o JPEG y recibe lo siguiente:

```
HTTP_ACCEPT = image/gif, */*
```

En este caso generará una imagen GIF antes que una JPEG de acuerdo con las preferencias del cliente.

HTTP_ACCEPT_CHARSET

Esta variable especifica con qué conjunto de caracteres puede trabajar el cliente. Por ejemplo:

```
HTTP_ACCEPT_CHARSET = iso-8859-1,*,utf-8
```

HTTP_ACCEPT_LANGUAGE

Con esta variable se especifica el idioma con el que trabaja el cliente. Por ejemplo:

```
HTTP_ACCEPT_LANGUAGE = en
```

En este caso el cliente puede trabajar con los caracteres ingleses.

HTTP_USER_AGENT

Esta variable especifica qué software y sistema operativo está utilizando el cliente. Por ejemplo:

```
HTTP_USER_AGENT = Mozilla/4.04 [en] (WinNT; I)
```

La información facilitada es la siguiente:

- Software del cliente: Netscape Navitagor (Mozilla es la palabra clave que utiliza Netscape).
- Versión del software del cliente: 4.04 (versión inglesa).
- Sistema operativo: Windows NT (Intel).

Observe que, aunque la palabra Mozilla se usa con los exploradores Web de Netscape, muchos otros vendedores la utilizan como parte de su cabecera. Por ejemplo, Microsoft IE 4.0 produce el siguiente HTTP_USER_AGENT cuando se utiliza en la misma máquina.

```
HTTP_USER_AGENT = Mozilla/4.0 (compatible; MSIE 4.0; Windows NT)
```

La información del agente del usuario se utiliza en muchos sitios Web. Un sitio que esté optimizado para trabajar con Netscape Navigator (es decir, que utiliza una serie de propiedades HTML o JavaScript que funciona especialmente bien con Netscape Navigator) podría utilizar esta información para mostrarle una página distinta a Microsoft IE o a cualquiera de los exploradores menos populares. De todas formas, mi consejo es que se ajuste a la especificación HTML estándar (que se puede encontrar en www.w3g.org) y no se ciña a las características propias de un explorador. Esto quiere decir que las etiquetas HTML propias de un navegador o de una serie de plug-ins pueden ser inapropiadas para otros clientes que visiten sus páginas Web y que no utilicen su explorador favorito.

HTTP_REFERER

Esta variable se utiliza para determinar el identificador URI que precede a la petición que solicita un programa CGI. Por medio de esta variable puede decir si una petición procede de un enlace de una de sus páginas Web o de un URI remoto.

HTTP_CONNECTION

Esta variable se utiliza para determinar el tipo de conexión que utilizan el cliente y el servidor. Por ejemplo:

```
HTTP_CONNECTION = Keep-Alive
```

De esta forma se determina que el cliente puede trabajar con conexiones permanentes a través de Keep-Alive, y que de hecho es lo que está utilizando.

SERVER_PORT

El valor de esta variable le indica al programa CGI el puerto que está utilizando el servidor para acceder al programa. Un ejemplo podría ser:

```
SERVER_PORT = 80
```

Si un programa CGI crea una serie de URL que apuntan al servidor, convendría añadir la dirección del puerto que se encuentran en esta variable.

REMOTE_HOST

Esta variable informa al programa CGI sobre la dirección IP o el nombre IP del cliente, tal y como se indica a continuación:

```
REMOTE_HOST = ppp-007.speedlink.net
```

Observe que si el servidor Apache se compila con la opción MINIMAL_DNS, no se podría utilizar esta variable.

REMOTE_PORT

Especifica el número del puerto que utiliza el cliente para originar la conexión a través de la cual se le enviará la solicitud al programa CGI.

```
REMOTE_PORT = 1163
```

De todas formas, aún no me he encontrado con ningún caso en el que se utilice esta variable.

REMOTE_ADDR

Dirección IP del sistema del cliente:

```
REMOTE_ADDR = 206.171.50.51
```

Si el cliente se encuentra detrás de un firewall o un servidor proxy, la dirección IP que se almacena en esta variable no tiene por qué ser la dirección IP del sistema del cliente.

REMOTE_USER

Sólo se utilizará cuando para acceder al programa CGI haya que utilizar una autenticación HTTP básica. El nombre de usuario usado se guarda en esta variable y se pone a disposición del programa CGI. De todas formas, este programa no tendrá ninguna forma de identificar la contraseña de acceso del usuario. Si el valor de este usuario es el nombre del usuario, el programa CGI

puede asumir sin ningún problema que el usuario ha utilizado la contraseña apropiada para acceder a él.

SERVER_PROTOCOL

Es el número de versión y protocolo que utiliza el cliente que envía la petición al programa CGI:

```
SERVER_PROTOCOL = HTTP/1.1
```

REQUEST_METHOD

En esta variable se guarda el método que utiliza el cliente para enviar la solicitud al programa CGI. Los valores típicos son GET, POST, HEAD, etc.

```
REQUEST_METHOD=GET
```

Cuando se utiliza el método GET la entrada se almacena en la variable QUERY_STRING y cuando se utiliza POST, se almacena en STDIN.

REQUEST_URI

A esta variable se le asigna el valor de la petición URI.

```
REQUEST_URI = /cgi-bin/printenv2
```

REMOTE_IDENT

Sólo se utilizará en el caso en que se determine la directriz IdentityCheck. La variable guarda la información sobre la identificación del usuario que le muestra el demonio identd del sistema remoto. Como muchos procesos no ejecutan este demonio, no es conveniente fiarse de él como método para identificar usuarios. Mi consejo es que se use en una intranet o un entorno externo en el que el usuario o su organización ejecuten el servidor identd.

AUTH_TYPE

El programa CGI se encuentra en una sección de la Red en la que hay que utilizar una autenticación para acceder. Esta variable determina el método de autenticación que se utiliza.

CONTENT_TYPE

Esta variable especifica el tipo MIME de los datos asociados a la cabecera de la petición. Por ejemplo:

```
CONTENT_TYPE = application/x-www-form-urlencoded
```

Cuando se utiliza un formulario HTML y el sistema POST para efectuar las solicitudes se puede especificar el tipo del contenido en el formato HTML por medio del atributo TYPE que se ha de colocar dentro de las etiquetas <FORM>:

```
<FORM ACTION="/cgi-bin/search.cgi" METHOD="POST" TYPE= "application/
x-www-form-urlencoded">
```

CONTENT_LENGTH

Cuando se utiliza el método POST, Apache almacena los datos introducidos (asociados a la petición) en el STDIN del programa CGI. Pero el servidor no coloca un marcador EOF en STDIN, sino que lee el valor de esta variable, que le indicará la cantidad de bytes. Por ejemplo:

```
CONTENT_LENGTH = 21
```

le indicaría al programa CGI que leyese 21 bytes de datos de la entrada estándar STDIN.

SCRIPT_NAME

El identificador URI del programa CGI solicitado:

```
SCRIPT_NAME    /cgi-bin/search.cgi
```

SCRIPT_FILENAME

Path de la ubicación física del programa CGI:

```
SCRIPT_FILENAME = /www/kabir/public/cgi-bin/search.cgi
```

QUERY_STRING

Si el cliente utiliza el método HTTP GET y entrega una serie de datos de entrada precedidos de un signo ?, los datos serán los que se cojan como valor de esta variable. Por ejemplo, una petición para el programa CGI:

http://apache.nitec.com/cgi-bin/search.cgi?key1=value1&key2=value2

hará que Apache efectúe el siguiente ajuste:

```
QUERY_STRING = key1=value1&key2=value2
```

con lo que el programa CGI /cgi-bin/search.cgi podrá leer y descodificar los datos antes de utilizarlos.

PATH_INFO

Si los datos que se introducen para que los procese el programa CGI son parte del URI, toda la información adicional al path real (que es donde realmente se encuentra el programa) se guardará dentro de esta variable. Por ejemplo:

```
http://apache.nitec.com/cgi-bin/search.cgi/argument1/argument2
```

hará que Apache efectúe el siguiente ajuste:

```
PATH_INFO = /argument1/argument2
```

Obsérvese que PATH_INFO no tomará nada que no sea parte de la cadena de la petición.

En otras palabras, si el identificador URI incluye la cadena de la petición detrás del signo ?, esta parte de los datos se almacenarán en la variable QUERY_STRING. Por ejemplo:

http://apache.nitec.com/cgi-bin/search.cgi/CA/95825?book=apache&author=kabir

forzará a Apache a ajustar las siguientes variables:

```
PATH_INFO = /CA/95825
QUERY_STRING = book=apache&author=kabir
```

PATH_TRANSLATED

Path absoluto del archivo solicitado.

Configurar el servidor Apache para CGI

En esta sección veremos cómo configurar Apache para que procese las peticiones CGI. El proceso de configuración incluye decirle a Apache dónde guarda los programas CGI, configurar los controladores CGI para que trabajen con ciertas extensiones de archivos e indicar de qué extensiones se tendrán que hacer cargo los programas CGI. Una buena idea consiste en guardar todos los programas CGI en un directorio CGI. Así podrá tener un mejor control sobre ellos. Si los tiene distribuidos por toda la red terminará por perderlos, con lo que acabará por producir una serie de agujeros en la seguridad de su sistema.

Dar un alias al directorio de los programas CGI

Establecer un directorio central para los programas CGI no es más que el primer paso para configurar un entorno CGI. Lo ideal es que este directorio se encuentre fuera de DocumentRoot para que no se pueda acceder directamente a dichos programas. ¿Por qué? Porque lo general es que al trabajar con programas CGI se desee suministrar la menor cantidad de información posible al mundo exterior. Así es más sencillo conservar intacta la seguridad del sistema. Cuanta menos gente tenga conocimientos sobre la ubicación de los programas CGI, menos daño podrán hacerle.

El primer paso que hay que dar es crear un directorio que se encuentre fuera de la carpeta DocumentRoot. Por ejemplo, si /www.miempresa.com/public/htdocs es el directorio DocumentRoot de un sitio Web, entonces /www.miempresa.com/public/cgi-bin/ sería un buen candidato para guardar los programas CGI. Para darle un alias a este directorio tendría que utilizar la directriz ScriptAlias.

Si configura CGI para que pueda trabajar con el servidor Web principal, tendrá que editar el archivo httpd.conf e insertar la línea ScriptAlias con la siguiente sintaxis:

```
ScriptAlias       /alias/     /path/
to/the/CGI/program/directory/ending/with/
```

Por ejemplo:

```
ScriptAlias       /cgi-bin/   /www/mycompany/public/cgi-bin/
```

Si configura el soporte de CGI para que trabaje con un servidor virtual, tendrá que añadir una línea ScriptAlias en cada uno de los contenedores de los servidores virtuales (VirtualHost). Por ejemplo:

```
NameVirtualHost 206.171.50.60
<VirtualHost 206.171.50.60>
ServerName blackhole.nitec.com
ScriptAlias       /apps/      /www/nitec/blackhole/public/cgi-bin/
</VirtualHost>
```

En este caso, se utiliza el alias /apps/ para crear un sobrenombre para el directorio de programas CGI. Si hay un programa CGI que se llame feedback.cgi y se encuentra en el directorio /www/nitec/blackhole/public/cgi-bin, sólo se podrá acceder a él a través de:

```
http://blackhole.nitec.com/apps/feedback.cgi
```

Después de definir la directriz ScriptAlias tendrá que asegurarse de que dicho directorio cuenta con los permisos de lectura y ejecución.

Advertencia: el directorio al que señala ScriptAlias ha de tener una serie de permisos muy restrictivos. Nadie, a excepción del desarrollador del programa CGI o el administrador del sistema, debería tener todos los permisos del directorio (lectura, escritura y ejecución). Recuerde que puede definir varios alias para los directorios en los que se encuentran los programas CGI y que no se puede navegar por el directorio especificado en ScriptAlias por razones de seguridad.

Siempre que se le pida, Apache tratará de abrir cualquier archivo ejecutable (si el archivo cuenta con los permisos pertinentes) que se encuentre en el directorio especificado en la directriz ScriptAlias. Por ejemplo:

```
http://blackhole.nitec.com/apps/foo.cgi
http://blackhole.nitec.com/apps/foo.pl
http://blackhole.nitec.com/apps/foo.bak
http://blackhole.nitec.com/apps/foo.dat
```

Todos estos URL le pedirán al servidor Apache que trate de ejecutar los archivos foo.

La verdad es que no estoy completamente de acuerdo con que se pueda ejecutar cualquier archivo que se encuentre dentro del directorio especificado en ScriptAlias. Particularmente me gusta más la opción que me permite restringir los nombres de los programas CGI de tal forma que únicamente se puedan ejecutar los archivos con ciertas extensiones. En la próxima sección veremos cómo se puede desarrollar esta idea utilizando Apache Handler, un controlador que se encuentra en el módulo mod_cgi.

Seleccionar extensiones específicas de archivos CGI

En esta sección le voy a mostrar un ejemplo de configuración que me permite seleccionar una serie de extensiones de archivos para que el servidor los trate como si fuesen programas CGI. Para ello se usa el controlador AddHandler.

Supongamos que el nombre del servidor Apache es www.nitec.com y que el directorio DocumentRoot es /www/nitec/public/htdocs. El directorio de los programas CGI es /www/nitec/public/cgi-bin. Obsérvese que el directorio en el que se guardan los programas CGI se encuentra fuera del directorio

especificado en DocumentRoot. Es algo intencionado. De esta forma nos aseguramos que nadie puede explorar el directorio y que Apache únicamente accederá a él a través del alias.

Paso1. Eliminar/desactivar ScriptAlias

Lo primero que tiene que hacer es eliminar cualquier directriz ScriptAlias, ya sea quitándola del archivo de configuración (httpd.conf) o convirtiéndola en comentario insertando un signo # como primer carácter de la línea.

Paso 2. Crear un alias para el directorio de programas CGI

No hay ninguna forma de acceder al directorio de programas CGI si no se define un alias (o un enlace simbólico), ya que se encuentra fuera del árbol de directorios. Puede definir un alias usando la directriz Alias por medio de la siguiente sintaxis:

```
Alias /alias/ /path/to/cgi/dir/outside/doc/root/
```

A continuación hay que escribir la directriz en la que se define el alias:

```
Alias /cgi-bin/ /www/nitec/public/cgi-bin/
```

Ahora ha de instruir a Apache para que ejecute los programas CGI que se encuentren en este directorio. Para ello habrá que definir un contenedor <Directory...> para este directorio.

Paso 3. Definir el contenedor del directorio para el directorio de programas CGI

La definición del contenedor para el directorio de programas CGI (es decir, convertir un directorio normal y corriente en un directorio de programas CGI) se hace de la siguiente manera:

```
<Directory /path/to/cgi/dir/outside/doc/root>
Options ExecCGI
AddHandler cgi-script .extension .extension ...
</Directory>
```

Este contenedor de directorio le da una serie de instrucciones a Apache. En primer lugar nos encontramos Options ExecCGI que le dice a Apache que permita la ejecución de los programas CGI que se encuentren en este directorio. En segundo lugar aparece el controlador AddHandler cgi-script .extension .extension ... que le indica a Apache que considere las extensiones que aparecen en la lista como las de los programas CGI (en otras palabras, siempre que Apache se encuentre con un URL que le pide un archivo cuya extensión aparezca en la lista, lo ejecutará como si de tratase de un programa CGI).

La definición del contenedor actual tendría el siguiente aspecto:

```
<Directory /www/nitec/public/cgi-bin>
Options ExecCGI
AddHandler cgi-script .cgi  .pl
</Directory>
```

Se le ordena a Apache que tome los archivos cuyas terminaciones sean .cgi y .pl como extensiones de programas. Por lo tanto, siempre que se efectúe una petición:

```
www.nitec.com/cgi-bin/anything.cgi

www.nitec.com/cgi-bin/anything.cgi.pl
```

Apache tratará de ejecutar estos archivos como programas CGI. Si estos archivos no son ejecutables, Apache mostrará un mensaje en el que se indicará que es posible que haya un error.

Nota: los permisos del directorio en el que se encuentran los programas CGI de los que hemos hablado con anterioridad aún se aplican en esta configuración. Ocurre lo mismo con los servidores virtuales.

Permitir que sus usuarios tengan acceso cgi-bin

Muchos proveedores de Internet (ISP) ofrecen sitios Web junto con las cuentas de los usuarios. Sus URL suelen ser del estilo:

```
http://www.isp.net/~nombre_usuario
```

Y suelen recibir peticiones de acceso cgi-bin procedentes de estos mismos usuarios. El término "acceso cgi-bin" se utiliza para indicar que un servidor Web tiene una facilidad CGI. Tradicionalmente, el directorio en el que se guardan los programas CGI es /cgi-bin/, por eso se creó dicho término. Otro término que se ha hecho muy popular es "página", con el que se hace referencia al índice que se encuentra en el nivel superior del árbol de directorios de un usuario.

En esta sección, veremos dos formas de suministrar acceso cgi-bin a los usuarios de un servidor Apache. Todo lo que tiene que hacer es implementar uno de los dos métodos en su sistema.

Uso de los contenedores Directory o DirectoryMatch

Cuando el valor de la directriz UserDir es el nombre de un directorio, Apache lo considera como el directorio del nivel superior del sitio Web de un usuario. Por ejemplo:

```
ServerName www.yourcompany.com
UserDir public_html
```

Ahora bien, cuando se recibe una petición dirigida a www.suempresa.com/~nombre_usuario, Apache localiza el directorio de dicho usuario (generalmente revisando el contenido del archivo /etc/passwd de los sistemas UNIX) y le asigna a UserDir el nombre de dicho directorio. De esta forma se crea el nombre del path del nivel superior de un usuario. Por ejemplo:

```
www.suempresa.com/~joe
```

Con esta línea Apache buscará /home/joe/public_html (suponiendo que el directorio principal de joe sea /home/joe). Si lo encuentra, le enviará al cliente una página con el índice.

Una forma de añadir el soporte CGI a cada usuario es agregar estas líneas en alguno de los archivos de configuración de Apache:

```
<Directory ~ "/home/[a-z]+/public_html/cgi-bin">
Options ExecCGI
AddHandler cgi-script .cgi .pl
</Directory>
```

O, si utiliza la última versión del servidor Apache, puede utilizar la siguiente configuración:

```
<DirectoryMatch  "/home/[a-z]+/public_html/cgi-bin">
Options ExecCGI
AddHandler cgi-script .cgi .pl
</DirectoryMatch>
```

Con ambos métodos Apache transformará las peticiones dirigidas a www.suempresa.com/~nombre_usuario/cgi-bin a /home/nombre_usuario/public_html/cgi-bin y permitirá la ejecución de cualquier programa CGI que tenga la extensión adecuada.

Obsérvese que todos los nombres de los usuarios han de aparecen en letras minúsculas para que este método funcione. Si los nombres de los usuarios tienen caracteres alfanuméricos, tendrá entonces que utilizar una expresión diferente.

Uso de la directriz ScriptMatch

Por medio de la directriz ScriptAlias también podrá trabajar con los directorios de programas CGI de cada usuario. Por ejemplo:

```
ScriptAliasMatch ~([a-z]+)/
cgi-bin/(.*)/home/$1/public_html/cgi-bin/$2
```

Esta directriz envía el nombre a la variable $1, que es igual a ~([a-z]+). Todo lo que aparezca detrás de /cgi-bin/ lo mete en la variable $2, donde $2 es igual a (.*). A continuación utiliza las variables $1 y $2 para determinar la localización actual del directorio de programas CGI. por ejemplo:

```
http://www.yourcompany.com/~joe/
cgi-bin/ search.cgi?book=dummies&author=kabir
```

Aquí ~([a-z]+) introducirá uno o más caracteres escritos en minúsculas a los que siga el símbolo ~ en $1. En otras palabras, la pareja de paréntesis nos permite capturar todo lo que se encuentra entre ~ y la barra inclinada que se encuentra detrás del nombre de usuario. En el ejemplo que nos ocupa tenemos que el valor de $1 es kabir y (.*) nos permite asignar todo lo que se encuentra detrás de cgi-bin/ y el par de paréntesis a la variable $2. Así pues, $2 será search.cgi?book=dummies&author=kabir.

Ahora Apache puede determinar el path físico que conduce al directorio de programas CGI con:

```
/home/$1/public_html/cgi-bin/$2
```

Esta expresión regular se ha obtenido del path del ejemplo anterior:

```
/home/kabir/public_html/
cgi-bin/search.cgi?book-dummies&author=kabir
```

Como search.cgi se encuentra en este directorio, se procederá con su ejecución.

Si no le gusta que el directorio de programas CGI se encuentre dentro de public_html, puede colocarlo fuera:

```
ScriptAliasMatch ~([a-z]+)/cgi-bin/(.*)     /home/$1/cgi-bin/$2
```

De esta forma se tomará:

```
www.yourcompany.com/~joe/
cgi-bin/feedback.cgi?book=dummies&author=kabir
```

y se convertirá en un archivo físico:

```
/home/kabir/cgi-bin/search.cgi?book=dummies&author=kabir
```

Por supuesto, si tampoco le preocupa demasiado dejar el directorio de programas CGI dentro de una carpeta a la que el usuario tenga acceso de lectura, puede crear una partición Web (o un directorio) para cada usuario en la que coloque su página principal y expresiones regulares.

Crear nuevas extensiones CGI con la directriz AddType

Ya sabe cómo crear extensiones de programas CGI con la directriz AddHandler. De todas formas, si quiere crear nuevas extensiones para los programas CGI en un directorio determinado, tendrá que utilizar el archivo .htaccess (o el archivo especificado en la directriz AccessFileName).

Antes de añadir nuevas extensiones con archivo de control de acceso por directorio (.htaccess), tendrá que crear un contenedor para el directorio como sigue:

```
<Directory /path/to/your/directory>
Options ExecCGI
AllowOverride FileInfo
</Directory>
```

La primera de las directrices que aparece en el contenedor le dice al servidor Apache que ha de permitir la ejecución de programas CGI en dicho directorio. La segunda le indica que la propiedad FileInfo se encuentra en el archivo de control de acceso del directorio (.htaccess). Esta propiedad nos permite utilizar la directriz AddType dentro del archivo que controla el acceso a un directorio (.htaccess).

Para añadir una nueva extensión a los programas CGI (.wizard) todo lo que tiene que hacer es crear un nuevo archivo .htaccess (o usar el nombre que haya especificado en la directriz AccessFileName) en el directorio en el que desea trabajar. Escriba lo siguiente:

```
AddType application/x-httpd-cgi .wizard
```

Ahora, vuelva a nombrar el programa CGI que se encuentra dentro del directorio para que tenga la extensión .wizard y solicítelo a través de su explorador. Asegúrese de que todos los archivos del directorio cuentan con los permisos de lectura y ejecución de Apache.

Ejecutar programas CGI

Hasta ahora hemos visto las especificaciones básicas de CGI, cómo trabaja Apache con ellas y cómo configurar el servidor Apache para que ejecute programas CGI. Si es administrador de un servidor Apache, las opciones que tiene serán configurar los programas CGI o bien crearlos usted mismo. Recuerde que este libro no trata sobre la programación en CGI, por lo que no profundizaremos en este campo. La intención de esta sección es revelarle unos cuantos aspectos sobre los programas CGI que le ayudarán a administrar un servidor Apache que pueda trabajar con sitios Web.

Nota: muchos de los ejemplos que aparecen en esta sección usan Perl. Si no lo tiene instalado en su sistema, sepa que puede bajarse un formato binario de la Red (www.perl.com). Tenga en cuenta que siempre que sea posible conviene compilar nuestros propios archivos binarios antes que confiar en que funcionen correctamente los compilados por extraños.

Un sencillo programa en CGI

Perl es uno de los lenguajes de programación más utilizados en el desarrollo de programas CGI. Como es un lenguaje de interpretación, a los programas que crea se les conoce con el nombre general de *scripts*. En el listado 8.2 tiene un pequeño programa escrito en este lenguaje.

```
#!/usr/local/bin/perl

print "Content-type: text/html\n\n";
print <<HTML;

<HTML>
<HEAD>
<TITLE>Listado 8-2: Ejemplo de un programa CGI basado en Perl
  </TITLE>
</HEAD>
<BODY>
<CENTER>
<H1> Hello World </H1>
</CENTER>
</BODY>
</HTML>

HTML
```

Listado 8.2. helloworld.pl

Cuando se ejecuta este programa se obtiene una pantalla similar a la que aparece en la figura 8.4.

Figura 8.4. Salida de helloworld.pl

Si mira con detenimiento el script helloworld.pl verá que la mayor parte se encuentra escrito en texto HTML plano. De todas formas el script se puede simplificar un poco más para su análisis:

```
#!/usr/local/bin/perl

print "Content type: text/html\n\n";

print <<HTML
HTML TEXT GOES HERE
HTML
```

La primera línea es muy importante. Cuando Apache ejecuta este script, la primera línea le indica al sistema que se ha programado en Perl, por lo que se utiliza el intérprete de Perl /usr/local/bin/perl. De esta forma se le permite al sistema que ejecute este intérprete y que le entregue el archivo. Obsérvese que este método de informar al sistema sobre el intérprete se utiliza con todos los lenguajes de programación. En el ejemplo siguiente se muestra la versión script Shell C de helloworld.pl.

```
#!/bin/csh
echo "Content-type: text/html"
echo ""
echo "<HTML>"
echo "HTML TEST GOES HERE"
echo  "</HTML>"
```

Como puede observar, la primera línea es muy parecida a la del script en Perl. Aquí, el path del intérprete es /bin/csh/, de modo que cuando se baje un script gratuito de la Red asegúrese de que en la primera línea aparece correctamente el path del intérprete de comandos de su sistema. Es lo primero que se suele hacer.

Ahora nos vamos a fijar en el listado del script en Perl y veremos que la primera línea imprime Content-Type:text/html\n\n, que quiere decir que en una línea imprime Content-Type:text/html y luego deja otra en blanco (en Perl /n significa "nueva línea"). Es absolutamente necesario. Recuerde que antes hemos comentado que los script CGI tiene que suministrar la información de la cabecera al servidor Apache para que pueda crear la correspondiente en HTTP. La cabecera Content-Type le indica a Apache que la salida contiene datos de texto de tipo HTML. La línea en blanco separa la declaración del resto de datos, por eso se requiere su presencia. El shell script hace exactamente lo mismo pero con la ayuda de una declaración echo vacía. Lo que sigue a esta cabecera en texto HTML.

Advertencia: no es conveniente dejar que se pueda acceder desde Internet a un intérprete de comandos como Perl. Mucha gente lo hace y es un error. Como Perl y otros intérpretes de lenguajes de interpretación tienen muchas propiedades muy potentes relacionadas con la ejecución de comandos del sistema, es posible que algún pirata, si puede acceder a él desde la Red, ataque el sistema. Por eso se suele mantener apartado de los directorios generales dedicados a la Red.

Un script muy útil

Acabamos de ver un programa CGI muy simple que apenas hace nada. En esta sección vamos a ver un script algo más útil que podrá utilizar haciendo un par de modificaciones.

Pero antes de continuar me gustaría remarcar qué propiedades ha de tener un buen programa CGI:

- Utiliza plantillas HTML en vez de código HTML. Una plantilla HTML suele ser un archivo de texto con etiquetas HTML y otras personalizadas. Estas últimas se sustituyen por los datos del programa CGI. Se aconseja el uso de estas plantillas porque así los programadores HTML y los grafistas pueden efectuar las modificaciones que deseen para que su aplicación consiga el aspecto deseado, todo ello sin necesitar la asistencia de un desarrollador CGI.
- Utiliza el mecanismo apropiado para se puedan ejecutar varias copias de una misma aplicación simultáneamente. Por ejemplo, si un programa CGI escribe un archivo en una ubicación determinada, tendrá que tener en cuenta que es posible que se esté ejecutando más de una copia del

programa CGI, por lo que tendrá que tener cuidado al controlar la situación. Siempre que sea posible, conviene utilizar alguna técnica de bloqueo de archivos.

- Utiliza rutinas de librerías y paquetes externos. De esta forma el mantenimiento del programa es más sencillo.

Independientemente de si un administrador de un sistema Apache es el responsable de la creación de programas CGI, sí que será el encargado de su mantenimiento. Es decir, tendrá que trabajar a un nivel en el que se le permita modificar los programas CGI, su configuración e instalación. He trabajado con muchos servidores Apache que eran los responsables de la organización de sus sitios Web y he encontrado que su visión de CGI les ha ayudado mucho a la hora de tomar decisiones relacionadas el desarrollo de programas CGI.

Ahora vamos a echar un vistazo a un programa CGI para que lo examine y modifique. Como la mayoría de los programas CGI procesan los datos que introduce el usuario a través de formularios HTML, le mostraré como ejemplo un programa CGI escrito en Perl que procesa formularios HTML. En el listado 8.3 tiene su código.

```
#!/usr/local/bin/perl
#
# Nombre del Script: wizform.cgi
#
# Propósito: es un script especializado en el proceso de datos que
# se encuentran en formularios HTML y almacenarlos en un fichero
# y/o enviarlo a otras personas a través del correo electrónico.
#
# Requisitos: el formulario HTML ha de tener una variable
# HIDDEN (oculta) llamada conf_file a través de la cual se
# determina el nombre del archivo de configuración que se tiene
# que guardar en el mismo archivo que el script.
#
# Información sobre el Control de la Versión:
# $Id$
# $Author$
# $Revision$
# $Status$
# $Lock$
#
# Copyright (C) 1997 Mohammed J. Kabir (kabir@nitec.com)
#
##############################################################
# Le dice a Perl que quiero utilizar el paquete CGI.pm
# con este script.
use CGI;

# Crea un objeto CGI global llamado $cgi
local $cgi = new CGI;
```

```
# Toma el nombre del formulario HTML del archivo de configuración
# y lo guarda en una variable llamada
# $conf_file.
my $conf_file = $cgi->param('conf_file');

# Toma el nombre de la página de texto del formulario HTML (si la
# hay) y la guarda en una variable llamada
# $next_template.
my $next_template = $cgi->param('next_form');

# Otras declaraciones de variables locales.
my $key;
my $value;

# Guarda los datos de configuración en una array (hash) asociada.
local %conf;

# Imprime las cabecera Content-Type necesarias.
# Como tan sólo se muestra la salida del archivo de texto HTML,
# por defecto debería ser text/html.
print $cgi->header();

# Lee el archivo de configuración al que señala la variable
# $conf_file y lo guarda en el fichero de configuración especificado
# en de la array asociada.
#
&readConf($conf_file);

# Si la variable $next_template (que es la que se ha guardado el
# nombre del archivo del siguiente formulario HTML) está vacía, no
# habrá más formularios HTML que procesar y se completará la
# ejecución del formulario HTML llamando a la subrutina processData.
if($next_template eq ''){
    &processData;
    }

# Tenemos que hacer uno o más formularios HTML para procesar
# por lo que mostraremos el siguiente e introduciremos más datos
# a través de la subrutina getInput.
else {
    &getInput($next_template);
    }

# Ahora salimos del script.
exit 0;

sub processData{
   # Propósito: esta subrutina se encarga de procesar los datos.

   # una array para guardar el contenido de los campos de datos.
   my @saving_order = split(/\,/,$conf{'saving_order'});

   # una variable en la que se guarda el contenido de los campos o
   # el nombre
   my $datum;
```

```
    # una array en la que se guardan todos los errores.
    my @errors;

    # una variable en la que se guardan los registros que van a ir a
    # un fichero.
    my $record;

    # una variable en la que se guardan los mensajes de correo
    my $mail_msg;

    # cuenta el número de métodos de salida desarrollados con éxito.
    my $output_successful = 0;

    # guarda la cuenta de los métodos de salida solicitados.
    my $output_type = 0;

    # Comprueba las entradas perdidas (requerido) llamando a la
    # subrutina checkInputs que genera una lista HTML con los
    # nombres las variables perdidas.

    # Almacena la lista en la array @errors.
    @errors = &checkInputs;

    # Si ha habido algún error (por ejemplo, entradas perdidas)
    # habrá que mostrar una página con el mensaje de error
    if ($#errors > -1){
        &showErrors(@errors);
        return 1;
        }

    # Permite la creación de los mensajes de mail y los registros de
    # datos que hay que enviar para su almacenamiento.
    # Así que pase por todos los que se encuentran en
    # @saving_order y añádalos a las variables $record
    # y $mail_msg, en ese orden. Utilice el separador que se ha
    # especificado en el archivo de configuración.
    foreach $datum (@saving_order){
         $record .= $cgi->param($datum) . $conf{'field_separator'};
         $mail_msg .= ucfirst($datum) . "\t=\t" . $cgi->param($datum)
. "\n";
    }
      # Elimine el último separador de campo del bucle.
      $record =~ s/$conf{'field_separator'}$//;

      # Si en el archivo de configuración se solicita que se envíe
      # la salida por mail, habrá que enviar un mensaje de correo
      # electrónico a todas las personas que aparezcan en una lista
      # (que se encuentra en el archivo de configuración).
      # Se guarda un 1 ó un 0 (dependiendo de si la rutina sendMail
      # ha tenido éxito) en la variable $output_successful.
      # Además se incrementa el número de tipos que hay que procesar.
      if($conf{'output_method'} =~ /mail/i){
            $output_successful    &sendMail($mail_msg);
          $output_type++;
         }
```

```
    # Si hay que guardar la salida en un fichero (especificado
    # en el archivo de configuración), tendrá que crear un registro
    # para el fichero especificado.
    # Se incrementa la variable $output_successful con cualquier
    # valor que devuelva appendFile. Además se incrementa el número
    # de los tipos de salidas que hay que procesar.
    #
    if($conf{'output_method'} =~ /file/i){
       $output_successful += &appendFile($record,$conf{'data_file'});
$output_type++;
       }

    # Ahora, para ver si tenemos éxito en la producción de
    # los tipos de salida, tenemos que comparar las variables
    # $output_successful $output_type. Si coinciden habremos
    # tenido éxito. En caso contrario, el valor de la variable
    # $output_type será el mayor y habrá que informar del error
    # por medio de una página de fallos (especificada en el archivo
    # de configuración.)
    #
    if($output_successful >0 && $output_successful == $output_type){
       &showPage($conf{'success_template'});
       }
    else{
       &showPage($conf{'failure_template'});
       }

    # Por último, escribimos el archivo de registro.
    &writeLog;
    }

sub writeLog{
    # Propósito: esta subrutina escribe la dirección IP del cliente,
    # la fecha, la hora, el URL, etc. en el archivo de registro.

   # Declaraciones de variables locales
   my $log_file;
   my $record;
   my $status;

   # Tenemos que crear una cadena en la que se incluya
   # la fecha y la hora actual. Utilizamos la función localtime().
   my (@tefields) = localtime(time);
   my $sec          = $datefields[0];
   my $min          = $datefields[1];
   my $hr           = $datefields[2];
   my $mday         = $datefields[3];
   my $mon          = $datefields[4]   + 1;
   my $year           $datefields[5];
   my $today = sprintf("%02d:%02d:%02d %s/%s/
%s",$hr,$min,$sec,$mon,$mday,$year);

   # Crea el registro que queremos guardar en el archivo de registro
   $record = sprintf("%s, %s, %s, %s, %s",
                     $today,
```

```
                        $ENV{'SERVER_NAME'},
                        $ENV{'REMOTE_ADDR'},
                        $ENV{'HTTP_USER_AGENT'},
                        $ENV{'HTTP_REFERER'}
                      );

    # Escriba la entrada en el archivo de registro especificado en
    # el fichero de configuración.
    $status = &appendFile($record,$conf{'log_file'});

    # devuelve un estado de éxito o de error al intento de escritura.
    return ($status);
    }

sub sendMail{
    # Propósito: esta subrutina envía un mail a las direcciones que
    # aparecen en la lista especificada en el archivo de
    # configuración.

    # Toma los datos (cuerpo del mensaje de correo electrónico) de
    # la lista de parámetros que se le ha entregado.
    my $data = shift;

    # Toma FQPN del demonio de correo del archivo de configuración.
    my $sendmail = $conf{'mail_daemon'};

    # Toma la lista de destinatarios que se encuentra en el archivo
    # de configuración.
    my $to = $conf{'mail_to'};

    # Toma el nombre del remitente del archivo de configuración.
    my $from = $conf{'mail_from'};

    # Toma el nombre del asunto del archivo de configuración.
    my $subject = $conf{'mail_subject'};

    # Abre un PIPE directamente con el demonio de mail.
    # Si se falla al establecer la conexión con el demonio, se
    # muestra un estado de error.
    open(MAIL,"|$sendmail -t") || return 0;

    # Escribe un mensaje al PIPE
    print MAIL <<DATA;
To: $to
From: $from
Subject: $subject

    $data
DATA

# Cierra la conexión
    close(MAIL);

    # Se ha completado la transmisión del mail y se obtiene el estado
    # de éxito.
```

```
    return 1;
    }

sub showPage{
    # Propósito: esta subrutina muestra una página HTML

   # Toma el nombre de la página de una lista de parámetros
    my $file = shift;

    # Variables locales
    my @html;
    my %vars;
    my $key;
    my $value;

    # Crea una array asociada a las entradas del formulario HTML
    # de las que se conoce el valor.
    foreach $key ($cgi->param()){
       $value = $cgi->param($key);
       $var{$key} = $value;
       }

    # Crea FQPN para la página HTML que se desea mostrar.
    $file = sprintf("%s/%s/
%s",$conf{'base_dir'},$conf{'template_dir'}, $file);

    # Lee el texto de la página que se encuentra dentro del buffer
    # local (una array)
    @html = &readFile($file);

    # Antes de imprimir cada una de las líneas que se encuentra en la
    # página HTML, sustituye los datos personalizados reemplazables
    # que suelen estar en formato <VISITOR-STRING>.
    foreach $line (@html){
      $line =~ s/<(VISITOR-\w+)>/ucfirst($var{$conf{$1}})/eg;
      print $line, "\n";
      }

    }

sub appendFile{
     # Propósito: esta subrutina añade una línea a un archivo.
     # Toma el registro, de la lista de parámetros, que se desea
     # añadir al archivo.
     my $record = shift;

     # Toma el nombre del fichero de los parámetros.
     my $filename = shift;

     # Crea el FQPN del archivo
     $datafile = sprintf("%s/%s/
%s",$conf{'base_dir'},$conf{'data_dir'},$filename);

     # Abre el archivo que está listo para que se le añadan líneas.
     # Si falla el intento se envía el estado de fallo (0)
```

```
    open(FP,">$datafile") || return 0;

    # ***  Zona Crítica ***
    # Trata de bloquear en exclusiva el archivo
    flock FP, 2;

    # Escribe el registro en el archivo.
    print FP $record, "\n";

    # Deja de bloquear el fichero
    flock FP, 8;

    # ** fin de la Zona Crítica ***

    # Cierra el archivo.
    close(FP);

    # como la escritura del archivo ha tenido éxito, se presenta el
    # estado correspondiente.
    return 1;
    }

sub checkInputs{
  # Propósito: comprueba las entradas de los campos perdidos.
  # Los campos solicitados son los que se encuentran en el archivo
  # de configuración.

# Obtiene una lista de los campos solicitados del archivo de
# configuración.
  my @required_data = split(/\,/,$conf{'required_fields'});

  # Variables locales
  my $datam;
  my $key;
  my $value;
  my @errors;

# Repasa cada uno de los campos solicitados para ver si ha recibido
# algún valor. Si no es así, pone el nombre de la variable en una
# lista de errores.
  foreach $datam (@required_data){
     $value = $cgi->param($datam);
     if($value eq ''){
       push(@errors,$datam);
       }
     }

  # Devuelve la lista de errores.
  return (@errors);
  }

sub getInput{
   # Propósito: Muestra un formulario HTML pero antes inserta todas
   # las entradas que ha recopilado de otros formularios a través
   # de variables ocultas.
```

```
    # Toma el nombre del archivo en el que se encuentra el
    # formulario HTML que se desea mostrar
    my $html_template = shift;

    # Variables locales
    my $hidden_data;
    my $key;
    my $value;
    my @html;

    # Crea FQPN para la plantilla
    $html_template = sprintf("%s/%s/%s",$conf{'base_dir'},
    $conf{'template_dir'},$html_template);

   # Crea una lista de datos oculta;
      foreach $key (Scgi->param()){
          next if ($key =~ /next_form/i);
          $value = $cgi->param($key);
          $hidden_data .= <<HIDDEN;
          <INPUT TYPE="HIDDEN" NAME="$key" VALUE="$value">
HIDDEN
          }

    # Se hace con el texto de la página leyendo los datos
    # almacenados en un buffer.
    @html = &readFile($html_template);

    # si el buffer tiene datos, se obtendrá el texto de la página.
    # Así que, se imprime la página y se sustituyen las etiquetas
    # <HIDDEN DATA> con los valores de la variable $hidden_data.
    if($#html > -1){
       foreach $line (@html){
          $line =~ s/<HIDDEN-DATA>/$hidden_data/g;
          print $line,"\n";
          }
       }

     # Por otro lado, si no hay texto en esta página, habrá un
     # problema de acceso, por lo que se mostrará un mensaje de
     # error, para el cual se utiliza el valor failure_template que
     # se encuentra en la configuración.
     else{
        &showPage($conf{'failure_template'});
       }
    }

sub readConf{
   # Propósito: esta subrutina lee el archivo de configuración y
   # guarda sus datos en una array asociada.
   # Toma el nombre del archivo de configuración
   my $conf_file = shift;

   # Variables locales
   my @buffer;
   my $line;
```

```
    # Toma el nombre del archivo en el que se encuentra el
    # formulario HTML que se desea mostrar
    my $html_template = shift;

    # Variables locales
    my $hidden_data;
    my $key;
    my $value;
    my @html;

    # Crea FQPN para la plantilla
    $html_template = sprintf("%s/%s/%s",$conf{'base_dir'},
    $conf{'template_dir'},$html_template);

   # Crea una lista de datos oculta;
      foreach $key (Scgi->param()){
          next if ($key =~ /next_form/i);
          $value = $cgi->param($key);
          $hidden_data .= <<HIDDEN;
          <INPUT TYPE="HIDDEN" NAME="$key" VALUE="$value">
HIDDEN
          }

    # Se hace con el texto de la página leyendo los datos
    # almacenados en un buffer.
    @html = &readFile($html_template);

    # si el buffer tiene datos, se obtendrá el texto de la página.
    # Así que, se imprime la página y se sustituyen las etiquetas
    # <HIDDEN DATA> con los valores de la variable $hidden_data.
    if($#html > -1){
       foreach $line (@html){
          $line =~ s/<HIDDEN-DATA>/$hidden_data/g;
          print $line,"\n";
          }
       }

     # Por otro lado, si no hay texto en esta página, habrá un
     # problema de acceso, por lo que se mostrará un mensaje de
     # error, para el cual se utiliza el valor failure_template que
     # se encuentra en la configuración.
     else{
        &showPage($conf{'failure_template'});
       }
    }

sub readConf{
   # Propósito: esta subrutina lee el archivo de configuración y
   # guarda sus datos en una array asociada.
   # Toma el nombre del archivo de configuración
   my $conf_file = shift;

   # Variables locales
   my @buffer;
   my $line;
```

```
    # Toma los datos del archivo y los guarda en el buffer local.
    @buffer = &readFile($conf_file);

    # Recorre todas las líneas del buffer y localiza los datos válidos
    # de configuración e ignora las líneas en blanco y comentarios que
    # comiencen con el signo '#'.
    foreach $line (@buffer){
       $line =~ s/  */ /g;
       next if ($line =~ /^#/ |  $line =~ /^$/ || $line !~ /=/);
       ($key, $value) = split(/=/,$line);

      # Elimina los caracteres que son espacios en blanco del nombre y
      # valor de la configuración
       $key =~ s/\s//g;
       $value =~ s/\s//g;

      # Guarda el par clave=valor en la array (hash) asociada
      # %conf.
       $conf{$key} = $value;
       }
    }

sub readFile{
    # Propósito: lee un archivo y devuelve líneas.

    # Toma el nombre del archivo de una lista de parámetros
    my $file = shift;

    # Variables locales
    my @buffer;
    my $line;

    # Si el archivo existe, se abre y se leen las líneas que se
    # encuentran en un buffer. Pero elimina los nuevos caracteres de
    # cada una de ellas antes de almacenarlas en el buffer. Después
    # de leer el archivo, se cierra.
    if(-e  $file){
       open(FP,$file) || die "Can not read $file \n";
       while($line = <FP>){
         chomp($line);
         push(@buffer,$line);
         }
       close(FP);
       }

    # Devuelve el buffer.
    return (@buffer);
    }

sub showErrors{

    # Propósito: esta subrutina muestra un mensaje de error en una
    # página. Toma el error de una lista de parámetros y lo guarda en
    # una lista local.
    my @fields = @_;
```

```
    # Variabas locales
    my $err_template;
    my @html;
    my $line;
    my $errorlist;

    # Toma el FQPN del archivo en el que se encuentra la plantilla de
    # errores.
    $err_template = sprintf("%s/%s/
%s",$conf{'base_dir'},$conf{'template_dir'},
$conf{'input_error_template'});

    # Crea una lista HTML para mostrar los campos en los que se ha
    # producido un error.
    $errorlist = "<UL>";
    foreach (@fields){
       $errorlist .= "<LI> $_ ";
       }
    $errorlist .= "</UL>";

    # Lee el archivo en el que se encuentra la plantilla de error y
    # guarda la página en el buffer local.
    @html = &readFile($err_template);

    # Recorre el texto de la página y sustituye la variable
    # <ERROR-LIST> con la lista de errores que acaba de crear.
    # Imprime la página.
    foreach $line (@html){
      $line  ~ s/<ERROR-LIST>/$errorlist/g;
      print $line;
      }
  }
```

Listado 8.3. wizform.cgi

La mejor forma de comprender este script es analizando una de sus aplicaciones. En la siguiente sección muestro cómo crear una aplicación ejemplo, llamada On line Newletter Singup Tool, que se utiliza para procesar formularios de carácter general.

Una aplicación ejemplo

El propósito de esta aplicación es permitir que los visitantes de un sitio Web se registren para recibir un boletín que distribuye periódicamente dicho sistema. La herramienta de registro tiene que recopilar los nombres de los visitantes y la dirección de correo electrónico y los guarda en un archivo que activa el propietario del sistema para que envíe los boletines basándose en los datos recopilados.

En las próximas secciones le mostraré cómo crear el archivo de configuración y las plantillas HTML que necesita esta aplicación. Para este ejemplo, asumiremos lo siguiente:

```
DocumentRoot - /www/nitec/public/htdocs
```

El primer paso que hay que dar es crear un archivo de configuración para la herramienta de registro. Utilice un editor de textos para crear un archivo llamado singup.conf. Escriba las siguientes líneas de configuración:

```
base_dir=/www/nitec/public
template_dir = htdocs
```

La línea base_dir le indica al script el nombre del directorio que utilizará como base. Como este directorio tiene que salir un nivel del directorio DocumentRoot (/www/nitec/public/htdocs), se configura para que sea /www/nitec/public. La línea template_dir especifica el nombre del directorio en el que se guardan las plantillas. En este ejemplo, se utiliza el directorio del nivel más alto (DocumentRoot). Observe que el valor de template_dir ha de ser relativo al path utilizado en base_dir.

Tiene que especificar la información de la configuración de la entrada en el mismo archivo:

```
input_fields=firstName,lastName,emailAddress,companyName,officePhone
required_fields=emailAddress,firstName,lastName
```

Aquí, firstName, lastName, emailAddress, companyName y officePhone son los nombres de los campos de entrada del formulario HTML de esta aplicación. Observe que los nombres de los campos que aparecen en la lista están separados por comas. La segunda línea le especifica al script cuáles son los campos necesarios (emailAddress, firstName, lastName). En otras palabras, si el usuario no rellena estos campos, el script mostrará un mensaje de error.

Ahora hay que mostrarle al script dónde tiene que escribir los datos que ha ido recopilando de los usuarios. Habrá que añadir las siguientes líneas en el archivo de configuración:

```
data_dir=data
data_file=newsletter.cvs
field_separator=,
saving_order=emailAddress,firstName,lastName,companyName,officePhone
```

La primera línea especifica el nombre del directorio en el que hay que escribir los archivos de datos. El nombre de este directorio está asociado al valor de base_dir para que sea completamente válido como nombre de directorio.

Obsérvese que la línea data_dir determina que el directorio ha de ser /www/nitec/public/data, que se encuentra fuera del especificado en DocumentRoot. Asegúrese de que el servidor Web puede leer y escribir en los archivos que se encuentran en dicho directorio. Conviene mantener alejada esta carpeta fuera del alcance de la Red, por cuestiones de seguridad. La segunda línea se utiliza para especificar el nombre del archivo de datos que se va a escribir. La tercera especifica el separador de campo que se utilizará con dichos datos. En este ejemplo se utiliza la coma (,) como separador. La cuarta línea se utiliza para especificar el orden en el que se han de escribir los datos en el archivo.

Ahora le tendrá que indicar al script qué plantillas HTML desea utilizar. Para ello se utilizan las siguientes líneas:

```
input_error_template = newsletter.err.html
success_template = newsletter.success.html
failure_template   newsletter.failed.html
```

La primera especifica el nombre de la plantilla que se va a usar para mostrar los mensajes personalizados correspondientes a los errores que vayan apareciendo. El path completo es /www/nitec/public/htdocs/newsletter.err.html (es decir, base_dir/template_dir/input_error_template). El archivo newsletter.html, que aparece en la segunda línea, se muestra cuando el script completa una tarea con éxito. Se utiliza para mantener informado al usuario.

El path completo para este archivo será /www/nitec/public/htdocs/newsletter.success.html (es decir, base_dir/template_dir/success_template).

El archivo newsletter.failed.html que aparece en la tercera línea se mostrará cuando ocurra un error interno (como un fallo en el envío de correo o un fallo de escritura debido a un error con los permisos).

El path completo para este archivo es /www/nitec/public/htdocs/newsletter.failed.html (es decir, base_dir/template_dir/failure _template).

A continuación, habrá que indicarle al script que escriba la salida en un archivo y que los mande por correo electrónico. Se utilizan las siguientes declaraciones:

```
output_method=file,email
mail_to=newsletter@nitec.com
mail_from=signup@nitec.com
mail_subject=New registration
mail_daemon=/usr/sbin/sendmail
```

La primera línea le indica al script que escriba los datos en un archivo y que los envíe por correo electrónico. En la segunda se especifica la dirección de correo a la que se han de enviar los datos. La tercera, la dirección del remitente que utilizará el programa. La cuarta, el asunto (subject) que aparecerá

en la cabecera. La quinta especifica el path completo del demonio de correo. Si utiliza otro servidor, tendrá que modificar los programas CGI. El script le permite crear un registro para cada una de las operaciones de entrada de datos que efectúa un visitante. Puede especificar el nombre del archivo de registro de la siguiente manera:

```
log_file=newsletter.log
```

El archivo de registro se escribe en el directorio data_dir. Por eso se utiliza base_dir/data_dir/log_file para construir el path completo.

El script también le permite personalizar la plantilla. Usa la definición de una etiqueta para localizar los valores de los datos que hay que sustituir. Estas etiquetas personalizadas comienzan con el prefijo VISITOR-. Por ejemplo:

```
VISITOR-LAST=lastName
```

Especificar estos valores en el archivo de configuración implica localizar las etiquetas personalizadas llamadas <VISITOR-LAST> que se encuentran en la plantilla de mensajes de éxito (success_template) y sustituirlas con el valor introducido en la variable lastName. Obsérvese que los nombres de las variables VISITOR- han de aparecer en mayúsculas tanto en el archivo de configuración como en el mensaje HTML que nos ocupa.

El aspecto final del archivo de configuración (con algunos comentarios, que son los que se encuentran en las líneas que comienzan por el símbolo #) se lo muestro en el listado 8.4.

```
#
# signup.conf
#
# Propósito: este archivo se utiliza como script de proceso de
#            formularios.
#
# Aplicación: Ejemplo de herramienta de registro.
# $Author$
# $Id$
# $Version$
# $Status$
##################################################################


# directorio base y de plantillas
base_dir=/www/nitec/public
template_dir = htdocs

# requisitos de la entrada
input_fields=firstName,lastName,emailAddress,companyName,officePhone
required_fields=emailAddress,firstName,lastName
```

```
# datos
data_dir=data
data_file=newsletter.cvs
field_separator=,
saving_order=emailAddress,firstName,lastName,companyName,officePhone

# plantilla de respuesta
input_error_template = newsletter.err.html
success_template = newsletter.success.html
failure_template = newsletter.failed.html

# salida
output_method=file,email
mail_to=newsletter@ nitec.com
mail_from=signup@nitec.com
mail_subject=New registration
mail_daemon=/usr/sbin/sendmail

# Registro
log_file=newsletter.log

# Variables comunes
VISITOR-LAST=lastName
VISITOR-FIRST=firstName
VISITOR-EMAIL=emailAddress
VISITOR-HOMEADDR1=homeAddress1
VISITOR-HOMEADDR2=homeAddress2
VISITOR-HOMEPHONE=homePhone
VISITOR-OFFICEPHONE=officePhone
```

Listado 8.4. signup.conf

Guarde este archivo en el mismo directorio del script (es decir, en el directorio cgi-bin del servidor). Cree a continuación el formulario que aparece en el listado 8.5.

```
<HTML>
<HEAD>  <TITLE> Formulario Newsletter Sign-up </TITLE> </HEAD>

<BODY BGCOLOR="white">
<TABLE BORDER=1>
<TR BGCOLOR="#abcdef">
<TD>

<FORM ACTION="/cgi-bin/wizform.cgi" METHOD="POST">

<TABLE BORDER=0 CELLPADDING=6 CELLSPACING=0>

<TR> <TD BGCOLOR="#abcdef" ALIGN="LEFT" COLSPAN="2">
     <FONT FACE="Arial Bold"> Formulario Newsletter Sign-up
</FONT><BR>
     </TD>
</TR>
```

```
<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT"> Nombre * </TD>
     <TD BGCOLOR="#aabbcc"> <INPUT TYPE="TEXT" NAME="firstName"
SIZE="20" MAXSIZE="30"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT"> Apellido * </TD>
     <TD BGCOLOR="#aabbcc"> <INPUT TYPE="TEXT" NAME="lastName"
SIZE="20" MAXSIZE="30"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT"> Email * </TD>
     <TD BGCOLOR="#aabbcc"> <INPUT TYPE="TEXT" NAME="emailAddress"
SIZE="20" MAXSIZE="30"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT"> Empresa </TD>
     <TD BGCOLOR="#aabbcc"> <INPUT TYPE="TEXT" NAME="companyName"
SIZE="20" MAXSIZE="20"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT"> Teléfono </TD>
<TD BGCOLOR="#aabbcc"> <INPUT TYPE="TEXT" NAME="officePhone"
SIZE="20" MAXSIZE="15"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="RIGHT">   </TD>
     <TD BGCOLOR="#aabbcc"> <INPUT TYPE="RESET" VALUE="Borrar
todo ">     <INPUT TYPE="SUBMIT" VALUE="Enviar"> </TD>
</TR>

<TR> <TD BGCOLOR="#abcdef" ALIGN="LEFT" COLSPAN="2">
      <FONT SIZE="-1"> * Campo obligatorio.</FONT>
      </TD>
</TR>

</TABLE>

<INPUT TYPE=HIDDEN NAME="conf_file" VALUE="newsletter.conf">

</FORM>

</TD>
<TD BGCOLOR="#abcdef" VALIGN="TOP">
<FONT SIZE="+2">E</FONT>nviaremos un boletín mensual a todos
nuestros suscriptores. Si quiere recibirlo, regístrese aquí,
por favor.
</TD>
</TR>

</TABLE>
</BODY>
</HTML>
```

Listado 8.5. newsletter.html

Hay varias cosas del formulario que no merecen la pena. La primera a tener en cuenta es:

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

Aquí, se envía el ACTION del formulario al script wixform que se encuentra en el directorio cgi-bin. Se utiliza el método POST. Obsérvese que todos los datos de los campos de entrada tienen el mismo nombre (algo así como lastName o firstName) tal y como se especifica en el archivo de configuración. Este es uno de los requisitos del script. Si los nombres de los campos no fuesen los mismos, el script no funcionaría correctamente.

Otro punto a destacar es que hay un campo oculto:

```
<INPUT TYPE=HIDDEN NAME="conf_file" VALUE="newsletter.conf">
```

Especifica el nombre del archivo de configuración.

Después de crear este formulario, guárdelo en el directorio htdocs. Los dos listados que le muestro a continuación muestran una página HTML que se puede crear para personalizar los mensajes de error de entrada, la página de información (es decir, en la que se le comunica al usuario que el envío se ha efectuado sin ningún contratiempo) una página para los fallos internos. Estas son las páginas HTML más sencillas. La página de newsletter.success.html (listado 8.6) tiene unas cuantas etiquetas VISITOR para personalizar un poco el aspecto de la página.

```
<HTML>
<HEAD>
<TITLE> Formulario On-line Newsletter Sign-up (Éxito) </TITLE>
</HEAD>

<BODY BGCOLOR="#abcdef">
<BLOCKQUOTE>

<HR>
<FONT FACE="Arial Bold">
Hello <VISITOR-FIRST> <VISITOR-LAST><BR><BR>
</FONT>

Gracias por registrarse. Le enviaremos la última copia de
nuestro boletín a la dirección de correo electrónico
<STRONG><VISITOR-EMAIL></STRONG>) que especifique aquí.
Que tenga un buen día.
<BR>

<FORM ACTION="/" METHOD="GET"> <INPUT TYPE=SUBMIT VALUE="Volver a
índice "> </FORM>
<HR>
```

```
</BLOCKQUOTE>
</BODY>
</HTML>
```

Listado 8.6. newsletter.success.html

El archivo newsletter.err.html es el que se muestra a los visitantes que se olvidan de rellenar todos los campos. La etiqueta personalizada <ERROR-LIST> se sustituye por el nombre de todos los campos que se han perdido en el envío de un visitante.

```
<HTML>
<HEAD>
<TITLE> Formulario On-line Newsletter Sign-up (Error) </TITLE>

</HEAD>

<BODY BGCOLOR="#abodcf">
<BLOCKQUOTE>
<FONT FACE="Arial Bold" SIZE="+1">Solicitud de suscripción
incompleta </FONT><BR>
<HR>

No se ha podido procesar su registro, ya que se ha perdido parte
de la información.
<BR>
<ERROR-LIST>
<BR>
Por favor, inténtelo de nuevo.
<BR>

<FORM ACTION="/newsletter.html" METHOD="GET"> <INPUT TYPE SUBMIT
VALUE="Volver a intentarlo"> </FORM>
</BLOCKQUOTE>
</BODY>
</HTML>
```

Listado 8.7. newsletter.err.html

El archivo newsletter.failure.html se mostrará cuando el script detecte un problema en el acceso a los archivos, como puede ser un fallo de escritura debido a una mala configuración de los permisos.

```
<HTML>
<HEAD>
<TITLE> Formulario On-line Newsletter Sign up (Error Interno)
</TITLE>

</HEAD>
```

```
<BODY BGCOLOR="#abcdef">
<BLOCKQUOTE>
<FONT FACE="Arial Bold" SIZE="+1">Error Interno de Suscripción
</FONT><BR>
<HR>
Lo siento, pero no hemos podido procesar su registro.
<BR>
<BR>

<FORM ACTION="/ newsletter.html" METHOD="GET"> <INPUT TYPE=SUBMIT
VALUE="Volver a intentarlo "> </FORM>
</BLOCKQUOTE>
</BODY>
</HTML>
```

Listado 8.8. newsletter.failure.html

Guarde estas páginas (que no son otra cosa que plantillas) dentro de template_dir, variable especificada en el archivo de configuración. Asegúrese de que el script CGI.pm se encuentra en el directorio cgi-bin y que cuenta con los permisos de Apache de lectura y escritura.

Comprobar la herramienta Newsletter sign-up

Ahora vamos a comprobar cómo funciona nuestra creación. En la figura 8.5 se puede ver el aspecto que presentará la página newsletter.html al visualizarla a través de un explorador Web. Una vez que el visitante ha completado todos los campos del formulario, se ejecuta el script. En primer lugar, conviene comprobar si los campos de entrada que necesita el archivo de configuración han sido correctamente cumplimentados.

Si el usuario no ha cumplimentado los campos requeridos, se encontrará con el mensaje de error de la figura 8.6.

En caso contrario se le mostrará la página de la figura 8.7.

Después de enviar un par de solicitudes, el aspecto del archivo de datos newsletter.cvs tiene el siguiente aspecto:

```
kabir@nitec.com,Mohammed,Kabir,Nitec,555-1111
joe@gunchy.org,Joe,Gunchy,Gunchy Inc,555-2222
alguien@algunlugar.net,Nombre,Apellido,Empresa,555-3333
```

Y el archivo newsletter.log:

```
11:26:00 4/1/98, apache.nitec.com, 206.171.50.51, Mozilla/4.04 [en]
(WinNT; I)
11:27:00 4/1/98, apache.nitec.com, 206.171.50.51, Mozilla/4.04 [en]
(WinNT; I)
11:30:35 4/13/98, apache.nitec.com, 206.171.50.51, Mozilla/4.0
(compatible; MSIE 4.01; Windows NT)
```

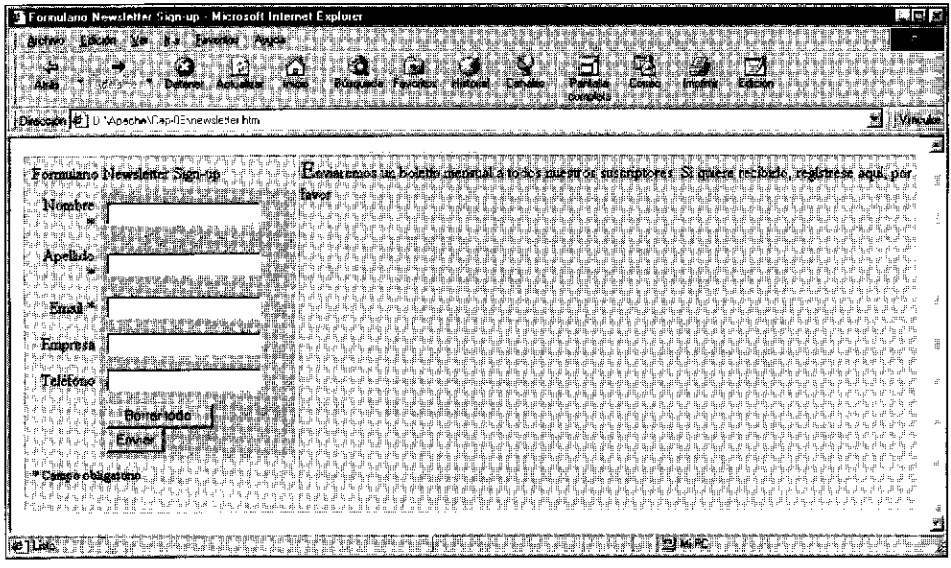


Figura 8.5. Página newspetter.html

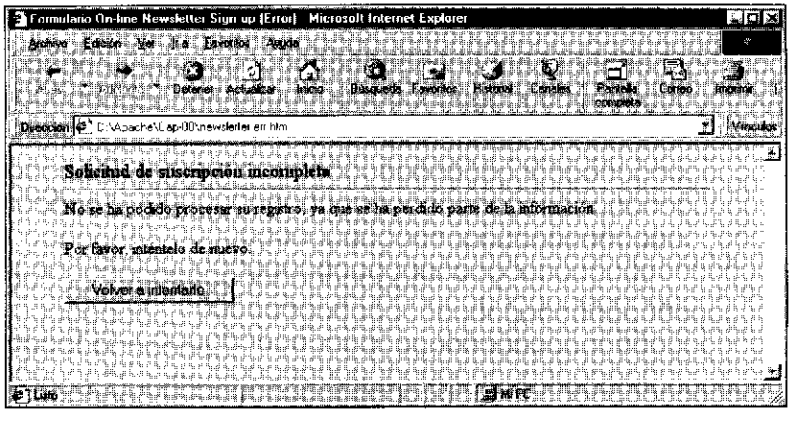


Figura 8.6. Error de entrada

Como puede observar, todo funciona correctamente. Si hubiese algún tipo de problema con los permisos a la hora de escribir los datos, aparecería el mensaje de error de la figura 8.8.

Modificar la herramienta newsletter sign-up

Es posible que ahora se esté preguntando si puede utilizar este script con una aplicación multiplataforma con la que se le mostrará a los visitantes más de un formulario HTML. La respuesta es clara: sí que puede hacerse.

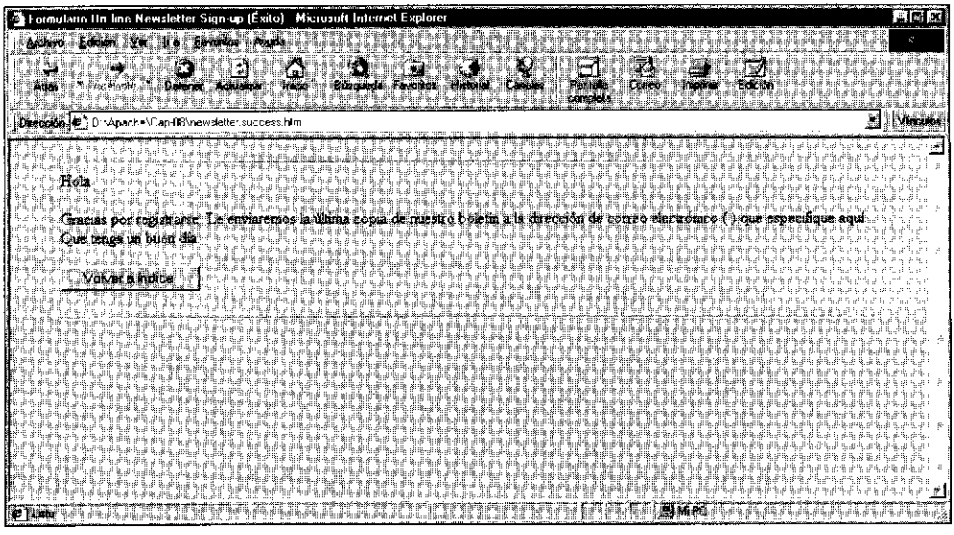


Figura 8.7. Página de información

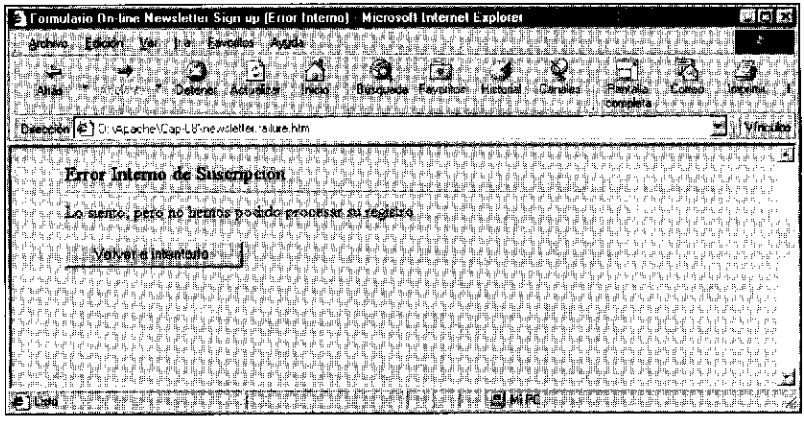


Figura 8.8. Mensaje de error

Todo lo que tiene que hacer es añadir una nueva variable oculta en todos los formatos HTML excepto en el último. Su aspecto es el que se muestra a continuación:

```
<INPUT TYPE=HIDDEN NAME="next form" VALUE="nextform.html">
```

Aquí, nextform.html es el formulario HTML que va detrás de aquél con el que se acaba de trabajar. Por ejemplo, si tengo tres formularios para una misma aplicación, tendré que hacer lo siguiente.

En el primer formulario HTML:

```
<INPUT TYPE=HIDDEN NAME="next_form" VALUE="2ndform.html">
```

en el segundo:

```
<INPUT TYPE=HIDDEN NAME="next_form" VALUE="lastform.html">
```

El último formulario no tendrá dicha variable oculta. Obsérvese que todos los formularios se han de guardar en el directorio que se especifique a través de la variable template_dir. Por medio de este generador de formularios se pueden crear las aplicaciones más utilizadas en la Red, por ejemplo, un libro de firmas para visitantes o una lista de correo electrónico.

Los módulos CGI para Perl más utilizados

En el script anterior especializado en la generación de formularios he usado uno de los paquetes más comunes, CGI.pm, creado por Lincoln D. Stein. Se pueden usar otros paquetes en Perl con los que desarrollar script CGI. Los más famosos son los siguientes:

- CGI.pm. Un gran paquete en Perl que desarrolla script CGI de forma rápida y limpia. Le permite crear objetos CGI gracias a los cuales podrá acceder a las cadenas de peticiones CGI, crear cabeceras, leer y escribir cookies y escribir formularios HTML sin necesidad de utilizar código HTML en su script. Puede localizar este paquete en:

  ```
  www-genome.wi.mit.edu/ftp/pub/software/WWW/cgi_docs.html
  ```

- Libwww. Si tiene interés por desarrollar potentes agentes, robots o pequeños servidores Web, no se aburra escribiendo grandes cantidades de código. Utilice los ocho módulos de Libwww: HTML, LWP, MIME, WWW, HTTP, URL File y Font. Los encontrará en:

  ```
  www.perl.com/CPAN-local/modules/by-module/LWP/
  ```

- cgi-bin.lib. Si aún no está preparado para la programación en un lenguaje orientado a objetos como Perl (los dos módulos anteriores están orientados a objetos), puede probar una librería llamada cgi-bin.pl. Su rutinas están especializadas en el análisis de las entradas de formularios y en la creación de cabeceras. Lo encontrará en:

  ```
  www.bio.cam.ac.uk/cgi-lib/
  ```

Depurar programas CGI en Apache

Cuando escribe o instala un programa CGI escrito por otra persona es muy probable que no funcione exactamente como estaba previsto. Analizar un programa para buscar errores es una tarea muy complicada, porque en la mayoría de los casos no se puede ejecutar el programa desde la línea de comandos y suministrarle datos para que los procese.

Truco: si trabaja con el paquete CGI.pm con sus script en Perl, si que puede utilizar la línea de comandos para depurar sus programas CGI. Este paquete le permite trabajar en modo interactivo, suministrando pares de valores (clave=valor) como si los datos provinieses de un formulario HTML.

Para ayudar a los desarrolladores CGI, Apache dispone de una serie de registros para las salidas CGI. Para cada error que produzca uno de estos programas, se graba una serie de líneas explicativas en un archivo de registro. En las dos primeras aparece la hora en la que se formuló la petición, el URI solicitado, estado HTTP, nombre del programa CGI, etc. Si dicho programa no se puede llegar a ejecutar, aparecerán dos líneas en las que se almacena información sobre el error. Por otro lado, si el error viene provocado por una información incorrecta en el contenido de una cabecera, el registro será como sigue: todas las cabeceras de las peticiones HTTP, todas las cabeceras generadas por el programa CGI y STDIN y STDOUT del programa. Si el script fallase a la hora de enviar datos a STDOUT, no se registraría en el informe.

Para registrar una salida CGI en Apache se utilizan las siguientes directrices, que se encuentran en el módulo mod_cgi (forma parte de la distribución estándar).

ScriptLog

```
Sintaxis: ScriptLog nombre_archivo
Predeterminado: nada
Contexto: configuración del recurso
```

Esta directriz determina el nombre del archivo de registro en el que se grabarán los errores del programa CGI. Si dicho nombre es relativo (es decir, no comienza por /), se referirá automáticamente al directorio raíz del servidor especificado en la directriz ServerRoot.

Cuando utilice esta directriz, asegúrese de que el usuario declarado en la directriz UserDir tiene permiso para escribir en el directorio de registro. Tenga en cuenta que no es conveniente utilizar esta directriz para el trabajo diario ya que se perdería bastante eficacia. Mi consejo es que cuando se tenga que usar se active y se devuelva a su estado inicial cuando se complete el proceso de depuración.

ScriptLogLength

```
Sintaxis: ScriptLogLength tamaño
Predeterminado: 10385760
Contexto: configuración del recurso
```

Esta directriz limita el tamaño del archivo de registro especificado en ScriptLog. En dicho fichero se puede guardar gran cantidad de información por cada error CGI que tenga lugar, por lo que puede crecer rápidamente. Al utilizar ScriptLogLength se limita el tamaño del archivo, por lo que alcanzada dicha cantidad de bytes se dejará de almacenar información.

ScriptLogBuffer

```
Sintaxis: ScriptLogBuffer tamaño
Predeterminado: 1024
Contexto: configuración del recurso
```

Esta directriz limita el tamaño de los datos POST y PUT que se guardan.

Con ella se puede configurar el registro de programas CGI que desarrolla o que trata de instalar en el sistema.

9 FastCGI

Desde hace unos años, CGI se ha convertido en el factor estándar del desarrollo de aplicaciones para servidores Web. Se ha podido comprobar que los sistemas Web que tienen que trabajar con grandes CGI reducen considerablemente su rendimiento. Las investigaciones consiguieron determinar cuál era el cuello de botella de estas aplicaciones: cada vez que un cliente solicitaba una aplicación CGI, el servidor Web iniciaba un nuevo proceso. Dicho proceso efectuaba su trabajo y luego se extinguía. La verdad es que este método funciona muy bien cuando la carga de trabajo es pequeña. Pero con una carga importante de trabajo, la configuración e iniciación del proceso reducían considerablemente la eficacia del sistema.

La situación está cambiando gracias a la aparición de nuevos estándares. Aunque aún no ha sido plenamente aceptado por todo el mundo, ha aparecido un nuevo estándar que promete cambiar radicalmente esta situación: FastCGI. En este capítulo lo veremos y aprenderemos a implementarlo en Apache.

FastCGI como una nueva alternativa

Una aplicación FastCGI se comporta exactamente igual que una propia del servidor. A diferencia de las CGI, permanece activa y se hace cargo de todas

las aplicaciones que recibe el servidor Web. Esta es su principal ventaja. Al contrario que con CGI, el sistema no se satura por tener que iniciar nuevos procesos ni por tener que iniciar la aplicación (por ejemplo, al estar obligado a establecer una conexión con una base de datos) cada vez que se tiene que hacer cargo de una petición. Esta alternativa es más eficaz que CGI y permite escribir aplicaciones para servidores Web en varios lenguajes, entre los cuales tenemos Perl, C, C++, Java y Phyton.

Hay que resaltar que, de momento, FastCGI es una propuesta de un estándar abierto, pero que es muy posible que se acepte como norma para la Red en un futuro cercano. De todas formas, son varios los desarrolladores de aplicaciones que no acaban de comulgar con FastCGI. Son empresas que han desarrollado sus propios estándares, cuya funcionalidad es parecida a la de FastCGI y, en ciertos casos, sus prestaciones son incluso superiores. Estos estándares se pueden clasificar como interfaces para la programación de aplicaciones para servidores (API). Por ejemplo, los desarrolladores más importantes (Netscape y Microsoft) tienen sus propios API: NSAPI e ISAPI respectivamente. Incluso el propio Apache tiene su propio API.

Nota: si desea recibir la última información sobre FastCGI, suscríbase a la lista de correo electrónico que se encuentra en fastcgi-developers@openmarket.com. Como esta lista la utilizan desarrolladores de FastCGI, es posible que no sea el sitio adecuado para hacer preguntas de nivel básico. Para unirse a la lista de mail, envíe un mensaje a dicha dirección y escriba en Asunto, *subscribe*. Si quiere consultar los mensajes publicados, puede hacerlo en

www.fastcgi.com/mail/

La existencia de CGI, FastCGI y Server API crea cierta confusión entre los desarrolladores y administradores Web. En la tabla 9.1 tiene el resultado de comparar ciertos aspectos de estas tecnologías.

Tabla 9.1. Comparación de CGI, Server API y FastCGI

Propiedad	CGI	Server API	FastCGI
Dependencia del lenguaje de programación	Independiente del lenguaje de programación. Casi se puede	Hay que escribir las aplicaciones en un lenguaje con el que pueda	Independiente del lenguaje de programación. Casi se puede utilizar cual-

Propiedad	CGI	Server API	FastCGI
	utilizar cualquier lenguaje de programación para escribir aplicaciones CGI.	trabajar el desarrollador del API (generalmente C/C++).	lenguaje de programación para escribir aplicaciones CGI.
Aislamiento del proceso	Soportado. Las aplicaciones se ejecutan en procesos separados. Así, las aplicaciones con errores no podrán colgar al servidor Web o acceder a zonas privadas.	No soportado. Como las aplicaciones se ejecutan en el espacio asignado por el servidor, las que tienen errores pueden corromper el servidor y comprometer la seguridad. Los errores que aparezcan en el corazón del servidor podrán corromper las aplicaciones.	Soportado. Una aplicación FastCGI con errores no podrá corromper ni el corazón del servidor ni otras aplicaciones.
Tipo de estándar	Estándar abierto. Algunos formularios CGI han llegado a implementarse en todos los servidores Web.	Propiedad. Codifique su aplicación para un API en particular y utilícelo con un único servidor Web.	Sin propiedad, propuesto estándar abierto. Se está desarrollando su soporte para servidores Web, incluidos los servidores comerciales de Microsoft y Netscape. En la actualidad Apache puede trabajar con FastCGI como un módulo externo.
Dependencia de la plataforma	Independiente de la plataforma. CGI no está sujeto a ningu-	Sujeto a la arquitectura propia del servidor. Las aplicaciones API tie-	Independiente de la plataforma. No está sujeto a ninguna arquitectura

Propiedad	CGI	Server API	FastCGI
	na arquitectura en particular (multitarea, monotarea, etc.)	nen que compartir la misma arquitectura que el servidor. Si éste es multitarea, la aplicación tendrá que ser segura. Si el servidor no es multitarea, no podrá aprovechar las ventajas de una aplicación multitarea.	en particular. Cualquier servidor Web puede implementar una interfaz FastCGI.
Eficacia	Se crea un nuevo proceso para cada solicitud que se extingue en el momento en que se completa la petición. De esta forma no se obtiene demasiada eficacia.	Las aplicaciones se ejecutan en el proceso del servidor y son permanentes, es decir, sobreviven a todas las peticiones. No hay ningún problema relacionado con la configuración/inicio.	Los procesos FastCGI son persistentes. No se pueden hacer cargo de varias solicitudes. No hay ningún problema relacionado con la configuración/inicio.
Complejidad	Fácil de comprender.	Muy complicado. Los desarrolladores de API trataron de aumentar la implementación y conservar los costes.	Simple, con una sencilla migración de CGI.
Arquitectura distribuida	No la soporta. Para ejecutar aplicaciones CGI en un sistema remoto, dicho sistema tendrá que contar con un servidor Web y con aplicaciones CGI.	Depende del fabricante.	Soportado. Las aplicaciones FastCGI pueden trabajar con cualquier host TCP/IP.

Ahora que conoce las distintas opciones que tiene a su disposición, si desea darle una oportunidad a FastCGI, se ha dirigido al sitio correcto. Vamos a ver los beneficios que tiene el uso de FastCGI.

Ventajas de FastCGI

A continuación veremos las ventajas que tiene el uso de FastCGI sobre CGI y sobre un servidor API.

Gran eficacia gracias a la memoria caché

¿Cómo es de rápido FastCGI? La respuesta depende de la aplicación. Si ésta lee los datos que se encuentran en un archivo y los mete en memoria caché, nos encontraremos con que la versión que se haya creado utilizando FastCGI será mucho más rápida que la creada con CGI o con un servidor API. Una aplicación CGI por especificación no puede utilizar la memoria caché porque se abre una nueva aplicación por cada petición recibida que se extingue en el momento en que se completa el proceso. Del mismo modo, la mayoría de las aplicaciones basadas en la interfaz API de los servidores Web ejecutan subprocesos que no comparten memoria, por lo que tampoco se pueden almacenar en la memoria caché. Aún en el supuesto de que se implemente el almacenamiento en memoria caché de este proceso, el resultado obtenido será bastante pobre porque se tiene que hacer una copia de todos los subprocesos en la memoria caché, con lo que la pérdida de memoria es considerable.

FastCGI se ha diseñado para que pueda trabajar con la memoria caché. Las peticiones pasan del subproceso al servidor encargado de las aplicaciones FastCGI. El proceso de esta aplicación es el encargado de mantener esta memoria. Obsérvese que en algunos casos no bastará con un único servidor FastCGI para que se haga cargo de las peticiones recibidas. Con un sistema de multitarea puede ejecutar una aplicación que se encargue de controlar todas las peticiones al mismo tiempo. Los thread de control comparten la memoria destinada a los procesos, por lo que tendrán acceso a la misma caché.

El desarrollador de FastCGI efectuó una serie de pruebas, para las que utilizó tres versiones distintas de una aplicación (basadas es CGI, FastCGI y en una especificación API) que interactuaba con un servidor de una base de datos. Lo que se encontraron fue que cuando la versión FastCGI utilizaba la memoria caché y establecía comunicación con el servidor de la base de datos, la eficacia alcanzada era claramente superior a la conseguida por las versiones CGI y API.

Cuando se desactivaba el uso de la memoria caché y se utilizaba una conexión persistente con la versión API, la eficacia conseguida era ligeramente mayor que la obtenida por la versión FastCGI en las mismas condiciones. Con esto se tiene que únicamente cuando se trabaja a nivel de campos (es decir, cuando se desactivan propiedades que puede aprovechar FastCGI, como es el uso de la memoria caché), la versión API es la ganadora de la prueba. Pero, ¿por qué desactivar el uso de la caché? En otras palabras, mientras no escriba un script en FastCGI que falle, el rendimiento alcanzado será muy superior al de las versiones CGI y API.

La prueba demuestra que la eficacia obtenida con las aplicaciones FastCGI era tres veces superior que las API. Este factor se acrecienta considerablemente cuando la aplicación se tiene que poner en contacto con un servidor de una base de datos. De todas formas, con un servidor que pueda trabajar con multitarea y se haga cargo de la memoria caché y de las conexiones persistentes de los thread API. Esto se debe a la ausencia de comunicación entre procesos que se encuentran por encima del entorno. De todas formas, conviene resaltar que para desarrollar aplicaciones multitarea es necesario efectuar un diseño y una programación muy cuidadosa, ya que un solo thread defectuoso puede provocar la caída del servidor Web.

Por otro lado, los procesos FastCGI aprovechan el modelo basado en procesos independientes a la hora de ejecutar procesos externos. De esta forma se consigue un sistema de servidores Web instalados en red. En caso de que aparezca una aplicación FastCGI defectuosa, el servidor Web seguirá funcionando. Así que si adora trabajar con multitarea y no puede vivir sin ella, siempre podrá escribir sus aplicaciones FastCGI con este modelo, que aprovecha las ventajas del sistema.

Escalabilidad a través de aplicaciones distribuidas

A diferencia de las aplicaciones CGI, las FastCGI no toman las variables de entorno CGI de la tabla de procesos. En vez de eso crea una conexión de doble sentido entre la aplicación y el servidor Web que se utiliza para comunicar con la información del entorno, entradas estándar, salida y errores. De esta forma se permite que todas las aplicaciones FastCGI puedan ejecutarse en servidores remotos utilizando conexiones TCP/IP con el servidor Web, tal y como nos muestra la figura 9.1.

Cuando la eficacia de las aplicaciones CGI y API termina por convertirse en un cuello de botella por la carga de trabajo, la solución suele ser hacerse

con un servidor Web más potente o repartir esos procesos entre varios servidores. Con FastCGI hay una tercera solución a este problema. Las aplicaciones FastCGI se pueden ejecutar en servidores dedicados que estén conectados a una red de trabajo, liberando de esta forma al servidor Web para que haga lo que se le da mejor: atender peticiones Web. Los servidores Web se pueden optimizar para que presten el mejor servicio a la vez que se organiza el servidor de aplicaciones FastCGI para que se haga cargo de ejecutar las aplicaciones con la mayor eficacia. El administrador Web nunca tendrá que preocuparse por equilibrar las necesidades de los recursos del servidor Web y de las aplicaciones que se encuentren en la misma máquina. De esta forma se consigue una configuración más flexible tanto en el lado del servidor Web como en el de las aplicaciones FastCGI.

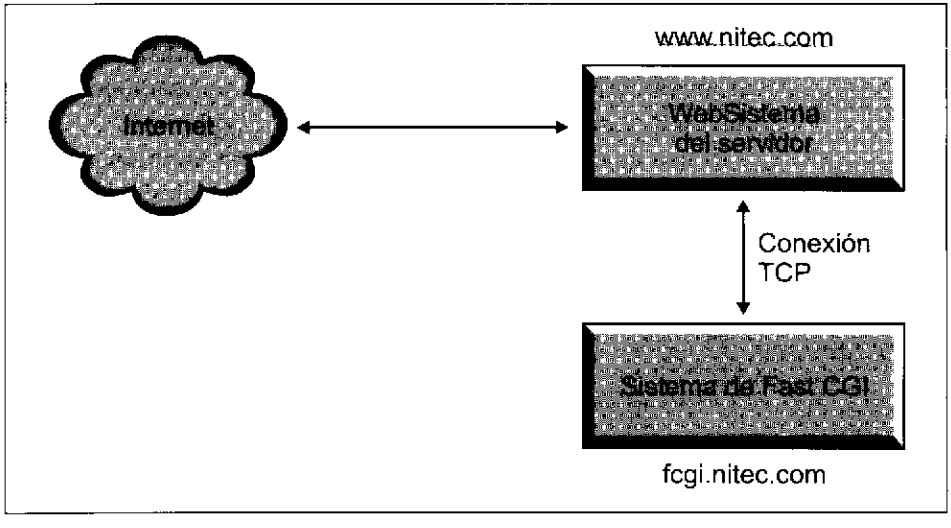


Figura 9.1. FastCGI con una máquina remota

Muchas organizaciones quieren disponer de acceso a bases de datos a través de sus sitios Web. Pero debido a las limitaciones de CGI y de los desarrolladores de API, hay que contestar a un número limitado de versiones de la base de datos en dicho servidor para que puedan proveer este servicio. Este sistema supone una carga considerable de trabajo para el administrador. Con un FastCGI remoto, la aplicación se puede ejecutar dentro de la red interna, simplificando enormemente el trabajo del administrador. Cuando se usa con una configuración de firewall apropiada y se permite el uso de registros, este sistema proporciona el método más seguro, de gran efectividad y escalabilidad, a través del cual se pueden incluir aplicaciones internas y datos en Internet.

Nota: las conexiones de FastCGI remotas tienen dos problemas con respecto a la seguridad: la autenticación y la privacidad. Las aplicaciones FastCGI únicamente deberían aceptar conexiones procedentes de los servidores Web en los que confían (la librería de la aplicación incluye soporte para la validación de direcciones IP). En las futuras versiones del protocolo se incluirá un soporte para las aplicaciones que autentican servidores Web y otro que permitirá establecer conexiones remotas sobre los protocolos seguros, como SSL.

Explicación de FastCGI

Las aplicaciones FastCGI usan una conexión independiente para ponerse en comunicación con un servidor Web. Esta conexión se utiliza para entregar variables de entorno y datos STDIN a las aplicaciones y datos STDOUT y STDERR al servidor Web.

El uso de este protocolo de comunicación también permite que las aplicaciones FastCGI residan en otra máquina (o máquinas) distintas del servidor Web. De esta forma, las aplicaciones pueden distribuirse por más de un sistema, con lo que su integración será más sencilla. Para trabajar con las aplicaciones remotas el servidor utiliza una conexión TCP/IP.

El protocolo FastCGI que se usa en la comunicación entre el servidor Web y las aplicaciones utiliza otro formato diferente. La mayoría de los desarrolladores de aplicaciones ejecutarán sin más la aplicación FastCGI y no tendrán que preocuparse sobre los detalles del protocolo. De todas formas, las aplicaciones especializadas pueden implementar directamente este protocolo.

Como CGI se parece mucho a FastCGI, vamos a revisar la secuencia de peticiones. En la figura 9.2 tiene el esquema del proceso.

Para cada petición CGI, el servidor Web crea e inicia un nuevo proceso CGI. Utiliza las variables de entorno para suministrar al programa toda la información que necesite. Dependiendo del método de petición utilizado (GET o POST), los datos del usuario se guardarán en una variable de entorno llamada QUERY_STRING o se colocarán en la entrada estándar de datos del proceso.

La aplicación CGI es la que se encarga de hacerlo y de enviar todo lo generado a la salida estándar, para que lo lea y analice el servidor (con la excepción de las aplicaciones que generan cabeceras que especifican que no se ha de analizar el contenido). El programa CGI se cierra y el servidor le devuelve la salida CGI al cliente.

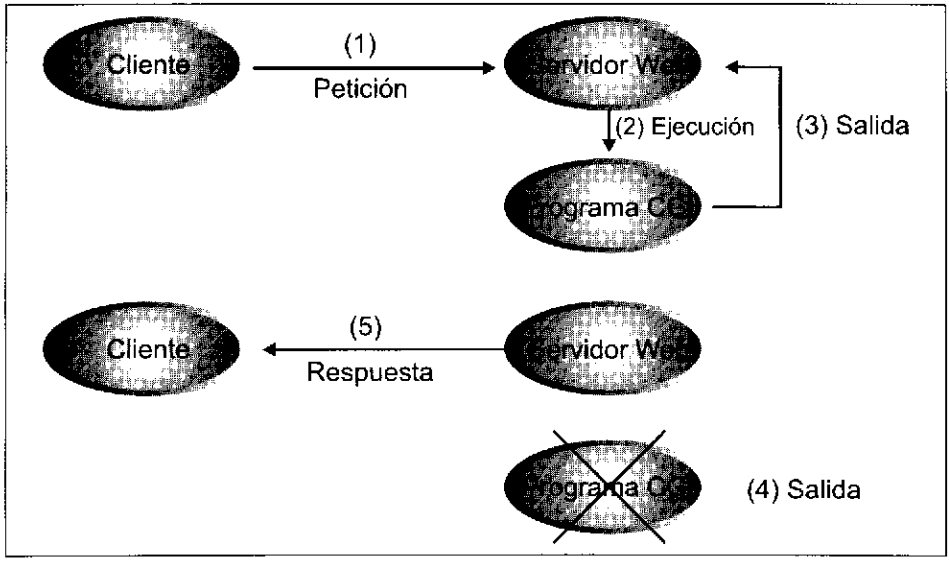


Figura 9.2. Modelo de proceso de una petición CGI

Los procesos FastCGI son persistentes. Después de que se completa una petición, en vez de cerrarse, esperan a que llegue la siguiente, tal y como se puede apreciar en la figura 9.3.

Nota: en el caso de encontrarse con cabeceras de no-análisis, la aplicación CGI será la responsable de producir las cabeceras HTTP apropiadas basándose en el tipo de los datos que se encuentra en STDOUT del programa. Cualquier información sobre los errores escrita en el STDERR del programa CGI será registrado por el servidor Web.

La solicitud del cliente que procesa una aplicación FastCGI tiene lugar como sigue:

1. El servidor Web crea una aplicación FastCGI que analiza el controlador de peticiones. A continuación ya se pueden crear procesos durante el inicio o en el momento en que se necesiten.
2. El programa FastCGI se inicia él solo y espera a que el servidor Web establezca una conexión.
3. Cuando llega un cliente, el servidor Web abre la conexión del proceso FastCGI. Envía las variables de entorno CGI con información y la entrada estándar.

4. Los procesos FastCGI utilizan la misma conexión para enviar de vuelta al servidor Web la información estándar de la salida y de los errores.
5. La petición se habrá completado cuando el proceso FastCGI cierre la conexión. El proceso esperará a que el servidor Web establezca otra conexión.

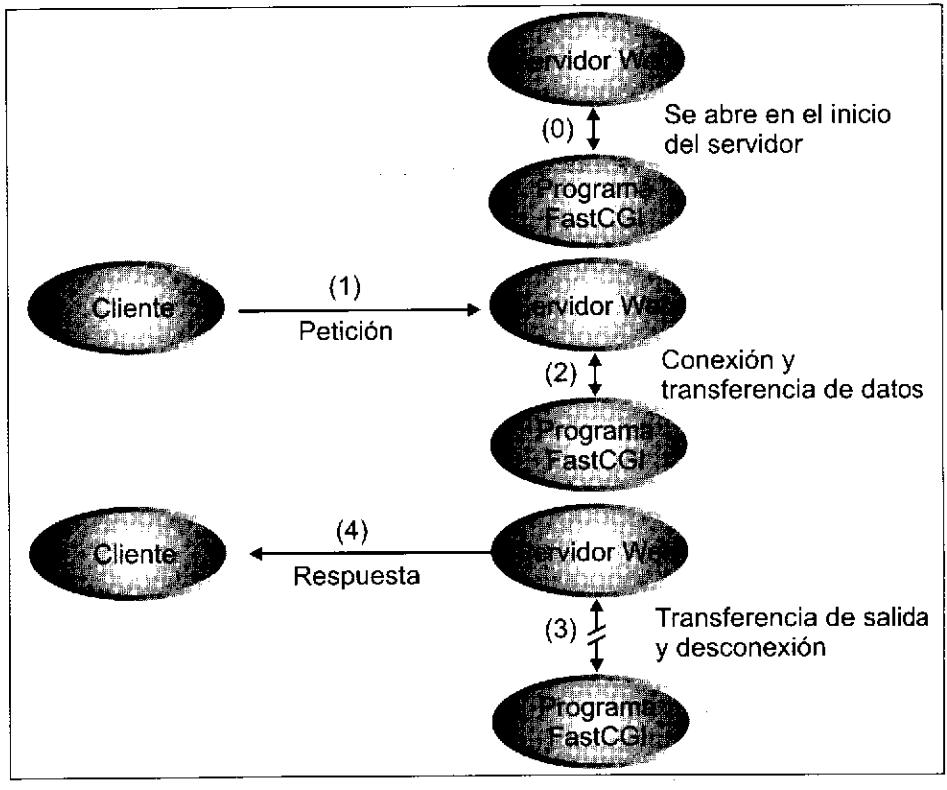


Figura 9.3. Modelo de proceso de una petición FastCGI

Arquitectura básica de una aplicación FastCGI

A diferencia de un programa CGI, uno FastCGI no se extingue después de procesar la petición del cliente. De esta forma se podrá hacer cargo de las futuras peticiones que provengan de un programa CGI. La ejecución de estos programas es secuencial y termina por extinguirse, mientras que uno FastCGI se ejecuta de forma secuencial pero dentro de un bucle infinito. En la figura 9.4 se puede ver la arquitectura básica de una aplicación FastCGI.

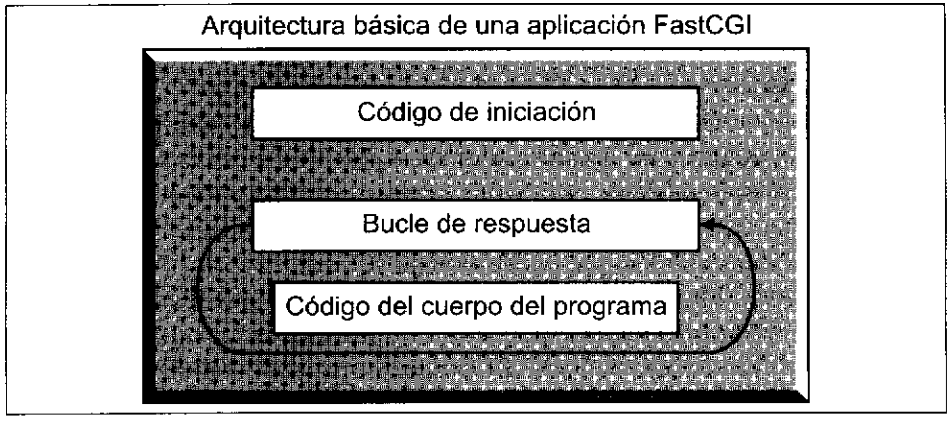


Figura 9.4. Arquitectura básica de una aplicación FastCGI

Como se puede apreciar en la figura, un programa FastCGI tiene un código de iniciación y un bucle dentro del cual se encuentra en cuerpo del programa. El código de iniciación tan sólo se utiliza una vez, cuando hay que abrir la aplicación. Se encarga de efectuar operaciones que consumen tiempo, como abrir bases de datos o calcular los valores de una tabla.

El bucle de respuesta no deja de ejecutarse. Espera la llegada de la petición del cliente. Comienza con una llamada a FCGI_Accept, una rutina que se encuentra en la librería FastCGI y se encarga de bloquear la ejecución del programa hasta que alguna de las peticiones de los clientes solicite la aplicación FastCGI. Cuando llega una de estas peticiones, FCGI_Accept desbloquea, ejecuta una repetición del bucle y se vuelve a bloquear mientras espera la llegada de otra petición. El bucle termina cuando el administrador del sistema o el servidor Web finaliza la aplicación FastCGI.

El cuerpo del programa se ejecuta en cada repetición. Es decir, que para cada petición recibida se ejecuta una vez. Antes de cada repetición FastCGI configurará la información de la petición, como las variables de entorno y los datos de entrada.

Obsérvese que cuando se ejecuta el código del cuerpo de un programa, una llamada a FCGI_Accept informa al servidor de que el programa ha completado el proceso y espera la siguiente petición.

Las aplicaciones FastCGI pueden ser multitarea o no. Para las que no lo son, el servidor Web conserva una pila de procesos FastCGI (si la aplicación se ejecuta localmente) para controlar las peticiones de los clientes. El usuario puede configurar el tamaño de esta pila. Las aplicaciones FastCGI pueden aceptar varias conexiones procedentes de los servidores Web y controlarlas a través de un único proceso.

Tipos de aplicaciones FastCGI

Otro aspecto importante de FastCGI es que puede trabajar con distintas reglas para las aplicaciones. A diferencia de las CGI, una aplicación FastCGI es persistente, por lo que se puede utilizar para otros propósitos, distintos que el proceso de aplicaciones CGI. Vamos a ver los distintos tipos de aplicaciones con los que puede trabajar CGI.

Obviamente, una aplicación FastCGI puede hacer lo mismo que una CGI. Por lo tanto, las aplicaciones típicas serán las mismas que las CGI. Una de las novedades de FastCGI es que ahora puede trabajar con filtros y con aplicaciones externas.

Se pueden crear aplicaciones para filtrar FastCGI que se encarguen de procesar las peticiones antes de enviárselas a los clientes. Por ejemplo, supongamos que se quiere aplicar ciertos estándares a un grupo de cabeceras y pies de páginas HTML (.html) que devuelve un servidor Web. Se puede hacer por medio de una aplicación de filtrado FastCGI. Cuando el servidor recibe una petición para un archivo .html, envía la petición al filtro FastCGI responsable de añadir las cabeceras y los pies. La aplicación FastCGI le entrega los resultados en forma de página HTML al servidor, que se encargará de transmitírselos al cliente.

Las aplicaciones de filtrado FastCGI pueden mejorar significativamente su efectividad si se les permite guardar en memoria caché el resultado de sus filtraciones (el servidor proporciona un tiempo de modificación para que se pueda vaciar la memoria cuando se modifique el archivo del servidor). Las aplicaciones de filtrado pueden resultar de utilidad para desarrollar analizadores para páginas HTML incluidas en las declaraciones SQL, convertidores de formato sobre la marcha, etc.

Otros nuevos tipos de aplicaciones se pueden desarrollar utilizando soporte FastCGI para los programas de autenticación externos y gateways para aplicaciones especializadas en la autenticación. Por ejemplo, si utiliza un servidor de base de datos externo en el que almacene información como el nombre del usuario, contraseñas o cualquier otro dato relacionado con los permisos, puede crear una aplicación FastCGI para que se encargue de establecer una conexión permanente con el servidor de la base de datos y efectuar las consultas necesarias para autenticar el acceso de las particiones. ¿Se podría hacer esto mismo con una aplicación CGI? Sí, con la salvedad que la aplicación CGI tiene que abrir una conexión con el servidor de la base de datos cada vez que se ejecute, lo que podría consumir muchos recursos de sistema (CPU, red, etc.)

Por otro lado, la versión FastCGI de esa misma aplicación conservaría abierta la conexión con el servidor de la base de datos, efectuaría consultas y devolvería el código HTML adecuado basándose en los resultados de las peticiones. Por ejemplo, cuando una petición de acceso va acompañada del par nombre del usuario/contraseña, la aplicación FastCGI le pedirá al servidor en el que se encuentra la base de datos que localice si dicho cliente puede acceder a la fuente solicitada. Si la contestación del servidor es positiva, la aplicación FastCGI devolverá un código de estado HTTP "200 OK". Cuando no se obtiene la autorización, envía un código distinto como "401 Unauthorized".

Ahora que ya está familiarizado con los distintos tipos de aplicaciones FastCGI, vamos a ver cómo se puede pasar de un programa CGI ya existente a FastCGI.

Paso de CGI a FastCGI

Otra de las ventajas de FastCGI es que la conversión de una aplicación CGI a una FastCGI es muy simple. En esta sección veremos cómo convertir una aplicación CGI (escrita en Perl), fontsize.cgi, cuyo código se muestra en el listado 9.1.

```
#!/usr/local/bin/perl

# Variables
my $MAX = 10;
my $i;

# Cabecera de Contenido
print "Content-type text/html\n\n";

# Bucle principal
for ($i=0; $i < $MAX; $i++){
print "<FONT SIZE=$i>Tamaño de la fuente = $i</FONT><BR>";
}
exit 0;
```

Listado 9.1. fontsize.cgi

Esta aplicación CGI produce la salida que se puede apreciar en la figura 9.5.

Vamos a convertirla en una aplicación FastCGI. Para simplificar asumiremos que en el sistema ya se ha instalado de antemano la versión FastCGI de Perl. La línea que le mostramos a continuación le dice al shell que cargue el intérprete de Perl:

```
#!/usr/local/bin/perl
```

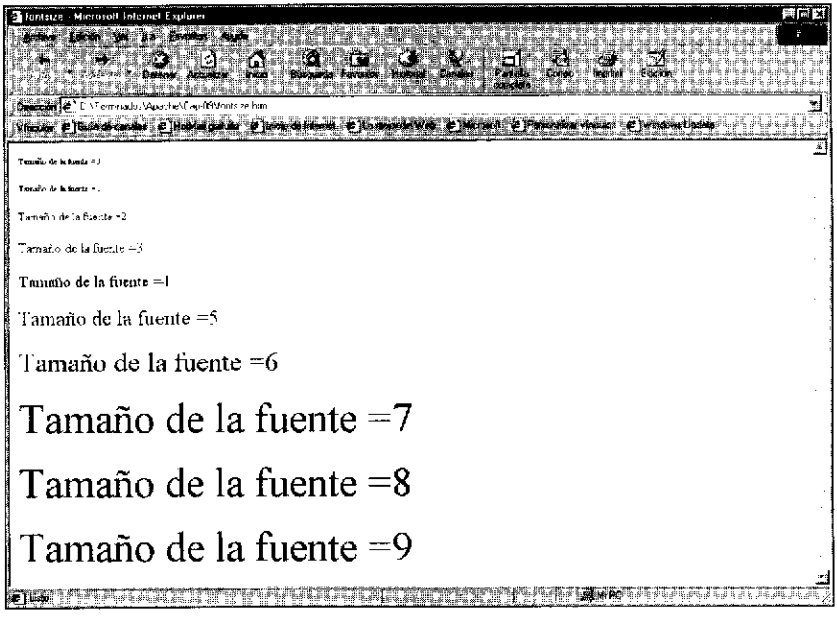


Figura 9.5. Salida generada por la aplicación fontsize.cgi

Si la versión FastCGI del intérprete de Perl tiene otro nombre o se encuentra en otra ubicación, se tendrá que modificar para que su configuración sea la correcta. La configuración que aparece en este ejemplo es la que utilizo en mi sistema.

Para usar el paquete FastCGI de Perl habrá que añadir la siguiente línea:

```
Use FCGI;
```

A continuación, se identifica el bloque de inicio de la aplicación CGI. Para que el script inicie el bloque se utiliza una declaración de dos variables:

```
my $MAX = 10;
my $i;
```

Ahora hay que identificar el bucle de respuesta y los bloques del cuerpo del programa de la aplicación CGI. Como hay que responder a cada petición con una cabecera Content-Type adecuada, habrá que utilizar la siguiente línea de código:

```
print "Content type text/html\n\n";
```

La parte del bucle que se hace cargo del proceso actual es el resto del cuerpo del programa.

Ahora que se ha identificado el código, tendrá que preparar la respuesta. Para ello se utiliza el siguiente código:

```
while(FCGIaccept() >= 0) {
# aquí aparecería el código del cuerpo
    }
```

El aspecto del cuerpo será el siguiente:

```
while(FCGI::accept() >= 0) {
print "Content-type text/html\n\n";

for ($i=0; $i < $MAX; $i++){
       print "<FONT SIZE=$i>Tamaño de la fuente = $i</FONT><BR>";
}
    }
```

Las aplicaciones FastCGI se comportan como aplicaciones propias de servidor, por lo que la línea que le muestro a continuación únicamente se ejecutará si el servidor Web o el administrador le envía una señal a la aplicación:

```
exit 0;
```

En el listado 9.2 podrá encontrar el código completo de la versión FastCGI de la aplicación que acabamos de ver.

```
#!/usr/local/bin/perl

# Se le indica a Perl que queremos utilizar el paquete de FastCGI
use FCGI;

# Variables
my $MAX = 10;
my $i;

# Bucle FastCGI
while(FCGI::accept() >= 0) {

# Cabecera de contenido
print "Content-type text/html\n\n";

# Bucle principal
for ($i=0; $i < $MAX; $i++){
       print "<FONT SIZE=$i>Tamaño de la fuente = $i</FONT><BR>";
}
    }

exit 0;
```

Listado 9.2. fontsize.fcgi

Para facilitar la conversión a FastCGI, los ejecutables que se encuentran dentro de la librería de la aplicación FastCGI (se suministran con el Kit para Desarrolladores de FastCGI) se pueden ejecutar como programas en CGI o como en FastCGI, dependiendo del método utilizado para efectuar la petición. La librería detecta el entorno de ejecución y automáticamente selecciona rutinas FastCGI o I/O, según convenga.

Algo que conviene recordar a la hora de convertir las aplicaciones CGI en FastCGI es que las primeras están escritas para que no traten de ejecutar ninguna operación de control de la memoria del sistema. Esto es así porque las aplicaciones se extinguen después de su ejecución y en la mayoría de los casos el sistema puede restaurar la memoria para utilizarla en otras cosas. Además, tenemos que las aplicaciones CGI no intentan cerrar archivos, por lo que delegan esta responsabilidad en el sistema cuando terminan su proceso.

En muchos casos, es muy importante que se corrija este tipo de aplicaciones cuando se transforman a una versión FastCGI. Recuerde que las aplicaciones FastCGI residen en memoria hasta que el servidor Web o el administrador decida terminar con ellas. Si se convierte una aplicación CGI en una FastCGI sin que nadie corrija este comportamiento con la memoria, la versión FastCGI se encontrará con una serie de errores relacionados con la memoria de tal forma que hasta puede provocar un fallo del recurso. Para evitar este tipo de situaciones, conviene fijarse en estos detalles antes de efectuar la conversión. Si la aplicación CGI es muy compleja y para corregir este error hay que invertir mucho tiempo y dinero, sepa que aún tiene otra solución.

Puede tomar nota de cuántas veces ha saltado la aplicación a la memoria para atender la petición y ha finalizado la ejecución del proceso. El listado 9.3 le muestra un ejemplo para que vea que dicha cuenta se puede efectuar a través de una aplicación FastCGI basada en C.

```
#include <fcgi_stdio.h>
void main(void){
int maxRequests = 100;          // número máximo de peticiones antes
                                // de salir.
int requestCount = 0;           // se utiliza como contador.

while(FCGI_Accept() >= 0) {     // inicia el bucle de respuesta
/* body code */

printf("Content-type: text/html\r\n");
printf("\r\n");
printf("El número de peticiones procesadas es %d.", requestCount++);

/* aquí podría aparecer algún tipo de código relacionado con los
problemas de la memoria */

/* fin del código de proceso de aplicaciones */
```

```
if(requestCount >= maxRequests) {
  /* aquí aparecería el código de limpieza */
  exit 0;
  }

/* fin del código del cuerpo */

}
/ * código de limpieza o de finalización (si lo hay) */
exit(0);
}
```

Listado 9.3. memory_hog.c

Como se puede apreciar en el código fuente anterior, cuando se alcanza el número máximo de peticiones permitidas, el programa se cierra.

Otro punto con el que hay que tener cuidado es éste: si la aplicación CGI que se quiere transformar tiene código que puede llegar a interferir con otro código, hay que corregirlo. La solución a este problema es tan simple como añadir un código que se encargue de reiniciar algunas variables, arrays, etc. La aplicación ha de estar completamente segura de que cualquier estado que se cree durante el proceso no tendrá ningún efecto secundario no deseado.

Por último debe saber que la mayoría de desarrolladores CGI suelen dividir una aplicación grande en una serie de applets CGI más pequeños. Esto se hace así para compensar las faltas de iniciación asociadas a las aplicaciones CGI. Con FastCGI conviene tener una funcionalidad relativa en un mismo ejecutable para que sólo haya que controlar unos cuantos procesos y las aplicaciones puedan aprovecharse de la compartición de información a través de distintas funciones. El desarrollador de especificaciones FastCGI suministra un kit de desarrollo gratuito que facilita el desarrollo de aplicaciones en FastCGI. El kit se puede obtener a través de las páginas Web de FastCGI:

```
www.fastcgi.com/appplibs
```

Este kit, que se distribuye comprimido en formato tar, le ayudará a escribir aplicaciones FastCGI en C, C++, Perl, Tcl y Java. Cuando descomprima este archivo, creará un directorio fcgi-devel-kit. Dentro encontrará el archivo index.html, que contiene información sobre todo lo que le ofrece el kit.

FastCGI para Apache

Para que FastCGI pueda trabajar con Apache es necesario añadir un módulo. Si no tiene previsto agregar el módulo mod_fastcgi.c en su servidor

Web Apache, puede ejecutar las aplicaciones FastCGI utilizando una aplicación CGI tradicional llamada cgi-fcgi. Esta aplicación se incluye en el kit de desarrollo.

El programa cgi-fcgi le permite ejecutar aplicaciones FastCGI utilizando Apache o cualquier otro servidor que pueda trabajar con CGI. Este programa usa un dominio en UNIX o las conexiones TCP/IP para comunicar con la aplicación FastCGI. El programa cgi-fcgi toma el path o el host/puerto de la conexión por la que está escuchando y lo utiliza como parámetro. A continuación envía las variables de entorno CGI y los datos STDIN a la aplicación FastCGI que se encuentra en el servidor Web. Cuando ésta señala el final de su respuesta, cgi-fcgi limpia sus buffers y finaliza.

Obsérvese que usar cgi-fcgi no es tan bueno como tener el servidor Apache con un soporte FastCGI integrado. Como hay que abrir una aplicación CGI (cgi-fcgi) por cada petición recibida, la llamada al ejecutable tendrá que formar parte de la petición. Con este sistema tampoco se efectúa ningún control de la aplicación, por lo que tendrá que encargarse de ella el propio usuario. De todas formas, cgi-fcgi le permite construir aplicaciones que conservan el estado de la conexión en memoria, con lo que se mejora el rendimiento que es capaz de dar una aplicación CGI. Y todas las aplicaciones que desarrolle utilizando cgi-fcgi funcionarán en todos los servidores que admitan FastCGI. El archivo examples/tiny-cgi-fcgi le muestra cómo utilizar cgi-fcgi para ejecutar una aplicación, en este caso la aplicación examples/tiny-fcgi:

```
#!/path/to/cgi-fcgi/cgi-fcgi -f -connect sockets/tiny fcgi tiny-fcgi
```

En la mayoría de las plataformas UNIX, este intérprete de comandos ejecuta cgi-fcgi con los argumentos -f y examples/tiny-fcgi.cgi.

Advertencia: en algunas plataformas UNIX entre las que se encuentra HP-UX, la primera línea del archivo del intérprete de comandos no puede tener más de 32 caracteres, incluyendo el indicador de nueva línea. Es posible que tenga que instalar la aplicación cgi-fcgi en una ubicación estándar como /usr/local/bin o crear un enlace simbólico con la aplicación cgi-fcgi que se encuentra en el directorio en el que se aloja la aplicación.

Mi consejo es que si quiere comprobar las capacidades de FastCGI, utilice entonces el módulo mod_fastcgi.c en vez de la aplicación cgi-fcgi. Este módulo no es parte de la distribución estándar de Apache y tiene las siguientes directrices.

AppClass

```
Sintaxis: AppClass <path de la aplicación FastCGI> [-processes N]
[-listen-queue-depth N] [-restart delay N] [-priority N] [-port N]
[-socket sock-name] [-initial-env key=value]
Contexto: configuración del servidor
```

Esta directriz le permite iniciar las aplicaciones FastCGI. Por ejemplo:

```
AppClass /www/development/fcgi-devel-kit/echo.fcg -port 9000
```

También puede activar el módulo mod_fascgi para que cargue la aplicación FastCGI /www/development/fcgi-devel-kit/echo.fcg. La aplicación escuchará a través del puerto 9000. Cuando finaliza una de las aplicaciones FastCGI que carga esta directriz, mod_fastcgi reinicia la aplicación y escribe una entrada en el archivo de registro de errores. Los parámetros opcionales de AppClass son los siguientes:

- processes. Esta opción especifica cuántos procesos CGI se pueden llegar a crear. En un sistema sometido a mucha carga, la carga de varios procesos en la misma aplicación CGI puede afectar a la eficacia. El valor predeterminado es 1.
- listen-queue-depth. Esta opción especifica la profundidad de la cola de escucha de la aplicación FastCGI que carga esta directriz. En la mayoría de los casos basta con el valor predeterminado, pero en los sistemas sometidos a grandes cantidades de carga se puede incrementar la profundidad de la escucha. De esta forma se reduce la probabilidad de que se rechace una petición porque la aplicación está muy atareada. De todas formas, si se va a trabajar con mucha carga y su servidor Web puede ejecutar varios procesos FastCGI extra, conviene aumentar la profundidad de la cola de escucha. El valor predeterminado es de 5.
- restart-delay. Esta opción especifica el número de segundos que se utilizarán para retrasar la finalización de un proceso FastCGI. Sólo es útil cuando se usan varios ejemplos de una misma aplicación. En el caso de una única aplicación FastCGI, se reinicia inmediatamente, por lo que esta opción no tiene ningún defecto. Su valor predeterminado es 5.
  Posiblemente se esté preguntando por qué es necesario este retraso. Lo normal sería que la aplicación FastCGI no finalizase. Si se cuelga, es posible que haya algún error con ella. En este caso, este retraso le permite al servidor dedicarse a otras tareas mientras reinicia la aplicación las veces que sea necesario.

- priority. Con esta opción se determina la prioridad de proceso de una aplicación FastCGI. El valor predeterminado permite que la aplicación FastCGI tenga la misma prioridad que el propio servidor Apache. A través del ajuste de prioridad de llamada de su sistema operativo se pueden utilizar otros valores. En RedHast Linux se puede utilizar cualquier valor comprendido entre -20 y 20. Cuanto menor sea el número, tanto más favorable será el programa utilizado con el proceso. De todas formas, mod_fastcgi no admite valores negativos, por lo que un proceso FastCGI no podrá tener una mayor prioridad que el propio servidor Apache. Los números positivos reducen la prioridad de la aplicación. El valor predeterminado es 0.
- port. Esta opción especifica el puerto TCP a través del cual escuchará la aplicación FastCGI. Como los puertos que se encuentran por debajo de 1024 están reservados para los servicios estándar, tendrá que utilizar un número mayor. Con esta opción podrá acceder a su aplicación desde un sistema remoto. Por defecto no se utiliza esta variable.
- socket. Esta opción determina el nombre del path de la conexión del dominio de UNIX a través de la cual escucha la aplicación. El módulo crea esta conexión dentro del directorio especificado en la directriz FastCgiIpcDir. El valor predeterminado es Default Socket (conexión por omisión).
- initial-env. Se puede utilizar esta opción para insertar una variable de entorno (con valor incluido) en la tabla de entorno de la aplicación FastCGI. Con esta opción se multiplicarán el número de inserciones del par clave=valor en la tabla. Por defecto no se utiliza esta variable.

Nota: las opciones -socket y -port son incompatibles. En nombre del path no ha de ser igual que el suministrado en una directriz AppClass o ExternalClass anterior.

ExternalAppClass

```
Sintaxis: ExternalAppClass <nombre de aplicación FastCGI> [-host
host:port] [-socket sock-name]
Contexto: configuración del servidor
```

Utilice esta directriz cuando su aplicación FastCGI se ejecute en otro sistema. Por ejemplo:

```
ExternalAppClass echo.fcg      -host fcgi.mitec.com:9090
```

En este caso, la aplicación echo.fcg se está ejecutando en un servidor llamado fcgi.nitec.com y escucha a través del puerto 9090. <nombre de aplicación FastCGI> no es más que un identificador que se puede utilizar para determinar la aplicación que se está ejecutando en el host remoto, por eso puede tener cualquier nombre. Asegúrese de que el nombre que escoja no está siendo utilizado por otra directriz AppClass o ExternalAppClass. Los parámetros opcionales de esta directriz son:

- host. Esta opción le permite especificar el host y el número del puerto TCP de la aplicación FastCGI que se está ejecutando en otro sistema. Utilice el formato host:puerto para especificar el servidor y el número del puerto. Puede utilizar indistintamente el nombre del servidor o su dirección IP.
- socket. Esta opción le permite especificar el path de la conexión del dominio UNIX que utiliza la aplicación FastCGI.

FastCgiIpcDir

```
Sintaxis: FastCgiIpcDir path
Default: FastCgiIpcDir /tmp
Contexto: configuración del servidor
```

Esta directriz especifica el path predeterminado de la conexión de un dominio UNIX que crea el módulo. Por defecto se ubica en /tmp siempre que no tenga ningún programa cron que se encargue de limpiar su contenido. El nombre de la conexión tendrá el siguiente formato:

```
OW_WS_n.pid
```

Aquí *n* es un número y *pid* es el proceso ID del proceso principal de Apache. Cuando la utilice asegúrese de que el path determinado cuenta con los permisos de escritura y lectura adecuados.

Compilar mod_fastcgi en Apache

Añadirle a Apache el soporte para FastCGI es muy sencillo. Lo primero que hay que hacer es conseguir el último módulo de Apache. Este lo encontrará en:

```
www.fastcgi.com
```

A continuación tendrá que preparar el archivo de configuración de Apache. Copie el módulo mod_fastcgi.c) en un subdirectorio de src/modules del árbol de directorios de Apache. A mí me gusta guardar todos los módulos en el directorio src/modules/standard. Una vez que el archivo se encuentra dentro del árbol de directorios de Apache, habrá que añadir esta línea en el archivo de configuración que se encuentra en el directorio src.

```
AddModule modules/standard/mod_fastcgi.o
```

Notar si guarda el módulo en otro directorio, asegúrese de configurar correctamente esta línea código con el path utilizado en su sistema.

Ahora ya está listo para compilar Apache con el módulo FastCGI. Ejecute el script Configure. Aparecerá un nuevo ejecutable (httpd) en el directorio fuente. Para comprobar si este ejecutable utiliza el nuevo módulo bastará con escribir en el directorio fuente./httpd -I. Se generará una lista en la que tiene que aparecer el módulo mod_fastcgi.c.

Usted si su sistema puede trabajar con los comandos strip conseguirá reducir ligeramente el tamaño del ejecutable. Por ejemplo, RedHat Linux utiliza dicho comando para eliminar toda la información simbólica del binario de Apache que no tiene ninguna utilidad.

Sustituya el httpd ya existente por el nuevo ejecutable. Para configurar las aplicaciones FastCGI necesitará una combinación de las directrices de mod_fastcgi y otras propias del servidor Apache.

Para iniciar las aplicaciones FastCGI que quiera controlar con el servidor Web tendrá que utilizar la directriz AppClass. El control que efectúa el servidor es doble. Por un lado registra un mensaje de error cuando un proceso finaliza y por otro trata de reiniciarlo.

Utilice cualquiera de las directrices AppClass o ExternalAppClass (o las dos) para definir una relación entre la información de la conexión y la aplicación FastCGI. Como información de la conexión se entiende tanto el path de la conexión de un dominio UNIX, como la dirección IP y el número de un puerto TCP. La diferencia entre las dos directrices es la siguiente: una única directriz AppClass inicia una aplicación y establece una asociación con la comunicación con la que tiene que trabajar, mientras que ExternalAppClass únicamente se limita a establecer dicha relación. En el caso de AppClass, el

path que se utiliza en la asociación es el del ejecutable de la aplicación. Con ExternalAppClass, dicho path es completamente arbitrario.

Para que mod_fastcgi procese una petición HTTP, su controlador tiene que ser fastcgi-script o su tipo MIME ha de ser application/x-httpd-fcgi. Apache proporciona varias formas de configurar el controlador y el tipo MIME de la petición:

- SetHandler (en el contexto de una sección Location o Directory que se encuentre dentro del archivo de configuración de un directorio) puede asociar el controlador fastcgi-script con un archivo determinado o con todos los ficheros de un directorio.
- AddHandler puede asociar el controlador fastcgi-script con una serie de archivos basándose en su extensión.
- ForceType (en el contexto de Location, Directory o bien en el archivo .htaccess) puede asociar el tipo MIME application/x-httpd-fcgi con un archivo determinado o con todos los ficheros de un directorio.
- AddType puede asociar el tipo MIME application/x-httpd-fcgi con una serie de archivos basándose en su extensión.

Una vez que se ha configurado el módulo mod_fastcgi, estará listo para atender las peticiones de la siguiente manera:

1. Recupera la información sobre la conexión relacionada con el path. Si no la encuentra mostrará el error de código "404 Not Found".
2. El módulo mod_fastcgi conecta con el proceso de la aplicación FastCGI. Si dicha conexión falla, mostrará el error de código "500 Server Error".
3. Transmite la petición al proceso de la aplicación FastCGI, quien genera una respuesta.
4. Se recibe y convierte la respuesta de la aplicación en una respuesta HTTP. El servidor se la envía al cliente.

Ejemplo de un archivo de configuración

Le aconsejo que haga una copia de seguridad del archivo httpd.conf que tiene en su sistema y que de momento no utilice el fichero de configuración que vamos a ver como ejemplo. Su código fuente aparece en el listado 9.4 y lo he creado simplemente para probar FastCGI. Cuando vea que funciona, copie

los aspectos de la configuración propios de FastCGI en el fichero con el que trabaja en su sistema.

```
# Configuración mínima de httpd.conf para mod_fastcgi
#
# Un archivo de configuración
#
ResourceConfig /dev/null
AccessConfig   /dev/null

# El número del puerto ha de ser superior a 1024 porque no
# queremos que el servidor se confunda con el de otros servicios
# estándar.
#
Port 9999

# Conviene que sustituya la directriz User/Group por el nombre
# user/group apropiado
User $HTTP_USER
Group $HTTP_GROUP

# Configure un único subproceso para simplificar el análisis de
# errores.
StartServers 1
MinSpareServers 1
MaxSpareServers 1

# Dígale a httpd dónde tiene que residir, active el acceso y el
# registro de errores.
#ServerRoot     $APACHE
ErrorLog       logs/error.log
TransferLog    logs/access.log
ScoreBoardFile logs/httpd.scoreboard

# Indíquele a httpd dónde se encuentran los documentos
#
#DocumentRoot $FASTCGI

# Aquí se explica cómo debería colocar los archivos propios de
# las conexiones del dominio UNIX en el directorio de registro
# (es posible que prefiera crear un subdirectorio para ellos).
# No haga nada hasta que no compruebe que todo funciona
# correctamente colocando los archivos en /tmp.
# FastCgiIpcDir $APACHE/logs
# Start the echo app
#
AppClass $FASTCGI/examples/echo -initial-env SOMETHING=NOTHING

# ¿Se hace mod_fastcgi cargo de la aplicación? #
# (en cualquier otro caso el servidor devolverá un archivo binario)
#
<Location /examples/echo>
SetHandler fastcgi-script
</Location>
```

```
# Inicia una aplicación FastCGI a la que se puede acceder desde
# otras máquinas
AppClass $FastCGI/examples/echo.fcg -port 8978
<Location /examples/echo.fcg>
SetHandler fastcgi-script
</Location>

# Conecta con la aplicación remota que se acaba de iniciar. Como la
# aplicación actúa localmente, la comunicación tendrá que ser TCP.
# Para comprobar si dicha conexión remota es una realidad,
# inicie una copia de este servidor Web en una de las máquinas y
# una copia de "localhost" en la línea que se encuentra debajo
# de la sentencia que modifica el nombre de la primera máquina.
#
#ExternalAppClass remote-echo -host localhost:8978
<Location /examples/remote-echo>
SetHandler fastcgi-script
</Location>

# Es la forma en la que mod_fastcgi controla cualquier solicitud
# en la que se solicite un archivo cuyo nombre termina en .fcg:
# AddHandler fastcgi-script fcg
# Fin de httpd.conf
```

Listado 9.4. Ejemplo de configuración de httpd.conf

Asegúrese de construir un nuevo httpd con el módulo mod_fastcgi y de que el kit de desarrollo de FastCGI se ha construido correctamente. No se olvide de reiniciar Apache después de colocar el archivo httpd.conf del listado 9.4 en el directorio $APACHE/conf. Utilice un explorador Web para acceder a:

```
http://$SU_HOST:9999/ejemplos/echo
```

donde $SU_HOST es la dirección IP del host httpd que se está ejecutando. El contador de peticiones incrementará un valor cada vez que se cargue esta página. Antes de que se pueda utilizar esta configuración tendrá que efectuar las sustituciones que aparecen en la tabla 9.2.

Tabla 9.2. Sustituciones en la configuración del ejemplo

Clave	Sustituir por
$APACHE	Path del directorio en el que se encuentra Apache.
$FASTCGI	Path del directorio en el que se encuentra el kit de desarrollo de FastCGI.
$HTTP_USER	Nombre de usuario que utiliza con la directriz User.
$HTTP_GROUP	Nombre del grupo que utiliza con la directriz Group.

Parte III
La seguridad

10 Autenticación básica

Generalmente, cuando un cliente se registra en una red, la sesión se mantiene con el servidor hasta que el cliente abandona la red. En la World Wide Web, el cliente suele ser un explorador Web y el servidor, un servidor Web. De todas formas, el protocolo HTTP no es permanente, por lo que las sesiones no se mantienen de modo indefinido entre el explorador Web y un servidor Web como Apache. Tan pronto como el servidor termina de atender el URL de una petición, se corta la conexión. Apache puede utilizar Keep-Alive (conservar) para dejar la conexión abierta con el fin de procesar futuras peticiones, pero la verdad es que no todos los exploradores Web pueden trabajar con esta propiedad.

Tampoco se utiliza para conservar sesiones que necesiten una autenticación. En otras palabras, el único mecanismo que garantiza la disponibilidad en la Red es el protocolo HTTP. El resto será específico de un servidor. Es decir, los otros mecanismos de autenticación que se basan en el servidor requie n cierto grado de programación. Un sistema de autenticación como éste, que se implemente en un servidor Apache estándar, le permite controlar a qué servidores se puede acceder desde un sitio Web determinado.

De todas formas, la autenticación basada en el servidor tiene varias limitaciones. Por ejemplo, no podrá utilizarla para autenticar a los usuarios que acceden desde de la Red a través de un proxy o un firewall. Como nunca se le

revelará el IP del host al servidor Apache, el servidor no podrá diferenciar a los usuarios. La mayoría de la gente en Internet utiliza direcciones IP distribuidas dinámicamente por su ISP (proveedor de servicios de Internet), por lo que hay casos en donde una autenticación basada en el host no es muy recomendable. Por eso se usa un sistema más tradicional basado en el nombre del usuario y una contraseña. La autenticación básica de HTTP tiene esta facilidad.

El registro de autenticación apenas se utiliza en la Red porque HTTP dispone de un soporte mínimo para el acceso autenticado. Si está interesado en las arquitecturas de alta seguridad, podría plantearse trabajar con el protocolo SSL. Exceptuándolo a él, el resto de sistemas de autenticación utilizados en la Red son muy básicos.

En este capítulo veremos dos sistemas de autenticación básicos: autenticación basada en el host y autenticación HTTP básica.

Proceso de autenticación basada en el host

En este sistema de autenticación el control del acceso se basa en el nombre del host o en su dirección IP. Cuando se efectúa una petición desde un recurso determinado, el servidor Web comprueba si el host que la efectúa tiene permiso para acceder al recurso solicitado y toma una acción u otra en función de lo que encuentre.

En la distribución estándar de Apache se incluye un módulo llamado mod_access, gracias al cual se puede controlar el acceso basándose en el nombre del host de un cliente Web. Este nombre puede ser el del dominio (FQDN), como blackhole.nitec.com, o una dirección IP, como 206.171.50.50. Para controlar el acceso trabaja con las siguientes directrices: allow, deny, order, allow from env=variable y deny from env=variable.

allow

```
Sintaxis: allow from host1 host2 host3 ...
Contexto: directorio, localización, archivo de control de acceso por
directorio
Override: Limit
```

Esta directriz le permite definir una lista con los host (que puede contener un host o más o direcciones IP) que tienen permiso para acceder a un directorio determinado. Cuando se especifica más de un directorio, habrá que separarlos por espacios en blanco. En la tabla 10.1 se muestran los posibles valores que puede tomar esta directriz.

Tabla 10.1. Posibles valores de la directriz allow

Valor	Ejemplo	Descripción
Todo	allow from all	Estas palabras reservadas permiten el acceso a todos los host. El ejemplo muestra cómo utilizar esta opción.
El nombre completo de un dominio (FQDN) o de un servidor.	allow from wormhole.nitec.com	Sólo podrá acceder el servidor especificado. La directriz allow del ejemplo permite que el host wormhole.nitec.com tenga acceso. Observe que con esta opción se comparan componentes, es decir, toys.com no es igual a etoys.com.
Parte del nombre del dominio de un host.	allow from .mainoffice.nitec.com	Sólo los servidores cuyo dominio coincida con la parte especificada tendrán acceso. En el ejemplo podrán acceder todos los host que pertenezcan a la red .mainoffice.nitec.com. Por ejemplo, desarrollador1.mainoffice.nitec.com o desarrollador2.mainoffice.nitec.com tienen acceso, pero desarrollador3.baoffice.nitec.com no.
Dirección IP completa de un host.	allow from 206.171.50.50	Sólo tendrá acceso la dirección especificada. El ejemplo muestra una dirección IP completa (aparecen los cuatro octetos de la dirección IP).
Parte de la dirección IP de un host.	Ejemplo 1 :allow from 206.171.50 Ejemplo 2: allow from 103.86	Cuando no aparecen los cuatro octetos de una dirección en la directriz allow, se permitirá el acceso a las direcciones cuyos primeros octetos coincidan con la parte especificada (es decir, que pertenezcan a una misma subred). En el primer ejemplo, tendrán acceso todos los host cuya dirección IP se encuentre dentro del rango 206.171.50.1 a 206.171.50.255. Mientras que en el segundo ejemplo, todos los host de la red 130.86 tendrán acceso.
Un par red/máscara de red.	allow from 206.171.50.0/255.255.255.0	Así podrá especificar un rango de direcciones IP utilizando la dirección de una red y la de su máscara. En el ejem-

Valor	Ejemplo	Descripción
		plo vemos que las direcciones IP comprendidas entre 206.171.50.1 y 206.171.50.255 pueden acceder. De todas formas esta propiedad sólo está disponible a partir de la versión 1.3 o superior.
Una especificación CIDR de red/nnn.	allow 206.171.50.0/24	Similar a la entrada anterior, exceptuando que la máscara de red consiste en un número nnn de orden superior a 1 bit. En el ejemplo tenemos que se permite el acceso a las direcciones 206.171.50.0/255.255.255.0. Propiedad únicamente disponible a partir de la versión 1.3 de Apache.

deny

```
Sintaxis: deny from host1 host2 host3 ...
Contexto: directorio, localización, archivo de control de acceso por
directorio
Override: Limit
```

Esta directriz es exactamente opuesta a allow. Le permite definir a qué servidores se le negará el acceso a un directorio determinado. Al igual que la directriz anterior, acepta los valores que aparecen en la tabla 10.1.

order

```
Sintaxis: order deny, allow   allow,  deny | mutual-failure
Contexto: directorio, localización, archivo de control de acceso por
directorio
Override: Limit
```

Esta directriz controla el sistema de evaluación que utiliza Apache con las directrices allow y deny. Por ejemplo:

```
<Directory /mysite/myboss/rants>
order deny, allow
deny from myboss.mycompany.com
allow from all
</Directory>
```

Este ejemplo niega el acceso del host myboss.mycompany.com al directorio, mientras que se lo permite al resto de los host. El valor de la directriz order es una lista separada por comas en la que se indica qué directriz tiene preferencia. Lo normal es que la directriz que afecta a todos los servidores sea la que tenga menor prioridad. En el ejemplo anterior, como la directriz allow es la que se aplica al resto de los host, es la que tiene menor prioridad.

Aunque allow, deny y deny, allow son las más utilizadas, hay otro valor que se puede utilizar con order, mutual-failure, que indica que sólo se garantiza el acceso a aquellos host que aparezcan en la lista allow y no a los que aparezcan en la de deny.

Obsérvese que en todos los datos se evalúan las directrices allow y deny.

allow from env=variable

```
Sintaxis: allow from env=variable
Contexto: directorio, localización, archivo de control de acceso por
directorio
Override: Limit
```

Esta directriz es una variación de allow. Permite el acceso cuando se utiliza una variable de entorno. Unicamente resultará de utilidad cuando se utilicen otras directrices como BrowserMatch para determinar variables de entorno. Por ejemplo, supongamos que queremos permitir el acceso de Microsoft Internet Explorer 4.01 al directorio en el que se guardan los archivos HTML que tienen VBScript. Como el otro explorador Web de importancia, Netscape Navigator, no puede trabajar directamente con VBScript, se opta por no dejar que los usuarios de Netscape accedan a dicho directorio. En este caso se puede utilizar la directriz BrowserMatch para configurar una variable de entorno que se active cuando se detecte Microsoft Internet Explorer 4.01. El comando sería:

```
BrowserMatch "MSIE 4.01" ms_browser
```

Ahora se puede utilizar el contener <Directory> para especificar la directriz allow:

```
<Directory /path/to/Vbscript_directory >
order deny,allow
deny from all
allow from env=ms_browser
</Directory>
```

Aquí, el servidor Apache utilizará la variable de entorno ms_browser con todos los exploradores Web que suministren "MSIE4.01" como parte del

agente de identificación. La directriz allow tan sólo permitirá el acceso a aquellos exploradores para los que se haya activado la variable ms_browser.

deny from env=variable

```
Sintaxis: deny from env-variable
Contexto: directorio, localización, archivo de control de acceso por
directorio
Override: Limit
```

Esta directriz es una variación de deny. Niega el acceso cuando se utiliza una variable de entorno. Por ejemplo, supongamos que queremos negar el acceso de Microsoft Internet Explorer 4.01 al directorio en el que se guardan los archivos HTML que tienen VBScript. En este caso se puede utilizar la directriz BrowserMatch para configurar una variable de entorno que se active cuando se detecte "MSIE4.01" como parte del agente de identificación del explorador Web. El comando sería:

```
BrowserMatch "MSIE" ms_browser
```

Ahora se puede utilizar el contener <Directory> para especificar la directriz deny:

```
<Directory /path/to/Vbscript_directory >
order deny,allow
allow from all
deny from env ms_browser
</Directory>
```

Si tiene interés en bloquear el acceso a un método de petición HTTP, como puede ser GET, POST y PUT, tendrá que utilizar el contenedor <Limit>. Por ejemplo:

```
<Location /cgi-bin>
<Limit POST>
order deny,allow
deny from all
allow from yourdomain.com
</Location>
```

Este ejemplo permite que las peticiones POST accedan al directorio cgi-bin sólo si proceden del host yourdomain.com. En otras palabras, si este sitio Web tiene algún formulario HTML para enviarle al usuario a través del método HTTP POST, los únicos que podrán utilizarlo serán los usuarios que pertenezcan al dominio yourdomain.com. Normalmente las aplicaciones CGI se guardan en el directorio cgi-bin y son muchos los sitios Web que utilizan

formularios HTML para saturar de datos las aplicaciones CGI. Con la configuración de control de acceso anterior basada en el host, un sitio permitirá que cualquiera que ejecute un script CGI acceda a una ubicación determinada (en este caso, yourdomain.com) para enviar los datos de uno o más script CGI. De esta forma se consigue que el acceso de los CGI a dicha ubicación tan sólo sea de lectura. Cada persona que ejecute aplicaciones puede llegar a generar gran cantidad de información sin llegar a tomar un solo dato de los introducidos por el cliente. De esta forma, únicamente los usuarios de cierto dominio pueden introducir datos.

Proceso de autenticación de HTTP básico

La autenticación HTTP básica es muy sencilla. Cuando un explorador Web solicita un URL que está protegido por una autenticación HTTP, el servidor Web le devuelve una cabecera de estado 401 y otra de respuesta para la autenticación (véase figura 10.1). La cabecera contiene el sistema de autenticación que se está utilizando (en la actualidad sólo soporta la autenticación HTTP básica) y el nombre del dominio.

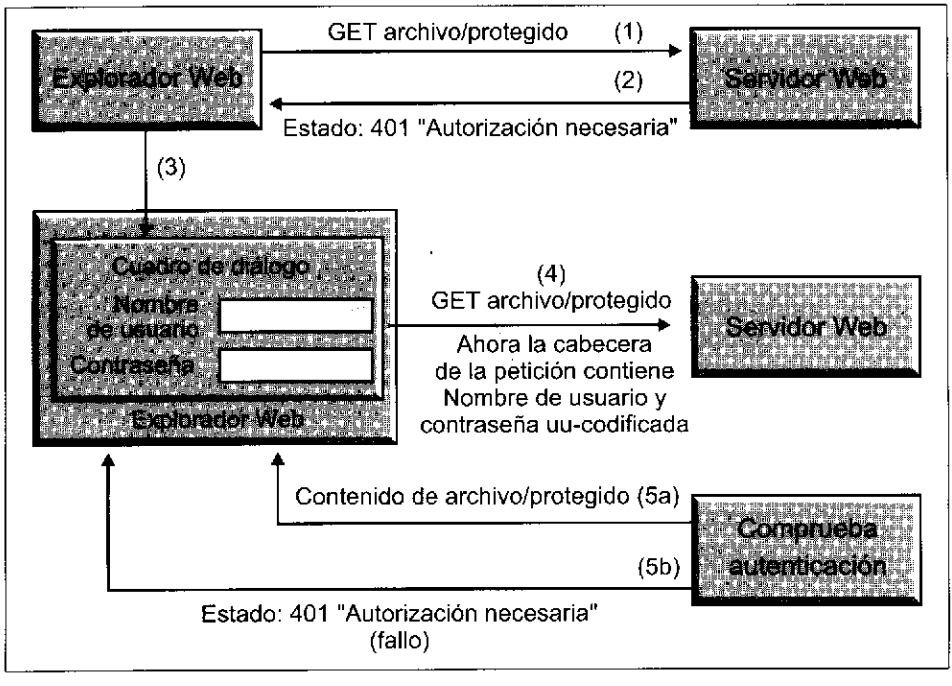


Figura 10.1. Proceso de autenticación HTTP básico

A continuación el explorador Web muestra un cuadro de diálogo que le pide al usuario que introduzca su nombre y contraseña. Cuando el usuario facilita esta información, el explorador se la envía de vuelta al usuario junto con el URL solicitado. El servidor comprueba si los datos son válidos. En caso afirmativo, mostrará la página solicitada. En caso negativo, responde con un estado 401 y vuelve a enviar la misma cabecera de autenticación.

En la siguiente llamada al servidor, el explorador se limitará a enviar la información facilitada por el usuario, con lo que se evita que el servidor genere un estado 401. Por ejemplo, si el URL http://apache.nitec.com/protected/ requiere una autenticación HTTP básica, las siguientes llamadas a http://apache.nitec.com/protected/a_page.html y http://apache.nitec.com/protected/b_page.html requerirán el par nombre/contraseña. Esta es la razón por la que el servidor envía el par antes de que se produzca ningún intercambio de autenticación (es decir, el envío de la cabecera de estado 401 y la respuesta de autenticación www) con el servidor. Este sistema es más rápido y práctico que generar una opción para cada petición recibida y hacer que el usuario introduzca una y otra vez su nombre y contraseña.

Cuando el sistema del cliente envía la contraseña (el host en el que se ejecuta el explorador Web), ni lo hace en texto plano ni la encripta. En vez de eso utiliza la codificación uu y la transmite por Internet. Este es el inconveniente de este método de autenticación, porque cualquiera que tenga un sniffer puede hacerse con el paquete IP que transporta la contraseña uu-codificada. La verdad es que este sistema de codificación está muy extendido en la actualidad y cualquiera puede conseguir un descodificador. Es decir, que cualquiera puede descodificar la contraseña y abusar de su uso. Es cierto que el sniffer tiene que localizar el paquete correcto para poder proceder a la descodificación, pero técnicamente es posible.

Esta es la razón por la que quiero resaltar que no es conveniente utilizar la autenticación HTTP básica con las aplicaciones realmente importantes. No proteja los secretos de estado con este sistema. Además, si tiene acceso Telnet o FTP a su sistema y utiliza métodos de autenticación (con esos servicios) debe saber que son muy parecidos a la autenticación HTTP básica. Si conecta su máquina a Internet sepa que cualquiera que lo desee podrá intentar acceder a ella a través de Telnet, por lo que no hay ninguna razón por la que le convenga confiar en ella.

Apache lleva algún tiempo soportando la autenticación HTTP básica. Y son muchos los módulos que se han escrito para tal fin.

Es muy sencillo crear un área restringida en la Red utilizando uno o más de los módulos de autenticación de Apache. Además de las directrices propias de dichos módulos, tendrá que utilizar las siguientes:

- AccessFileName.
- AllowOverride.
- AuthType.
- Satisfy.
- Require.

Todas estas directrices las hemos visto en capítulos anteriores. En las próximas secciones veremos los módulos de autenticación básica más comunes para Apache.

mod_auth

Este módulo se compila por defecto en la distribución estándar. Para confirmar la autenticación, utiliza los nombres de usuarios, los grupos y las contraseñas que están almacenados en archivos de texto. Este sistema funciona muy bien cuando el número de usuarios con el que se trabaja es muy pequeño. De todas formas, si tiene varios usuarios (varios cientos), el uso de mod_auth puede provocar una carencia en el rendimiento del sistema. En estos casos puede probar con algo más avanzado, como los archivos DBM (ficheros DB de Berkeley) o con una base de datos SQL.

El módulo estándar mod_auth aporta a Apache las directrices: AuthUserFile, AuthGroupFile y AuthAuthoritative.

Vamos a estudiarlas ahora con más detalle y ver algunos ejemplos de su utilización.

AuthUserFile

```
Sintaxis: AuthUserFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
Override: AuthConfig
```

Esta directriz determina el nombre del archivo de texto en el que se guardan los nombres de los usuarios y las contraseñas que se utilizan en el proceso de autenticación HTTP básico. Se ha de suministrar un path completo. Por ejemplo:

```
AuthUserFile /www/nitec/secrets/.htpasswd
```

Este archivo se suele crear con una utilidad llamada htpasswd, que se incluye como programa de soporte en la distribución estándar de Apache. El formato de este archivo es muy simple. Cada línea contiene el nombre del

usuario y la contraseña encriptada. Para encriptar la contraseña se utiliza la función estándar crypt( ).

Advertencia: es importante que el archivo AuthUserFile especificado se encuentre fuera del árbol de documentos del sitio Web, ya que si se sitúa dentro cualquiera podría copiarlo.

AuthGroupFile

```
Sintaxis: AuthGroupFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
Override: AuthConfig
```

Esta directriz especifica el archivo de texto que se utiliza como lista de los grupos de usuarios de una autenticación HTTP básica. Se ha de suministrar el path completo de dicho fichero.

Para crear este archivo se puede usar cualquier editor de texto. Su formato es el siguiente:

```
nombre_grupo: nombre_usuario nombre_usuario nombre_usuario ...
```

Por ejemplo:

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

Advertencia: es importante que el archivo AuthGroupFile especificado se encuentre fuera del árbol de documentos del sitio Web, ya que si se sitúa dentro cualquiera podría copiarlo.

AuthAuthoritative

```
Sintaxis: AuthAuthoritative on | off
Predeterminado: on
Contexto: directorio, archivo de control de acceso por directorio
Override: AuthConfig
```

Si utiliza más de un sistema de autenticación para un mismo directorio, puede usar esta directriz para que cuando haya algún error con el par nombre/contraseña del primer sistema se pase a otro.

Por ejemplo, si está utilizando mod_auth_db y mod_auth para que se hagan cargo de los servicios de autenticación y falla el par nombre/contraseña de uno de ellos, siempre que resulte posible se utilizará el siguiente módulo.

Para determinar qué módulo tiene menor prioridad se utiliza el archivo de configuración. Su contenido tendrá que ser similar a éste:

```
AddModule modules/standard/mod_auth.o
AddModule modules/standard/mod_auth_db.o
```

El módulo auth_mod_db tiene mayor prioridad pues los módulos que aparecen en el archivo de configuración se listan en orden de prioridad inverso.

En cualquier caso, cuando un par nombre/contraseña falla con todos los módulos, el servidor genera una cabecera de estado 401 y envía otra de respuesta para la autenticación WWW. Pero si el par pasa el primer módulo de autenticación, nunca se le llegará a enviar la información que contiene al segundo.

Ejemplo 1: solicitar el nombre y la contraseña del usuario

Este ejemplo le enseña cómo crear un directorio restringido para el que sea necesario suministrar el nombre de usuario y la contraseña para acceder a él. Para simplificar este ejemplo, supondremos la siguiente configuración para un sitio Web llamado apache.nitec.com:

```
DocumentRoot  /data/web/apache/public/htdocs
AccessFileName .htaccess
AllowOverride  All
```

Supongamos ahora que quiere restringir el acceso al directorio que le muestro a continuación de tal modo que la única persona que pueda acceder a él sea un usuario llamado "reader", cuya contraseña sea "bought-it".

```
/data/web/apache/public/htdocs/chapter_10/
```

Siga los pasos que le muestro a continuación para restringir el acceso.

Paso 1: crear un archivo para el usuario con htpasswd

La distribución estándar de Apache incluye un programa llamado htpasswd, que crea el archivo del usuario que necesita la directriz AuthUserFile. Utilice el programa de la siguiente manera:

```
htpasswd -c /data/web/apache/secrets/.htpasswd reader
```

La utilidad htpasswd solicita la contraseña de "reader". Introduzca "bought-it" dos veces (la segunda es para confirmar que la que se introdujo en primer lugar es correcta). Según se escribe la contraseña por segunda vez, la utilidad crea un archivo llamado .htaccess que se encuentra en el directorio /data/Web/apache/secrets. Conviene destacar los siguientes puntos:

- Utilice la opción -c para indicarle a htpasswd que lo que desea crear es un nuevo archivo de usuario. Si ya creó dicho archivo con anterioridad y lo que desea es agregar una nueva entrada, no utilice esta opción.
- Coloque el archivo del usuario fuera del árbol de directorios del sistema de apache.nitec.com, si no quiere que alguien lo pueda copiar desde la Red.
- Utilice un punto al principio del nombre del archivo para que no aparezca en la salida del sistema UNIX. La verdad es que de esta forma no se está consiguiendo ningún beneficio extra, sino que se sigue con una de las costumbres habituales de UNIX. La mayoría de los archivos de configuración (como .login y .profile) comienzan por un punto.

Para ahorrarse futuros dolores de cabeza, ejecute el siguiente comando:

```
cat /data/web/apache/secrets/.htpasswd
```

Debería aparecer una línea parecida a la siguiente:

```
reader:nu1R67Fh1sxK6
```

De esta forma se confirma que tiene un usuario llamado "reader" dentro del archivo .htpasswd. El programa ha encriptado la contraseña utilizando la función estándar crypt( ).

Paso 2: creación del archivo .htpasswd

Utilice un editor de texto, añada las siguientes líneas a un archivo llamado /data/Web/apache/public/htdocs/capítulo 10/.htaccess:

```
AuthName Apache Server Bible Readers Only
AuthType  Basic
AuthUserFile /data/web/apache/secrets/.htpasswd
require user reader
```

La primera directriz, AuthName, determina la extensión de la autenticación. No es más que una etiqueta que se le envía al explorador Web para que tenga alguna pista sobre a qué se quiere acceder. En este caso, "Unicamente Lectores de la Biblia del Servidor Apache" indicará que los únicos que pueden acceder a este directorio son los lectores de este libro. La segunda directriz, AuthType, especifica el tipo de autenticación utilizada. Como se puede trabajar con la autenticación básica, AuthType siempre toma el valor "Basic". La siguiente directriz, AuthUserFile, especifica el nombre del archivo del usuario. Aquí aparecerá el path de este archivo. La última directriz, require, especifica que el usuario "reader" puede acceder a este directorio.

Paso 3: configurar el archivo de permisos

Una vez que se hayan creado los archivos .htaccess y .htpasswd, es muy importante asegurarse de que el único que puede leerlos es Apache. Nadie a excepción del dueño y del propio Apache ha de acceder a ellos.

Paso 4: prueba

A continuación utilizamos un explorador Web para acceder al URL:

```
http://apache.nitec.com/chapter 10
```

Apache enviará una cabecera de estado 401 y una respuesta de autenticación WWW al explorador con la información sobre el alcance (determinado en AuthName) y el tipo de autenticación (determinado en AuthType). El explorador mostrará un cuadro de diálogo similar al que aparece en la figura 10.2.

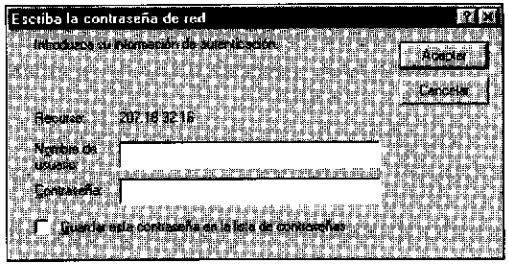


Figura 10.2. El explorador Web solicita el nombre y la contraseña

Es conveniente comprobar si se puede entrar sin utilizar ningún nombre de usuario ni contraseña, así que deje las casillas en blanco y pulse Aceptar. De esta forma se conseguirá que la autenticación falle. El explorador recibe la misma información una vez más, por lo que mostrará un cuadro de diálogo a través del cual informa que el proceso no ha tenido éxito y pregunta si se desea intentar de nuevo. Si contesta afirmativamente y repite lo hecho hasta ahora, el servidor le mostrará un mensaje similar al de la figura 10.3.

Si quiere acceder de nuevo al cuadro de diálogo a través del cual introduce su nombre de usuario y contraseña tendrá que pulsar el botón Actualizar. Si introduce esta vez como nombre de usuario "reader" y como contraseña "bought-it" y pulsa Aceptar, Apache le dejará acceder al directorio protegido.

Nota: puede cambiar el mensaje que aparece en la pantalla anterior por otro más general, utilizando la directriz ErrorDocument:

```
ErrorDocument 401 /nice_401message.html
```

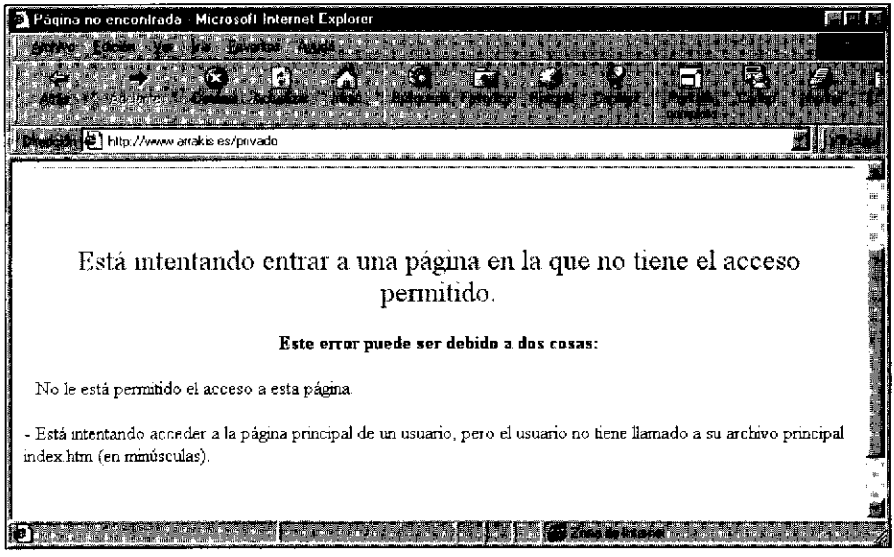


Figura 10.3. Mensaje obtenido al fallar el proceso de autenticación

Inserte esta línea en su archivo srm.conf y cree el mensaje que más le guste dentro del archivo nice_401message.html.

Ejemplo 2: permitir que un grupo de usuarios acceda a un directorio

En vez de permitir que un único usuario ("reader") acceda a un área restringida, ahora permitiremos que cualquiera que pertenezca a un grupo de usuarios llamado asb_readers acceda al mismo directorio. Supongamos que deseamos crear la tabla de usuarios que aparece a continuación.

Tabla 10.2. Tabla de usuarios

Grupo	Nombre de usuario	Contraseña
asb_readers	pikejb	Red#hat
asb_readers	bcaridad	Net#rat

Siga estos pasos para que los usuarios de la tabla anterior accedan al directorio protegido.

Paso 1: crear un archivo para el usuario con htpasswd

Cree los usuarios pikejb y bcaridad con htpasswd.

Paso 2: crear un archivo para el grupo

Por medio de un editor de texto como vi (disponible en la mayoría de sistemas UNIX), cree un nombre llamado /data/Web/apache/secrets/.htgroup. Este archivo no tendrá más que una línea, cuyo aspecto será el siguiente:

```
asb_readers: pikejb bcarida
```

Paso 3: crear un archivo .htaccess en /data/web/apache/public/ htdocs/capítulo_10

Con un editor de texto, tendrá que agregar estas líneas de código en el archivo /data/web/apache/public/ htdocs/capítulo_10/.htaccess:

```
AuthName Apache Server Bible Readers Only
AuthType  Basic
AuthUserFile /data/web/apache/secrets/.htpasswd
AuthGroupFile /data/web/apache/secrets/.htgroup
require group  asb_readers
```

Casi es la misma configuración que hemos visto en el ejemplo anterior, pero con dos cambios. El primero es que se ha añadido una directriz nueva, AuthGroupFile, que señala al archivo .htaccess del grupo que acabamos de crear. El siguiente cambio se encuentra en la línea de la directriz, que ahora solicita un grupo llamado "asb_readers". En otras palabras, Apache permitirá el acceso a cualquiera que pertenezca a ese grupo. Observe que podría haber bastado con agregar esta líneas:

```
require user pikejb bcaridad
```

en vez de:

```
require group asb_readers
```

De todas formas, listar a todos los usuarios en dicha línea podría haber supuesto un fuerte dolor de cabeza. Por medio de los grupos se pueden eliminar y agregar conjuntos de usuarios sin ninguna dificultad.

Paso 4: permisos de los archivos

Al igual que ocurría en el ejemplo anterior, es importante hacer que Apache sea el único que pueda leer los archivos .htaccess, .htpasswd y .htgroup y que nadie, a excepción de su dueño, pueda acceder a ellos.

Paso 5: prueba

La prueba de esta nueva configuración es muy parecida a la vista en el ejemplo anterior, así que no repetiremos el proceso aquí.

Ejemplo 3: mezclar el control de acceso basado en el host con la autenticación HTTP básica

En este ejemplo, verá cómo se puede combinar el sistema de control de acceso basado en el host y el método de autenticación básica. Supongamos que queremos que el grupo "asb_readers" acceda al mismo directorio que se ha visto en el ejemplo 2 y que cualquiera que provenga de un dominio llamado apache-training.nitec.com pueda acceder al mismo directorio sin necesidad de contraseña. En otras palabras, si una petición para el URL http://apache.nitec.com/capítulo_10 proviene del dominio apache-training.nitec.com, la petición se atenderá sin ningún tipo de autenticación. Siga estos pasos.

Paso 1: modificar el archivo .htaccess

Modifique el archivo .htaccess (que se encuentra en el ejemplo 2) hasta que ofrezca el siguiente aspecto:

```
AuthName Apache Server Bible Readers Only
AuthType  Basic
AuthUserFile /data/web/apache/secrets/.htpasswd
AuthGroupFile /data/web/apache/secrets/.htgroup
require group  asb_readers
order deny, allow
deny from all
allow from ap
```

Con estas líneas de código hemos añadido las tres directrices vistas en las secciones anteriores. La primera es la directriz order, que le dice a Apache que evalúe en primer lugar la directriz deny antes de proceder con allow. Con ella se le ordenará al servidor que no le permita el acceso a nadie. Allow le indica que podrán acceder los usuarios procedentes del dominio apache-training.nitec.com.

Paso 2: prueba

Utilice un explorador Web desde un host llamado user1.apache-training.nitec.com. Si intenta acceder al URL http://apache-training.nitec.com/capítulo_10, el explorador le mostrará el cuadro de diálogo de autenticación a través del cual tiene que insertar el nombre y la contraseña. Esto no es lo que queremos que suceda. Entonces, ¿qué ocurre? Apache asume que en este directorio hay que aplicar los dos sistemas de autenticación, por lo que no permite que se acceda al directorio sin pasar por ambos métodos. Una solución a este problema podría ser utilizar la directriz satisfy:

```
AuthName Apache Server Bible Readers Only
AuthType  Basic
```

```
AuthUserFile /data/web/apache/secrets/.htpasswd
AuthGroupFile /data/web/apache/secrets/.htgroup
require group  asp_readers
order deny, allow
deny from all
allow from apache-training.nitec.com
satisfy any
```

La directriz satisfy toma o bien todos los valores o bien cualquiera de ellos. Como sólo queremos que sea la autenticación HTTP básica la que se utilice cuando se accede desde el dominio apache-training.nitec.com, utilizaremos "any". En realidad se le está diciendo al servidor Apache lo siguiente:

```
Si la petición proviene de cualquier host que se encuentre en el
dominio apache-training.nitec.com entonces
  No hay que utilizar la autenticación HTTP básica
Si no
  Necesaria autenticación HTTP básica
Fin
```

Si cambia de opinión y desea que únicamente los usuarios del dominio apache-training.nitec.com accedan al directorio con la autenticación HTTP básica, especifique "all". De esta forma le indicará a Apache que siempre tiene que utilizar ambos métodos de autenticación.

mod_auth_dbm

Como hemos comentado anteriormente, los archivos de texto .htpasswd y .htaccess no son adecuados en los procesos de gran velocidad y puede afectar negativamente a la eficacia de un servidor Web cuando se tiene que hacer cargo de cientos de usuarios (o más). En estos casos hay que automatizar el proceso de autenticación y para eso se utiliza el módulo mod_auth_dbm. En vez de trabajar con archivos txt, lo hace con DBM. Estos ficheros guardan los datos como pares clave=valor y mantienen una tabla indexada con todas las claves del archivo. Gracias a ella se puede recuperar la información asociada con la clave en menos tiempo del que haría falta para analizar un gran archivo de texto de cientos de entradas.

Aunque son muchas las DBM disponibles, las más comunes son GDBM, NDBM, SDBM y Berkeley DB (BSD-DB). La tabla 10.3 nos muestra las propiedades de estas DBM.

Esta tabla se basa en la información incluida en la documentación de Perl 5. Antes de que pueda usar una DBM con Apache tendrá que asegurarse de que la que escoja está instalada en su sistema. Compruebe que los archivos de la librería se encuentran en el directorio predeterminado para tal fin. También

es muy posible que vaya a necesitar Perl. Asegúrese de compilar la última versión de Perl.

Tabla 10.3. Propiedades de DBM

Propiedades	NDBM	SDBM	GDBM	BSD-DB
Restricciones por licencia	Desconocido	No	Sí	No
Independiente del orden de los bytes	No	No	No	Sí
Límite de tamaño predeterminado	4K	1K	Ninguno	Ninguno
Crea archivos FTP seguros	No	Sí	Sí	Sí
Velocidad	Desconocida	Lenta	Media	Mucha
Tamaño de la base de datos	Desconocido	Pequeña	Grande	Media
Tamaño del código	Desconocido	Pequeño	Grande	Grande
La fuente viene con Perl	No	Sí	No	No

Truco: puede bajarse Perl de http://www.perl.com. Configure Perl para que el soporte DMS sea de los más sencillos. Todo lo que tiene que hacer es ejecutar el script de configuración y le preguntará con qué DBM desea trabajar. Por ejemplo, si escoge NDBM o GDBM y tiene sus archivos instalados en el sistema, el script le preguntará qué parámetro desea utilizar en la compilación de Perl, -lndbm o -lgdbm.

Una vez que ha instalado las librerías DBM indicadas, tendrá que configurar Apache para que pueda trabajar con los archivos DBM, ya que la distribución estándar de Apache no puede. Antes de volver a compilarlo, conviene que revise el ejecutable de Apache (httpd) para ver si dispone de dicho módulo precompilado.

El siguiente comando le mostrará todos los módulos que se compilan en un ejecutable httpd:

```
/usr/sbin/httpd -l
```

Si en la lista que aparece en pantalla no se encuentra el módulo mod_auth_dbm.c, tendrá que volver a configurar y compilar Apache. Para añadir el módulo de autenticación DBM edite su archivo de configuración, que se encuentra en el directorio fuente de Apache, y quite el símbolo de comentario de la línea:

```
#AddModule modules/standard/mod_auth_dbm.o
```

Grabe el archivo y ejecute el script Configure para crear un nuevo Makefile. Cuando lo tenga, ejecute dicha utilidad para crear el nuevo ejecutable de Apache (httpd).

Truco: si tiene algún problema durante la compilación, añada -l<nombre de su dbm> a EXTRA_LIBS, que se encuentra dentro del archivo Configure. Tendrá que volver a ejecutar el script de configuración para que se utilicen los cambios. Si tuviese algún problema, lo mejor que puede hacer es intentarlo con GNU GDBM porque lleva bastante tiempo utilizándose en todo tipo de sistemas y porque obtendrá la ayuda que necesite a través de USENET.

Cuando tenga el nuevo httpd, ejecute el comando httpd -l para comprobar que mod_auth_dbm.c se encuentra dentro de la lista de módulos. Ahora puede eliminar todos los comandos inservibles que se encuentran dentro del ejecutable con el comando strip:

```
strip httpd
```

Sustituya el httpd antiguo por el nuevo.

Truco: antes de proceder a la sustitución, conviene que lo renombre por si se encuentra con que el nuevo causa problemas. En este caso siempre podrá volver a utilizar el httpd antiguo sin tener que suspender su servicio Web.

Una vez que Apache está correctamente compilado para que utilice los archivos DBM, habrá que utilizar dbmmanage para crear un archivo para el usuario DBM. Comience por utilizar el script dbmmanage que se encuentra en el directorio de soporte de la distribución estándar de Apache para crear un archivo para el usuario basado en DBM. El script dbmmanage en Perl puede crear algunos de los archivos DBM más populares, entre los que tenemos

NDBM, GDBM y Berkeley DB. También se puede utilizar este script para crear nuevos archivos DBM, añadir usuarios, modificar contraseñas, eliminar usuarios o ver la información disponible sobre ellos. Antes de utilizar este script, tendrá que modificar una de sus líneas de código para que aparezca la DBM que desea utilizar en primer lugar del array ISA.

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

Por ejemplo, si piensa trabajar con archivos GDBM, modifique la línea para que tenga el siguiente aspecto:

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

Para saber qué opciones puede utilizar ejecute el siguiente comando:

```
./dbmmanage
```

De esta forma accederá a todas las opciones posibles. Para crear un archivo DBM nuevo que se llame /www/secrets/myuserdbm, añadiendo al usuario llamado "reader", escriba el siguiente comando:

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

El script le pedirá que introduzca (un par de veces) la contraseña del usuario "reader". Cuando lo haga habrá añadido el nombre del usuario y la contraseña encriptada al archivo DBM.

Para acceder a la lista en la que aparecen todos los usuarios incluidos en archivo DBM tendrá que utilizar el siguiente script:

```
dbmmanage /path/a/archivo/DBM view
```

Una vez que haya recompilado Apache para que disponga de soporte DBM, podrá utilizar el módulo mod_auth_dbm para que automatice el proceso de autenticación. Recuerde que con Berkeley DB tiene que usar el módulo mod_auth_db en vez de mod_auth_dbm.

El módulo mod_auth_dbm dispone de las siguientes directrices: AuthDBMUserFile, AuthDBMGroupFile y AuthDBMAuthoritative. Vamos a estudiar detalladamente cada una de ellas y a ver unos ejemplos de su uso.

AuthDBMUserFile

```
Sintaxis: AuthDBMUserFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el path del archivo DBM que va a utilizarse durante el proceso de autenticación. El archivo contiene un par clave=valor por cada entrada. El nombre del usuario es la entrada, mientras que la contraseña, que se encripta con crypt(), es el valor. Cada campo del registro está separado por dos puntos y después de los campos pertenecientes al nombre y a la contraseña del usuario se puede incluir cualquier dato.

Truco: nunca guarde los archivos de sus bases de datos dentro del árbol de directorios del sistema Web.

AuthDBMGroupFile

```
Sintaxis: AuthDBMGroupFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el path completo del archivo del grupo que contiene la lista de los grupos de usuarios. Cada registro lo forma un par clave=valor, donde el nombre del usuario es la clave y el valor es la lista de los nombres (separados por comas) de los grupos a los que pertenece.

Si prefiere no utilizar un grupo de usuarios separado, puede usar un único archivo DBM que se encargue de suministrar la información sobre las contraseñas y los grupos a los que pertenece el usuario. El formato de este archivo sería el siguiente:

```
nombre de usuario: contraseña encriptada: lista de grupos separada
por comas
```

El nombre de usuario es la clave, y la contraseña y la lista de grupos, los dos campos del valor. Después de otros dos puntos se puede incluir toda la información que se desee. El módulo de autenticación la ignorará. Si usa DBM para suministrar la información sobre las contraseñas o los grupos de usuarios, tendrá que hacer que las directrices AuthDBMGroup y AuthDBMGroupFile se dirijan al mismo archivo.

AuthDBMAuthoritative

```
Sintaxis: AuthDBMAuthoritative on | off
Predeterminado: on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se utilizan varios sistemas de autenticación, como mod_dbm y mod_auth en el mismo directorio, se puede aprovechar esta directriz para definir en qué casos mod_auth_dbm será el sistema que se va a utilizar.

Por defecto, el valor de la directriz permite que mod_auth_dbm sea el sistema de autenticación del directorio. Es decir, que si la autenticación DBM falla para algún usuario determinado, no se pasarán sus credenciales a otro sistema de autenticación. Cuando se toma el valor off, sí que se pasará dicha información al siguiente nivel de autenticación.

Un uso muy corriente de este sistema lo tenemos en la combinación de los módulos auth básicos, como mod_auth.c, mientras que el módulo DBM suministra el peso de la comprobación de las credenciales del usuario. Muy pocos administradores permiten que esta información pase al siguiente nivel de autenticación.

Ejemplo: solicitar el nombre y la contraseña de un usuario DBM

Una vez que hemos creado el archivo DBM del usuario, podemos restringir el acceso a cualquier directorio de la Red. En el siguiente ejemplo, asumiremos que el archivo DBM es /www/secrets/myuserdbm. Puede agregar el sistema de autenticación tanto al servidor global como al virtual, utilizando el contenedor <Diretory>, o bien puede utilizar el archivo .htaccess (no hay ninguna diferencia entre los dos métodos). La configuración de nuestro ejemplo tendría el siguiente aspecto.

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

Ahora Apache utilizará el módulo mod_auth_dbm para controlar la autenticación del directorio en el que se aplica esta configuración.

Advertencia: asegúrese de que el único que puede leer el archivo dbm es el dueño de Apache. No conviene que nadie más tenga permiso de lectura ni escritura sobre él.

mod_auth_db

Si su sistema no puede utilizar DBM, pero sí soporta los archivos Berkeley DB, aún puede utilizar el módulo mod_auth_db. Al igual que ocurría con el

módulo DBM, este módulo no está compilado en la distribución estándar de Apache, por lo que tendrá que volver a compilarlo para que el servidor lo soporte. Antes de volver a compilarlo, conviene que revise el ejecutable de Apache (httpd) para ver si dispone de dicho módulo precompilado. El siguiente comando le mostrará todos los módulos que se compilan en un ejecutable httpd:

```
/usr/sbin/httpd -l
```

Si en la lista que aparece en pantalla no se encuentra el módulo mod_auth_db.c, tendrá que volver a configurar y compilar Apache.

Antes de nada conviene comprobar que su sistema cuenta con los archivos de la librería DB. Por ejemplo, en un sistema Linux, los archivos se encuentran en el directorio /usr/lib. Si su sistema carece de estos archivos, tendrá que hacerse con el código fuente correspondiente y empezar por compilar el soporte DB. Puede encontrar información sobre la librería DB en el siguiente URL:

```
www.sleepycat.com/
```

Una vez que está seguro de que su sistema cuenta con las librerías DB, puede volver a configurar y compilar Apache. Quite el símbolo de comentario de la línea:

```
#AddModule modules/standard/mod_auth_db.o
```

Grabe el archivo y ejecute el script Configure para crear un nuevo Makefile. Cuando lo tenga, ejecute dicha utilidad para crear el nuevo ejecutable de Apache (httpd). Cuando tenga el nuevo httpd, ejecute el comando httpd -l para comprobar que mod_auth_db.c se encuentra dentro de la lista de módulos. Sustituya el httpd antiguo por el nuevo.

Por fin está listo para utilizar el módulo mod_auth_db. Las directrices propias de este módulo son las siguientes: AuthDBUserFile, AuthDBGroupFile y AuthDBAuthoritative.

AuthDBUserFile

```
Sintaxis: AuthDBUserFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el path completo del archivo DB del usuario que contiene la lista de usuarios y sus contraseñas encriptadas.

Al igual que DBM, el archivo de usuario DB tiene un par clave=valor, en el que la clave es el nombre del usuario y el valor, la contraseña encriptada con crypt( ).

Advertencia: asegúrese de que los archivos de los usuarios se encuentran fuera del árbol de directorios y de que el único que puede leerlos es Apache. Nadie, a excepción del propietario, debe tener acceso de lectura ni escritura a dichos ficheros.

AuthDBGroupFile

```
Sintaxis: AuthDBGroupFile nombre_archivo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el path completo del archivo del grupo que contiene la lista de los grupos de usuarios. Cada registro lo forma un par clave=valor, donde el nombre del usuario es la clave y el valor la lista de los nombres (separados por comas) de los grupos a los que pertenece. El valor no puede contener espacios en blanco y nunca ha de tener dos puntos.

Si prefiere no utilizar un grupo de usuarios separado, puede usar un único archivo DBM que se encargue de suministrar la información sobre las contraseñas y los grupos a los que pertenece el usuario. Su formato sería el siguiente:

```
nombre de usuario : contraseña encriptada: lista de grupos separada
por comas
```

El nombre de usuario es la clave, y la contraseña y la lista de grupos, los dos campos del valor. Después de otros dos puntos se puede incluir toda la información que se desee. El módulo de autenticación la ignorará. Si usa DBM para suministrar la información sobre las contraseñas o los grupos del usuario, tendrá que hacer que las directrices AuthDBGroup y AuthDBGroupFile se dirijan al mismo archivo.

AuthDBAuthoritative

```
Sintaxis: AuthDBAuthoritative  on | off
Predeterminado: on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se utilizan varios sistemas de autenticación, como mod_db, mod_dbm y mod_auth en el mismo directorio, se puede aprovechar esta directriz

para definir en qué casos mod_auth_db será el sistema que se va a utilizar. Por defecto, el valor de la directriz permite que mod_auth_db sea el sistema de autenticación del directorio. Es decir, que si la autenticación DB falla para algún usuario determinado, no se pasarán sus credenciales a otro sistema de autenticación. Cuando se toma el valor off, sí que se pasará dicha información al siguiente nivel de autenticación.

Aunque utilizar los archivos DBM o DB facilita la labor de administración de una base de datos grande, lo normal es que no sean la mejor opción para las organizaciones que tienen los datos de sus usuarios guardados en bases de datos SQL. Estas bases de datos proporcionan un gran nivel de disponibilidad, pero no pueden funcionar con archivos DBM o DB. Apache también tiene soporte para trabajar con los archivos de estas bases de datos, que reciben el nombre de mSQL.

mod_auth_msql

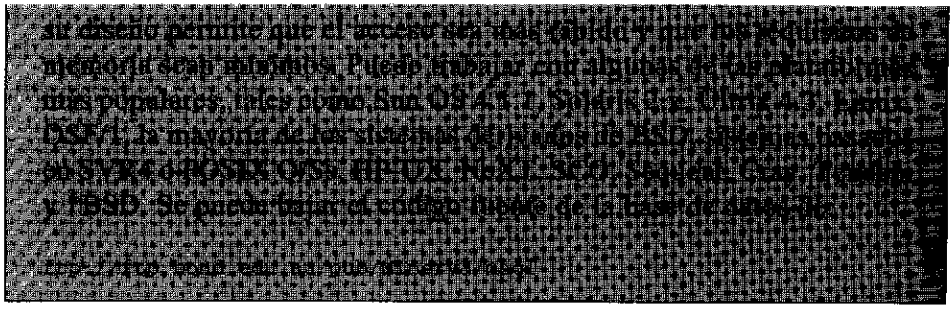
Con este módulo podrá usar la autenticación básica HTTP en los casos en los que la información sobre el nombre de los usuarios y sus contraseñas se encuentre en una base de datos mSQL. Tiene que crear una base de datos mSQL que contenga dicha información. Esto es algo que se escapa al propósito de esta obra. Si quiere detalles que le expliquen cómo hacerlo, consulte la documentación de mSQL. De todas formas, cuando cree una base de datos para usuarios y utilice el módulo mod_auth_msql, asegúrese de que tiene en cuenta las siguientes consideraciones:

- El campo en el que se guardará el nombre del usuario no puede tener más de 32 caracteres. Son muchos los exploradores Web que no pueden introducir más de 32 caracteres en los campos que se encuentran en el cuadro de diálogo de autenticación.
- Un nombre de usuario no puede tener ningún espacio en blanco

El campo de contraseñas tiene las mismas restricciones que el de usuario, pero nunca conviene utilizar contraseñas que tengan menos de 5 caracteres, ya que son muy fáciles de averiguar.

mSQL

El término mSQL proviene de mini-SQL. Es una máquina para bases de datos SQL desarrollada por David J. Hughes, de la universidad de Bond, Australia. Aunque mSQL tan sólo suministra un conjunto ANSI SQL,



Al igual que los módulos de autenticación DB o DBM, este módulo no se encuentra en la distribución estándar de Apache, por lo que tendrá que añadir la siguiente línea al archivo de configuración:

```
AddModule modules/standard/mod_auth_msql.o
```

Se supone que tiene los archivos de mod_auth_msql en el directorio modules/standard de la distribución fuente de Apache. Una vez que ha añadido esta línea de código, ejecute el script Configure y haga la utilidad, en este orden. Utilice el comando httpd -l para asegurarse de que httpd tiene el código de mod_auth_msq.c.

Vamos a ver un ejemplo de la configuración de control de acceso (que aparece en el listado 10.1).

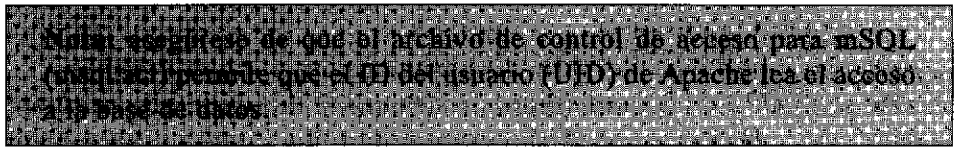
Por ejemplo, haré las siguiente suposiciones:

- El servidor mSQL se encuentra en la misma máquina que el servidor Apache.
- El nombre de la base de datos mSQL que contiene la información de los usuarios se llama asb_readers_db.
- El nombre de la tabla mSQL que contiene la información del usuario se llama asb_users_table.
- El nombre del archivo mSQL que contiene el campo del nombre del usuario se llama Uname.
- El nombre del archivo mSQL que contiene el campo de la contraseña del usuario se llama Passwd.
- El nombre de la tabla mSQL que contiene la información sobre los grupos se llama asb_group_table.
- El nombre del archivo mSQL que contiene el campo del grupo se llama asb_readers.

```
Auth_MSQLhost localhost
Auth_MSQLdatabase asb_readers_db.
Auth_MSQLpwd_table asb_users_table
Auth_MSQLuid_field Uname
Auth_MSQLpwd_field Passwd
Auth_MSQLgrp_table asb_group_table
Auth_MSQLgrp_field  asb_readers
Auth_MSQL_nopasswd off
Auth_MSQL_Authoritative on
Auth_MSQL_EncryptedPasswords on
AuthName Apache Server Bible Readers Only
AuthType  Basic
require valid-user
```

Listado 10.1. Ejemplo de configuración para el control de acceso

He utilizado el valor predeterminado con Auth_MSQL_nopasswd, Auth_MSQL_Authoritative y Auth_MSQL_EncryptedPasswords.



Después de esto, debería estar listo para utilizar el módulo mod_auth_msql, que tiene las siguientes directrices:

Auth_MSQLhost

```
Sintaxis: Auth_MSQLhost  nombre_host   dirección_IP
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre del host del sistema que ejecuta el servidor mSQL (un demonio). Si no se configura la directriz o su valor es localhost, el módulo tendrá que utilizar /dev/msql en vez de una conexión con el servidor mSQL. Si este servidor se encuentra en localhost, se recomienda el uso de /dev/msql.

Obsérvese que el servidor Apache (UID) tiene que permitir el acceso al servidor mSQL.

Auth_MSQLdatabase mSQL

```
Sintaxis: Auth_MSQLdatabase mSQL_nombre_de_la_base_de_datos
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre de la base de datos que utiliza el servidor mSQL y que contiene la información de los usuarios.

Auth_MSQLpwd_table mSQL

```
Sintaxis: Auth_MSQLpwd_table mSQL nombre de la tabla que contiene la
información del usuario
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre de la tabla que contiene (como mínimo) el nombre del usuario y su contraseña encriptada. El nombre del usuario tendrá que estar configurado como "clave" (key).

Auth_MSQLgrp_table mSQL

```
Sintaxis: Auth_MSQLgrp_table mSQL nombre de la tabla que contiene la
información del grupo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre de la tabla que contiene la información sobre el grupo. Como mínimo tiene los campos del nombre del usuario y el del grupo al que pertenece. Tenga en cuenta que si un usuario pertenece a varios grupos tendrá varias entradas en la tabla.

Auth_MSQLuid_field mSQL

```
Sintaxis: Auth_MSQLuid_field mSQL nombre del campo en el que se
guarda el nombre del usuario
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre del campo especificado en la tabla Auth_MSQLgrp_table en el que se guarda la información sobre el nombre del usuario. Si crea una tabla de un grupo usando la directriz Auth_MSQLgrp_table, con ella especificará el nombre del campo en el que se guarda la información sobre la tabla de grupos. En otras palabras, tiene que tener los mismos nombres para Auth_MSQLwgr_table y Auth_MSQLgrp_table.

Auth_MSQLpwd_field mSQL

```
Sintaxis: Auth_MSQLpwd_field mSQL nombre del campo en el que se
guarda la contraseña
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre del campo en el que se guarda la contraseña especificado en la tabla Auth_MSQLpwd_table.

Auth_MSQLgrp_field mSQL

```
Sintaxis: Auth_MSQLgrp_field mSQL nombre del campo en el que se
guarda el nombre del grupo
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Esta directriz determina el nombre del campo en el que se guarda el nombre del grupo de la tabla Auth_MSQLgrp_table.

Auth_MSQL_nopasswd

```
Sintaxis: Auth_MSQL_nopasswd on | off
Predeterminado: off
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se activa, el módulo mod_auth_msql no efectuará ninguna comparación de contraseñas si los campos de contraseñas de la tabla de la base de datos están vacíos. El valor predeterminado se asegura de que la gente no se aproveche de los campos de contraseñas vacíos y acceda al sistema.

Auth_MSQL_Authoritative

```
Sintaxis: Auth_MSQL_Authoritative on | off
Predeterminado: on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

El valor predeterminado hace que se utilice el módulo mod_auth_msql como sistema de autenticación. En otras palabras, cuando un usuario suministra sus credenciales (el par nombre/contraseña) y falla la autenticación, no se le entregan los datos al siguiente módulo.

Auth_MSQL_EncrytedPasswords

```
Sintaxis: Auth_MSQL_EncrytedPasswords on | off
Predeterminado: on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

El valor predeterminado le indica al módulo mod_auth_msql que encripte las contraseñas utilizando la función estándar crypt( ). Si el valor fuese off, la contraseña se guardaría en texto plano.

mod_auth_anon

Este módulo permite acceder de forma anónima a ciertas áreas del sistema. Si está familiarizado con los servidores FTP anónimos, su configuración es muy parecida. Todos los usuarios pueden utilizar un ID llamado "anonymous" y su dirección de correo electrónico para evitar que los incluya en los archivos de registro o en las listas de mail.

Al igual que los módulos DBM que hemos visto antes, tendrá que añadir este módulo al archivo de configuración Configuration:

```
AddModule modules/standard/mod_auth_anon.o
```

A continuación ejecute Configure y haga la utilidad que se encarga de crear el nuevo ejecutable httpd.

Ahora ya debería poder usar el módulo mod_auth_anon. Vamos a ver detenidamente las directrices de este módulo y algunos ejemplos de su utilización.

Anonymous

```
Sintaxis: Anonymous  usuario usuario ...
Predeterminado: nada
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Con esta directriz puede especificar uno o más nombres de usuario que se pueden utilizar para acceder a las áreas restringidas. Conviene que en la lista de usuarios haya una entrada para "anonymous", ya que su utilización está muy extendida. Si escoge un nombre de usuario en el que aparezca un espacio en blanco, asegúrese de que va entre comillas. Por ejemplo:

```
Anonymous "Usuario sin registrar" anonymous
```

o

```
Anonymous 'Usuario sin registrar' anonymous
```

En estas cadenas no se diferencian mayúsculas y minúsculas.

Anonymous_Authoritative

```
Sintaxis: Anonymous_Authoritative on | off
Predeterminado: Anonymous_Authoritative off
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se activa esta directriz (se usa el valor on), la autenticación anónima se convierte en la predeterminada para el directorio que se quiere proteger. En otras palabras, no se utilizará ningún otro método de autenticación.

Anonymous_LogEmail

```
Sintaxis: Anonymous_LogEmail on | off
Predeterminado: Anonymous_LogEmail on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se activa esta directriz (se utiliza el valor on), cualquier cosa que se introduzca en el campo contraseña del cuadro de diálogo que muestra el explorador Web se guardará en el archivo de registro de acceso de Apache.

Anonymous_MustGiveEmail

```
Sintaxis: Anonymous_MustGiveEmail on | off
Predeterminado: Anonymous_MustGiveEmail on
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se activa esta directriz se permite rechazar todas las solicitudes que no proporcionan su contraseña en forma de correo electrónico. De todas formas, no hay que fiarse de las direcciones que introduce la gente en este campo, ya que no hay ninguna forma de comprobar su veracidad.

Anonymous_NoUserID

```
Sintaxis: Anonymous_NoUserID on | off
Predeterminado: Anonymous_NoUserID off
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Si desea que sus usuarios dejen vacío el campo del nombre en el cuadro de diálogo emergente, active esta directriz. En caso contrario se pedirá que se introduzca un nombre de usuario que ha de coincidir con los valores que aparecen en la directriz Anonymous.

Anonymous_VerifyEmail

```
Sintaxis: Anonymous_VerifyEmail on | off
Predeterminado: Anonymous_VerifyEmail off
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
Override: AuthConfig
```

Cuando se activa esta directriz, la contraseña tendrá que ser una dirección válida de correo electrónico. De todas formas, se comprueba la validez. El módulo busca que el campo de la contraseña tenga el símbolo @ y un punto (.). Si la contraseña introducida tiene alguno de estos símbolos, entonces se acepta.

Ejemplo: restricción de acceso anónimo

La configuración que aparece a continuación le muestra cómo se pueden utilizar las directrices anteriores para suministrar un acceso anónimo a un directorio:

```
Anonymous_NoUserId off
Anonymous_MustGiveEmail on
Anonymous_VerifyEmail on
Anonymous_LogEmail on
Anonymous anonymous guest "No lo se"
AuthName Use 'anonymous' & Email address for guest entry
AuthType basic
require valid-user
```

mod_auth_external

Hasta ahora hemos hablado de varios métodos de autenticación para los que hemos utilizado el código que encontramos en los módulos. Pero, ¿qué pasaría si tuviésemos que trabajar con un sistema de autenticación externo? Pues que precisamente para eso tenemos el módulo mod_auth_external. Apache puede trabajar con programas externos porque este módulo está compilado en el ejecutable del servidor.

Para instalar mod_auth_external tendrá que añadir el módulo al archivo de configuración Configuration. Por medio de un editor de texto, agregue la siguiente línea:

```
AddModule modules/standard/mod_auth_external.o
```

Obsérvese que si tiene pensado guardar este módulo en otro directorio que no sea modules/standard que se encuentra en el árbol de directorios de Apache, tendrá que cambiar el path que aparece en la línea de código anterior por el que utilice en su sistema.

Ahora, ejecute el script Configure para crear un nuevo archivo Makefile, que al ejecutarlo creará el nuevo httpd del servidor. Ahora tiene que escribir el siguiente comando:

```
httpd -l
```

Si en la lista que aparece en pantalla se encuentra el módulo mod_auth_external.c, querrá decir que la instalación ha tenido éxito.

Si prefiere escribir su propio código en C, tendrá que dar los siguientes pasos (sálteselos si no tiene ningún interés en escribir sus propias funciones).

Paso 1: activar la función soporte de código del módulo

Edite el archivo mod_auth_external.c y quite el símbolo de comentario de las siguientes líneas de código.

```
#define _HARDCODE_
#ifdef HARDCODE
#include "su_función_va_aquí.c"
#endif
```

Sustituya "su_función_va_aquí.c" con el archivo cabecera de la función de autenticación. Por ejemplo, si el nombre de dicho archivo es my_auth.c y tiene el prototipo de la función en my_auth.h, tendrá que escribir las siguientes líneas de código en mod_auth_external.c_

```
#define _HARDCODE_
#ifdef HARDCODE
#include "my_auth.h"
#endif
```

Asegúrese de que su código fuente se encuentra en el mismo directorio que mod_auth_external.c.

Paso 2: programar su función en el módulo

Salte hasta la sección grande que aparece comentada en mitad del módulo mod_auth_external.c y siga las instrucciones que aparecen allí. Su función ha de comenzar con algo similar a:

```
int
function_name (char *user_name,char *user_passwd,char *config_path) {
The function call in mod_auth_external.c should look something like:

if (strcmp(check_type,"<type>")==0) {
            code    function_name(c->user,sent_pw,config_file);
}
```

Nota: no utilice exit( ) o cualquier otra llamada que pueda hacer que la función finalice de forma poco normal. Haría entonces que el httpd se bloquease con él y que el explorador Web mostrase un mensaje en el que se comunicase que no hay datos disponibles. Puede sustituir exit( ) por return( ).

Paso 3: guardar y compilar Apache

Ejecute la utilidad de construcción dentro del directorio fuente de Apache. Si no obtiene un código limpio, depúrelo. Siempre que efectúe cambios en el archivo de código y no modifique mod_auth_external.c, conviene que utilice la utilidad touch con mod_auth_external.c.

Una vez que se ha instalado mod_auth_external, dispondrá de tres mecanismos para acceder a las aplicaciones exteriores.

El primero es a través de la llamada system( ). Es el método predeterminado. En este caso el módulo llama al programa de autenticación de un usuario definido (en este caso, el administrador de Apache) y le pasa dos variables de entorno: USER y PASS. El cliente escribe su nombre y contraseña y las transmite a través de estas dos variables. El programa de autenticación externo las lee y efectúa las comprobaciones pertinentes para determinar si el cliente puede acceder o no al directorio solicitado.

El módulo necesita que el programa de autenticación externo le entregue un código 0, si es que la comprobación ha tenido éxito, o bien un 1 si no lo ha tenido.

Advertencia: en la mayoría de los sistemas UNIX es posible determinar qué variables de entorno se pueden utilizar para ejecutar procesos gracias a la utilidad ps. Por ejemplo:

```
ps -aeowe
```

Obtendrá el entorno de todos los programas que se están ejecutando en un sistema Linux. De esta forma cualquier usuario del sistema podrá comprobar qué variables USER y PASS se han utilizado durante la ejecución del programa autocheck. La verdad es que representa un problema para la seguridad del sistema.

El segundo método de autenticación externo lo tenemos en pipe( ). Es muy parecido al anterior, pero la forma en que el módulo escribe el nombre de usuario y la contraseña varía. Lo hace a través del STDIN del programa de autenticación externa. Los datos que se almacenan en el par clave=valor, donde USER= nombre de usuario y PASS=contraseña.

El tercer método de autenticación externa se obtiene a través de la codificación de una función de llamada. En este modo, el módulo llama a un usuario definido en la función y le entrega el nombre del usuario y contraseña, además del nombre del archivo de configuración. Este método permite que los desarrolladores puedan escribir gateways personalizadas o nuevas aplica-

ciones de autenticación. Ya hay ejemplos de este código, entre los cuales destacamos los siguientes:

- mod_auth_external_radius.c: el cliente Radius utiliza el código publicado por Merit Radius.
- mod_auth_external_sybase.c: una función que solicita una base de datos de sybase y compara las contraseñas del usuario.

AddExternalAuth

```
Sintaxis (versiones system y pipe): AddExternalAuth clave path/al/
comprobador
Sintaxis (versión function): AddExternalAuth tipo clave: path/a/
configuración
Contexto: configuración del servidor
```

Esta directriz asocia la clave con el autenticador que especifique el usuario. Por ejemplo:

```
AddExternalAuth archive_auth /usr/local/bin/authcheck
```

Con esta sentencia se asocia archive_auth con un programa de autenticación externo llamado /usr/local/bin/authcheck. Un ejemplo de esta directriz en la que se utilice una función codificada podría ser:

```
AddExternalAuth archive_auth RADIUS:
```

donde RADIUS es el nombre de la función codificada. Si esta función requiere un archivo de configuración se le tendrá que entregar de esta manera:

```
AddExternalAuth archive_auth RADIUS:/usr/local/raddb
```

A los programas de autenticación externos se les entrega el nombre del usuario, su contraseña y (opcionalmente) el nombre del archivo de configuración (sólo con las autenticaciones basadas en las funciones codificadas). Por defecto, el programa de autenticación externo recibe el nombre de usuario y la contraseña como variables de entorno USER y PASS, respectivamente. Si se especifica el modo pipe a través de la directriz AetExternalAuthMethod, el nombre de usuario y su contraseña se pasarán como clave=valor de USER= nombre de usuario y PASS=contraseña. Para las versiones en las que se trabaje con funciones codificadas, los datos se pasan como funciones en C.

Si el programa de autenticación externa finaliza con un código 0, la autenticación habrá tenido éxito. Cualquier valor distinto de 0 indicará que la autenticación no ha tenido éxito.

Recuerde que puede utilizar varios tipos de autenticación en un mismo servidor suministrando varias directrices a través del archivo srm.conf. Todo lo que tiene que hacer es estar seguro de que cada grupo tiene una clave diferente.

AddExternalGroupAuth

```
Sintaxis (versiones system y pipe): AddExternalGroupAuth clave
path/al/comprobador
Sintaxis (versión function): AddExternalGroupAuth tipo clave:
path/a/configuración
Contexto: configuración del servidor
```

Esta directriz asocia la clave con el autenticador del grupo especificado. Por ejemplo, la siguiente línea de código asocia archivo_auth con un programa de autenticación externo llamado /user/local/bin/groupcheck:

```
AddExternalGroupAuth archive_auth /usr/local/bin/groupcheck
```

A los programas de autenticación externa de grupos se les entrega el nombre del usuario, su contraseña y (opcionalmente) el nombre del archivo de configuración (sólo con las autenticaciones basadas en las funciones codificadas). Por defecto, el programa de autenticación externo recibe el nombre de usuario y la contraseña como variables de entorno USER y GROUP, respectivamente. Si se especifica el modo pipe a través de la directriz ActExternalAuthMethod, el nombre de usuario y su contraseña se pasarán entonces como clave=valor de USER=nombre de usuario y GROUP=grupo. Para las versiones en las que se trabaje con funciones codificadas, los datos se pasan como funciones en C.

Si el programa de autenticación externa finaliza con un código 0, la autenticación habrá tenido éxito. Cualquier valor distinto de 0 indicará que la autenticación no ha tenido éxito.

SetExternalAuthMethod

```
Sintaxis: SetExternalAuthMethod clave método
Contexto: configuración del servidor
```

Esta directriz determina el método que se va a utilizar para transferir datos al programa de autenticación externo. En la actualidad hay tres métodos disponibles.

El primero es environment. Aquí, el nombre del usuario y su contraseña se transmite a través del método environment. La variable USER contiene el nombre del usuario y PASS su contraseña. Es el método predeterminado.

Considere el siguiente ejemplo:

```
AddExternalAuth archive_auth /usr/local/bin/authcheck
SetExternalAuthMethod archive_auth environment
```

Aquí, el programa /usr/local/bin/authcheck recibirá el nombre y la contraseña que haya introducido el usuario a través de las variables USER y PASS.

El segundo método es function. Este método se utiliza con las funciones codificadas y se utiliza para controlar la autenticación.

Vamos a ver las siguientes líneas de código.

```
AddExternalAuth archive_auth /usr/local/bin/authcheck
SetExternalAuthMethod archive_auth pipe
```

En este caso, el programa /usr/local/bin/authcheck recibirá el nombre y la contraseña que haya introducido el usuario a través de pipe y que se envían al dispositivo de entrada (STDIN) del autenticador externo, donde USER= nombre de usuario y PASS=contraseña.

El tercer método es pipe. Le envía los datos a través de pipe al STDIN del autenticador externo. Los datos se transmiten en forma de clave=valor, donde USER= nombre de usuario y PASS=contraseña.

Vea el siguiente ejemplo:

```
AddExternalAuth archive_auth RADIUS:
SetExternalAuthMethod archive_auth function
```

Aquí se utiliza la función codificada RADIUS, que recibe los datos como parámetros de función.

SetExternalGroupMethod

```
Sintaxis: SetExternalGroupMethod clave método
Contexto: configuración del servidor
```

Esta directriz determina el método que se utiliza para transmitir datos a un programa externo de autenticación de grupos. En la actualidad hay tres métodos disponibles: environment, function y pipe. Véase SetExternalAuthMethod para más detalles.

AuthExternal

```
Sintaxis (versiones call y pipe): AddExternalAuth clave
Sintaxis (versión function): AddExternalAuth nombre función:
path/archivo de configuración
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
```

Esta directriz se utiliza en el contexto del directorio o en el archivo de control de acceso de cada directorio para indicar qué programa de autenticación externa se ha de utilizar. El servidor especifica la clave o el nombre de la función (en caso de que se trabaje con una función codificada) que determinará qué se tiene que hacer.

Por ejemplo:

```
AuthExternal archive_auth
```

Si aparece esta línea en el archivo .htaccess, Apache ejecutará el autenticador de usuarios asociado con dicha clave.

GroupExternal

```
Sintaxis (versiones call y pipe): GroupExternal clave
Sintaxis (versión function): GroupExternal nombre función:
path/archivo de configuración
Contexto: directorio, archivo de control de acceso por directorio
(.htaccess)
```

Esta directriz se utiliza en el contexto del directorio o en el archivo de control de acceso de cada directorio para indicar qué programa externo de autenticación para grupos se ha de utilizar. El servidor especifica la clave o el nombre de la función (en caso de que se trabaje con una función codificada) que determinará qué se tiene que hacer.

Por ejemplo, si aparece la línea de código que le mostramos a continuación en el archivo .htaccess, Apache ejecutará el autenticador de grupos asociado con dicha clave.

```
GroupExternal archive_auth
```

Ejemplo: uso de un script en Perl como autenticador externo

En este ejemplo, verá cómo se puede utilizar un sencillo script en Perl para suministrar el servicio de autenticación. Para que el ejemplo sea algo más interesante, hemos utilizado un script en Perl que utiliza los archivos /etc/passwd y /etc/group, que se encuentran en la mayoría de los sistemas UNIX, para efectuar la autenticación.

Muchos me dirán que no es muy conveniente utilizar estos dos archivos y la verdad es que estoy completamente de acuerdo. De todas formas, en una intranet bien protegida (tanto mejor si no está conectada a Internet), no parece tan malo. Cualquiera que tenga una cuenta UNIX en un servidor Web puede efectuar procesos de autenticación utilizando el script en Perl que aparece en el listado 10.2.

```
#!/usr/local/bin/perl
#
# Nombre del script: authcheck
# Propósito: un sencillo script de autenticación basado en los
# archivos /etc/passwd y /etc/group
#
############################################################

#  Variables y asignaciones
my $LOG_FILE = '/tmp/authcheck.log';    # archivo de registro a usar
my $this_user = $ENV{USER};       # toma el nombre de usuario de %ENV
my $this_pwd = $ENV{PASS};        # toma la contraseña de %ENV

# Si el usuario no introduce ni su nombre ni la contraseña, se
# finaliza con un valor distinto de cero. De esta forma
# mod_auth_external sabrá que la autenticación habrá fallado.
#
exit 1 if($this_user eq "" || $this_pwd eq "");

# Tan sólo queremos que los usuarios de cierto grupo puedan efectuar
# el proceso de autenticación a través de la red. Cuando cree el
# grupo en el directorio /etc/group, recuerde mantener a los
# posibles usuarios fuera de él. Por ejemplo, NUNCA coloque un
# usuario 'root' dentro de este grupo.
#
my $WEB_USER_GROUP = 'web-users';

# Toma nota de la fecha y la hora y la guarda en las variables
# correspondientes
my ($sec,$min, $hr, $mday, $mon, $year, $wday, $yday, $isdst) =
localtime(time);

# Incrementa el mes teniendo en cuenta que el rango de valores que
# se utilizan en localtime es 0-11
$mon++;

# Si el usuario es un miembro de un grupo de usuarios de la Red, y
# su contraseña es válida, habrá que dar el proceso de
# autenticación por válido.
#
if(isMember($this_user,$WEB_USER_GROUP) &&
validPassword($this_user,$this_pwd)){

   # Registra que la autenticación ha tenido éxito. De todas formas,
   # como cada una de las peticiones que efectúa un usuario pueden
   # pasar a través de este código de autenticación, procesar la
   # información de registro puede llegar a dar un problema de
   # rendimiento. Si quiere evitarlo, añada el símbolo de
   # comentario a la siguiente línea.
   #
    &log("$mon/$mday/$year $hr:$min:$sec - $this_user login
successful.");

   # Finaliza con el valor del código de éxito para que
   # mod_auth_external sepa que la autenticación ha sido
```

```
    # satisfactoria.
    exit 0;
    }
else{
      # El usuario no ha pasado la autenticación en este intento.
      &log("$mon/$mday/$year $hr:$min:$sec - $this_user login failed:
invalid password ($this_pwd) .");

     # Finaliza con un código distinto de cero para que
     # mod_auth_external sepa que la autenticación no ha sido
     # satisfactoria.
   exit 1;
   }

sub validPassword{
    #
    # Propósito: esta subrutina compara la contraseña introducida por
    # el usuario con la que tiene guardada en el archivo /etc/passwd.

   # Toma el nombre y la contraseña introducidos por el usuario de la
   # lista de parámetros de la subrutina.
    my $this_user = shift;
    my $guess = shift;

   # Toma la verdadera contraseña del usuario que se encuentra
   # almacenada en /etc/passwd.
    my (@pwdfields) = getpwnam($this_user);

   # Utiliza crypt() para encriptar la contraseña introducida por el
   # usuario y compararla con la que tiene guardada en /etc/passwd.

    if(crypt($guess, $pwdfields[1]) eq $pwdfields[1]) {

# El usuario ha introducido la contraseña correcta, por lo que se le
# devuelve un valor distinto de cero para invocar a la rutina.
   return 1;
   }
   # El usuario ha introducido una contraseña incorrecta. Se devuelve
   # el valor 0.
    return 0;
    }

sub isMember{
    #
    # Propósito: esta subrutina comprueba el archivo /etc/group para
    # determinar si el usuario pertenece a un grupo determinado.
    #

    # Toma el nombre del usuario y el grupo de la lista de parámetros.
    my $this_user = shift;
    my $web_group = shift;

   # Toma la lista de miembros del archivo /etc/group
    my ($groupName, $passwd, $gid, $memberList) =
getgrnam($web_group);
```

```
    # Si el usuario pertenece a este grupo, se devuelve un valor
    # distinto de cero o un cero.
     return 1 if ($memberList  =~ /$this_user/);
     return 0;
     }

sub log{
    #
    # Propósito: proporcionar un registro
    #

   # Toma la entrada del registro de la lista de parámetros.
    my $entry = shift;

   # Agrega la entrada a la lista de registros.
    open(FP,">>$LOG_FILE") || die "Can't open $LOG_FILE file";
    print FP $entry, "\n";
    close(FP);
            }
```

Listado 10.2. Script authcheck

Este script toma el nombre del usuario de la variable de entorno USER y la contraseña de PASS. A continuación comprueba si el usuario pertenece a un grupo determinado (en este caso a un grupo de usuarios de la Red), definido en /etc/group. En caso afirmativo, comprueba la validez de la contraseña. Si se cumplen ambas condiciones, el script devuelve un valor 0 o también un 1. De esta forma, el script cumple a la perfección las necesidades del módulo mod_auth_external relacionadas con el programa de autenticación externo. Se ha añadido la comprobación de pertenencia a un grupo de usuarios para asegurarse de que los únicos que pueden acceder al directorio protegido son los usuarios de un grupo determinado. Si utiliza este script, asegúrese de que nadie agrega usuarios potenciales (como 'root') a este grupo. Vamos a ver cómo se puede utilizar el script con el módulo mod_auth_external.

Paso 1: asociar el autenticador authcheck con una clave.

Agregue la siguiente línea de código en el archivo srm.conf:

```
AddExternalAuth myauth /usr/local/bin/authcheck
```

Le indica a Apache que la clave myauth está asociado con el script en Perl authcheck.

Paso 2: crear la configuración de acceso al directorio

Para restringir el acceso al directorio utilizando un autenticador externo, tendrá que agregar la siguiente línea de código en el archivo .htaccess:

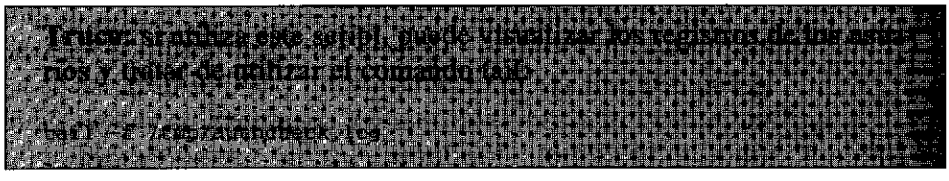
```
AuthType Basic
AuthName Access for Everyone
AuthExternal myauth
SetExternalAuthMethod myauth environment
require valid-user
```

La directriz AuthExternal le dice a Apache que el autenticador para este directorio se identifica a través de la clave myauth (que hace referencia al script authcheck). El método SetExternalAuthMethod le dice a Apache que utilice la variable de entorno para pasar los datos al autenticador asociado a myauth.

Por último, la línea require le indica a Apache que permita el acceso a todos los usuarios que pasen el proceso de autenticación.

Paso 3: prueba

Intentar acceder al directorio protegido hace que aparezca el cuadro de diálogo emergente de autenticación. Al introducir el par nombre/contraseña que pertenece al grupo de usuario de la Red, se permitirá el acceso.



Autenticación digest

Como ha podido ver, la autenticación HTTP básica no es segura, por lo que nunca se ha de utilizar con aplicaciones importantes. Hay otros métodos mejores, pero su implementación no depende de los diseñadores de los servidores, sino de los encargados de los exploradores. Uno de los métodos que parece que va a sustituir a la HTTP básica es la autenticación digest. La extensión de este método dependerá de la cantidad de desarrolladores de exploradores Web que lo incorporen en sus productos. Apache puede trabajar con digest gracias al módulo mod_digest.

El problema con la autenticación básica es que las contraseñas que se transmiten por la Red tan sólo están uu-codificadas, con lo que el proceso no será nada seguro. En la autenticación digest, las contraseñas nunca se transmiten por el cable. En lugar de eso, se genera una serie de valores numéricos basados en los datos que introduce el usuario y en otra información relacionada con la petición. Estos números se transforman utilizando un método de

encriptación llamado MD5. El resultado recibe el nombre de digest y se envía a través de la red. Se combina con otros elementos del servidor para verificar su autenticidad. Si no se encuentra ninguna coincidencia, se puede aceptar o no al usuario.

Aunque el método de autenticación digest es más seguro que la autenticación básica HTTP, es necesario que el servidor sea muy seguro porque tiene que almacenar todos los digest para efectuar las comparaciones pertinentes.

11 Registro y estado del servidor

Llegados a este punto, espero que tenga configurado y preparado su servidor Apache y que el resto del mundo tenga noticias de su existencia. Entonces, ¿qué viene a continuación? ¿Se ha preguntado alguna vez quién accede a su sitio Web? Si se parece en algo al resto del mundo, es probable que sí. También es posible que se pregunte cómo interactúa el servidor con el sistema. En este capítulo le mostraré como utilizar técnicas de registro y visualización con las que satisfacer sus ansias de conocimiento.

Visualizar Apache

Apache le permite visualizar dos tipos de información a través de la Red. Son los siguientes:

- Información sobre la configuración del servidor.
- Estado del servidor.

La información sobre la configuración del servidor es estática. De todas formas, siempre es conveniente disponer de un acceso rápido a la información del servidor. Por otro lado, la información sobre el estado del servidor es

dinámica. Gracias a las propiedades de monitorización del servidor se puede acceder a información como las actualizaciones, el número total de peticiones atendidas, el total de datos transferidos, el estado de los subprocesos y los recursos del sistema utilizados. Ambos tipos de información tienen un gran valor, por eso los trataremos en las siguientes secciones. Vamos a empezar con la información de la configuración del monitor.

Acceso a la información de configuración

Se puede hacer con mod_info. Este módulo proporciona una visión general de la configuración del servidor bastante amplia, incluyendo los módulos y directrices instalados en los archivos de configuración. Este módulo se encuentra dentro del fichero mod_info.c. Por defecto no se compila dentro del servidor. Tendrá que compilarlo siguiendo el proceso usual que consiste en añadir el módulo al archivo Configuration y después utilizar el script Configure y la utilidad make. Dentro del módulo mod_info nos encontramos con la directriz AddModuleInfo:

```
Sintaxis: AddModuleInfo nombre-módulo texto descriptivo
Contexto: configuración del servidor, host virtual
```

Esta directriz le permite añadir un texto descriptivo en la lista de módulos que proporciona mod_info. El texto descriptivo puede ser cualquier cosa, hasta un texto HTML. Por ejemplo:

```
AddModuleInfo mod_auth.c 'See <A HREF="http://www.apache.org/docs/
mod/mod_auth.html">http://www.apache.org/docs/mod/mod_auth.html</A>'
```

Con este código verá un enlace HTML que aparecerá junto a la lista de mod_auth.c. Este enlace es un sistema que le permitirá acceder a más información de la disponible en el módulo que se encuentra en el sitio Web de Apache. Para ver la información sobre la configuración del servidor a través de la Red, tendrá que añadir las siguientes líneas al archivo access.conf:

```
<location /server-info>
SetHandler server-info
</Location>
```

Puede darse el caso de que se desee añadir la cláusula <Limit> dentro de la directriz location para limitar el acceso a la información sobre la configuración del servidor. Cuando se completa el proceso de configuración, se accederá a la información sobre el servidor a través de:

```
http://your.host.dom/server-info
```

Obtendrá una página con toda la información del servidor y todos los módulos. Para acceder únicamente a la información del servidor, utilice:

```
http://your.host.dom/server-info?server
```

Y para hacerse con la configuración de un módulo, use:

```
http://your.host.dom/server-info?nombre_módulo
```

Para obtener una lista rápida con los módulos incluidos, emplee:

```
http://your.host.dom/server-info?list
```

Vamos a ver cómo se puede visualizar el estado del servidor Apache que se está ejecutando.

Activar las páginas de estado

El módulo mod_status permite que los administradores Apache visualicen el servidor desde la Red. Se crea una página HTML con las estadísticas del servidor. También se genera otra página legible en código máquina (que veremos en este mismo capítulo). La información que aparece en ambas páginas es la siguiente:

- Hora actualizada del sistema del servidor.
- Hora del último reinicio del servidor.
- Tiempo que ha pasado desde que se encendió el servidor.
- Número total de accesos que han tenido lugar desde entonces.
- Bytes totales transmitidos.
- Número total de los thread dedicados a atender peticiones.
- Número total de thread.
- Estado de cada thread, número de peticiones que ha atendido cada uno y número total de bytes servidos por cada uno de ellos.
- Promedios con el el número de peticiones efectuadas por segundo, el número de bytes servidos por segundo y los bytes por segundo.
- Porcentaje actual de CPU utilizada por cada thread y el total que utiliza Apache.
- Host y peticiones procesadas.

Nota: si en la página que le muestra su sistema no aparecen todos estos campos, tendrá que utilizar la opción -DSTATUS a AUX_CFLAGS que se encuentra en el archivo Configuration. Luego tendrá que volver a configurarlo y rehacer Apache. Observe que en algunas máquinas puede llegar a haber una pequeña pérdida de eficacia.

En la distribución estándar de Apache no se compila este módulo por defecto, por lo que tendrá que añadirlo al archivo Configuration, ejecutar el script Configure y utilizar la utilidad make para construir un nuevo ejecutable de Apache. No se olvide de utilizar httpd -l para comprobar que el módulo mod_status forma parte del ejecutable.

Visualización de las páginas de estado

Una vez que ha compilado el módulo mod_status y construido su nuevo ejecutable del servidor Apache, tendrá que definir un URL que utilizará para mostrar la información. En otras palabras, le tendrá que indicar a Apache que URL será el encargado de presentarle las estadísticas del servidor a su explorador Web.

Supongamos que su nombre de dominio es sudominio.com y que quiere utilizar el siguiente URL:

```
www.mydomain.com/apache-status
```

Utilizando el contenedor <Location ...> le puede indicar al servidor que desea controlar este URL por medio de server-status, que se encuentra en el módulo mod_status. El siguiente código le ayudará a hacer el trabajo.

```
<Location /apache-status>
SetHandler server-status
</Location>
```

Aquí, la directriz SetHandler determina el controlador (server-status) para URL mencionado anteriormente. Este segmento de configuración debería aparecer en el archivo access.conf, pero la verdad es que se puede encontrar en cualquiera de los tres archivos de configuración. Una vez que añada estas líneas a uno de estos ficheros, puede reiniciar el servidor y acceder al URL anterior desde cualquier explorador Web. Un ejemplo de una de estas páginas podría ser la que aparece en la figura 11.1.

El contenedor <Location ...> que aparece en la figura 11.1 permite que cualquiera que lo desee pueda ver el estado del servidor accediendo a este URL. La verdad es que, desde el punto de vista de la seguridad, no es buena

idea. Para asegurarse de que tan sólo las máquinas de su dominio pueden acceder a la página de estado, tendría que cambiar la configuración anterior por:

```
<Location /apache status>
SetHandler server-status
order deny, allow
deny from all
allow from .yourdomain.com
</Location>
```

Aquí, yourdomain.com tendrá que reemplazarse por el nombre de su dominio. Si únicamente quiere que sean unos cuantos host los elegidos para acceder a esta página, tendrá que utilizar una directriz allow con una lista con los nombres de dichos sistemas.

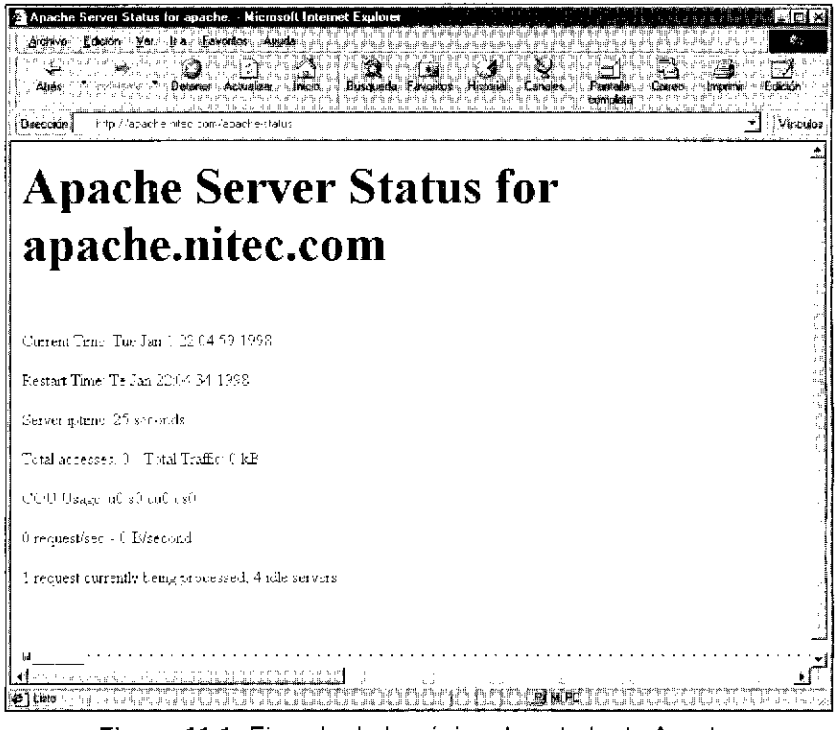


Figura 11.1. Ejemplo de la página de estado de Apache

Truco: si tiene un explorador que pueda trabajar con el comando refresh, puede hacer que la página de estado se actualice constantemente. Para ello tendrá que acceder a la página http://www.sudominio.com/server-status?refresh=N para actualizar la página cada N segundos.

Simplificar la página de estado

La página de estado que genera el módulo muestra información extra, inservible para algunos programas de análisis. Por ejemplo, si quiere crear un gráfico de los datos de su servidor por medio de una hoja de cálculo, tendrá que depurar los datos a mano. De todas formas, el módulo dispone de un sistema que le permitirá crear salidas en código máquina desde el mismo URL.

Para simplificar la pantalla de estado, añada ?auto al final del URL. Esta cadena le indica a Apache que simplifique los datos que aparecen en pantalla, tal y como se puede ver en la figura 11.2.

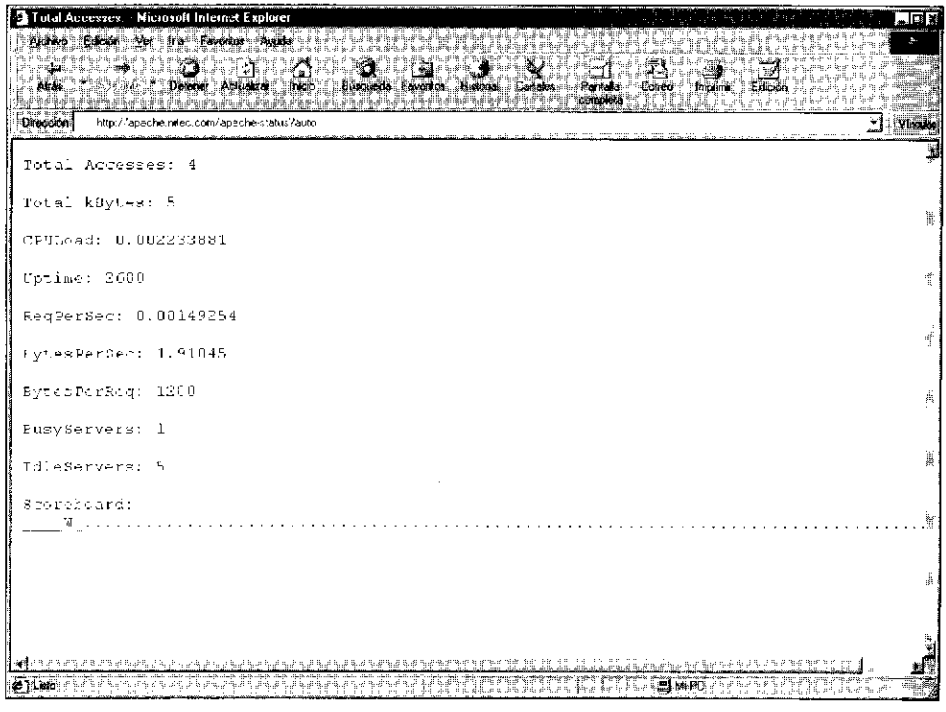


Figura 11.2. Salida sobre el estado generada en código máquina

Guardar la información sobre el estado del servidor

Apache incluye un script en Perl (que se encuentra en el directorio support de los directorios fuente) que se llama log_server_status y que se puede utilizar periódicamente para guardar la información sobre el estado del servidor (utilizando la cadena ?auto) en un archivo de texto plano.

Puede ejecutar este script como un trabajo programado que se encargue de guardar la información cada cierto tiempo. De todas formas, antes de que

pueda utilizar este script, tendrá que editar su código fuente para modificar los valores de las siguientes variables:

```
$wherelog
$port
$server
$request
```

El nuevo valor de $wherelog tiene que ser el del path de la ubicación en la que desee guardar el archivo que cree el script. El valor de la variable $port tiene que ser el número del puerto del servidor que quiere visualizar. Se suele utilizar el puerto 80 porque es el número del puerto HTTP estándar. $server será el nombre del servidor. Si el script se ejecuta en la misma máquina en la que se encuentra el servidor, se puede usar el valor localhost. Pero si el servidor se encuentra en otra máquina, habrá que especificar su nombre completo (por ejemplo, www.midominio.com). La variable $request debería tener el mismo valor que lo que se utilice en la directriz <Location ...>, además de la cadena ?auto.

Si no le gusta el formato que utiliza el módulo para guardar lo generado por el script, puede modificar la siguiente línea de código:

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

Nota: el script utiliza una conexión con el servidor Apache a través de la cual envía la petición URL. Por lo tanto, ha de comprobar que tiene soporte para la conexión para que trabaje con Perl.

Creación de los archivos de registro

Conocer la información sobre el estado y la configuración del servidor es muy útil a la hora de administrar el servidor. Pero también es muy importante saber qué o quién está accediendo en cada momento a su sitio Web. En las siguientes secciones veremos cómo funciona el registro y como aprovechar al máximo los módulos de registro de Apache.

La aparición del primer software para servidores Web en el mercado estuvo acompañada del surgimiento de programas de análisis de registros. Estos programas formaron parte del trabajo diario de los administradores Web. Y de entonces data la era de la incompatibilidad de datos, que hizo que el análisis de los archivos de registro fuera una tarea ardua y difícil. Un mismo

programa de análisis no trabaja con todos los formatos. Por eso apareció la especificación CLF (Common Log Format o formato común de registro). Permitió que los servidores Web escribiesen registros de una manera muy parecida, por lo que se facilitó el análisis de estos datos.

Por defecto, en la distribución estándar de Apache se incluye el módulo mod_log_config, que es el que se encarga de los registros básicos y los escribe en un archivo en formato CLF. Con la directriz LogFormat puede modificar este comportamiento. De todas formas, el formato CLF suele ser más que suficiente para la mayoría de los entornos. A continuación le explicamos el contenido de cada línea.

El archivo CLF contiene una línea separada para cada petición. Cada línea se compone de varios elementos separados por espacios.

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

Si uno de estos elementos carece de valor, entonces se representa por un guión. El significado de cada uno de ellos es el siguiente:

- Host. Se trata del nombre completo del dominio del cliente o su dirección IP.
- Ident. Si la directriz IdentityCheck está activada y la máquina del cliente ejecuta identd, entonces ésta será la información de identidad que se le pasará al cliente.
- Authuser. Si el URL solicitado necesita una autenticación HTTP básica, se tomará como valor de este elemento el nombre del usuario.
- Date. Fecha y hora de la petición.
- Request. La línea que solicita el cliente, entre dobles comillas (").
- Status. Código de estado de tres dígitos que se le entrega al cliente.
- bytes. Número de bytes del objeto que se le entrega al cliente, sin contar las cabeceras HTTP.

Los campos de la fecha y la hora pueden tener el siguiente formato:

```
date = [día/mes/año:hora:minuto:segundos zona]
```

Por ejemplo:

```
[02/Jan/1998:00:22:01 -0800]
```

Los tamaños de los campos aparecen en la tabla 11.1

Tabla 11.1. Tamaños de las fechas

Campos	Valor
Día	2 dígitos
Mes	3 letras
Año	4 dígitos
Hora	2 dígitos
Minuto	2 dígitos
Segundo	2 dígitos
Zona	(`+´|`- ´) 4 * dígitos

Vamos a ver las cuatro directrices que se pueden usar con mod_log_config.

TransferLog

```
Sintaxis: TransferLog nombre_archivo | "  /path/a/programa/externo"
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz determina el nombre del archivo de registro o del programa al que se le envía la información. Por defecto, la información registrada tiene formato CLF. Dicho formato se puede personalizar a través de la directriz LogFormat (la veremos más tarde).

Cuando aparece una directriz TransferLog en el contenedor de un servidor virtual, el formato de la información de registro lo aplica TransferLog. Pero si no estuviese esta directriz en el contenedor, se utilizaría el formato de registro propio del servidor.

TransferLog toma el path o la dirección de un programa externo como argumento. Si no aparece una barra inclinada al principio del argumento, el nombre del archivo de registro será relativo a ServerRoot. Por ejemplo, si el valor de ServerRoot es /etc/httpd, entonces la siguiente línea de código le indicará a Apache que envíe información al archivo /etc/httpd/logs/access.log:

```
TransferLog logs/access_log
```

Cuando el argumento es la dirección de un programa externo, la información de registro se envía a la estrada estándar (STDIN) del programa.

Obsérvese que si el servidor principal le pasa una directriz TransferLog a un servidor virtual (VirtualHost), no se abrirá ningún programa. Si se ejecuta

un programa, pertenecerá al usuario que inició httpd. Si es el usuario root quien inicia el servidor, será quien tenga su posición. Compruebe que el programa es seguro.

LogFormat

```
Sintaxis: LogFormat formato [alias]
Predeterminado: LogFormat "%h %l %u %t \"%r\" %s %b"
Contexto: configuración servidor, host virtual
```

Esta directriz determina el formato que se usará al guardar la información en el archivo especificado en TransferLog. Si incluye un alias para el formato en la misma línea de la directriz, podrá utilizarlo en LogFormat y CustomLog, con lo que se evitará tener que repetir toda la cadena de formato. Una directriz LogFormat que define un alias no implica nada más, es decir, que tan sólo hace eso, definir un alias, no aplicar su formato.

CustomLog

```
Sintaxis: CustomLog archivo-pipe formato-o-alias
Contexto: configuración servidor, host virtual
```

Al igual que ocurría con TransferLog, esta directriz sirve para enviar información al archivo de registro o a un programa externo. Pero, a diferencia de ella, permite el uso de un formato personalizado que se tendrá que especificar como argumento.

El formato del argumento especifica el formato de cada línea del archivo de registro. Las opciones disponibles para el formato son exactamente las mismas para la directriz LogFormat. Si aparece algún espacio en blanco dentro del formato (algo bastante común) debería aparecer entre comillas.

En vez de una cadena de formato actual, puede utilizar un alias que defina con la directriz LogFormat.

Nota: los alias se pueden utilizar a partir de la versión 1.3 de Apache. Recuerde que las directrices TransferLog y CustomLog se pueden utilizar todas las veces que se desee para que cada petición se registre en varios archivos.

CookieLog

```
Sintaxis: CookieLog nombre archivo
Contexto: configuración servidor, host virtual
```

CookieLog determina el nombre del archivo en el que se registrará la información sobre las cookies. Este nombre tomará como referencia ServerRoot. Esta directriz únicamente se incluye para conservar la compatibilidad con mod_cookies, pero la verdad es que está camino de la extinción (por eso no podemos recomendar su utilización). Para sustituirla se utiliza el módulo mod_usertrack.

Personalizar sus archivos de registro

Aunque CLF cumple la mayoría de los requisitos, a veces es conveniente personalizar la información de registro. Por ejemplo, es posible que desee tomar nota del tipo de exploradores que utilizan los usuarios que acceden a su sitio Web, para que su equipo de desarrollo pueda determinar para qué tipo de servidor conviene optimizar el sitio Web o cuáles ignorar. O quizá quiera saber qué sitios Web están remitiéndole visitantes. Con un servidor Apache todo esto es muy sencillo. El módulo de registro predeterminado, mod_log_config, soporta la personalización de esta información.

Los formatos personalizados se determinan con las directrices LogFormat y CustomFormat del módulo, El argumento del formato de cualquiera de ambas directrices será una cadena que puede tener tanto caracteres literales como especificadores de formato (%). Cuando se utilizan valores literales, con cada petición se copian en el archivo de registro. Sin embargo, los especificadores se sustituyen por su correspondiente valor. Los especificadores % especiales son:

%b	Bytes enviados, sin contar las cabeceras HTTP.
%f	Nombre del archivo solicitado.
%{variable}e	Contenido de la variable de entorno VARIABLE.
%h	Servidor remoto que efectúa la petición.
%{ IncomingHeader }i	Contenido de IncomingHeader, es decir, las líneas de las cabeceras de la petición que se le envía al servidor. El carácter "i" que aparece al final nos indica que la cabecera proviene de un cliente.
%I	Si se activa la directriz IdentityCheck y la máquina del cliente ejecuta identd, entonces será la información que aporta el cliente.

%{ Foobar }n	Contenido de Foobar.
%{ OutgoingHeader }o	Contenido de OutgoingHeader, es decir, las líneas de la cabecera de la respuesta. El carácter "o" que se encuentra al final indica que la cabecera la ha generado el servidor.
%p	Puerto por el cual se atiende la petición.
%p	ID del proceso del thread que atiende a la petición.
%r	Primera línea de la petición.
%s	Estado que devuelve el servidor en respuesta a la petición. Obsérvese que cuando se redirecciona la petición, el valor de este especificador de formato aún es el original. Si desea guardar el estado de la petición que se ha vuelto a enviar, tendrá que utilizar %...>s.
%t	Tiempo de petición. El formato del tiempo es CLF.
%{format}t	Hora. El formato determina su apariencia.
%t	Tiempo empleado en atender la petición (en segundos).
%u	Si el URL solicitado necesita una autenticación HTTP básica, el valor de este especificador será el nombre del usuario. Si el servidor devuelve un estado 401 (hace falta autenticación) se simulará su valor.
%u	Path del URL solicitado.
%v	Nombre del servidor o del host virtual que envía la petición.

Se puede incluir información condicional en cada especificador. La presencia (o ausencia) de los códigos de estado HTTP puede determinar las condiciones. Por ejemplo, supongamos que quiere registrar todo lo relacionado con los URL que apunten a una página de un usuario no existente. En este caso, el servidor producirá una cabecera de estado 404 (no encontrado). Luego, para registrar todo lo referido a este URL, tendrá que utilizar el siguiente código:

```
'%404{Referer'
```

De forma similar, para registrar todo lo relacionado con un URL que produzca un estado poco usual, puede utilizar:

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

Por ejemplo:

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

Esta especificación aplica el formato CLF a los datos del registro y añade la información Referer y User-agent que encuentre en las cabeceras que suministra el cliente a cada entrada del registro.

Ya ha visto cómo añadir datos que no sean CLF a un archivo de registro. Pero, ¿cómo almacenar los datos que se encuentran en más de un archivo de registro? En la siguiente sección veremos cómo trabajar con varios ficheros para almacenar datos CLF y no CLF.

Crear varios archivos de registro

A veces es necesario crear varios archivos de registro. Por ejemplo, si utiliza un programa de análisis que no pueda trabajar con datos que no sean CLF, es posible que prefiera escribirlos en un archivo diferente. Gracias a las directrices TransferLog y CustomLog, pertenecientes al módulo mod_log_config, puede crear varios archivos de registro. Todo lo que tiene que hacer es repetir estas directrices para crear más de un fichero. Si, por ejemplo, quiere crear un registro de acceso CLF estándar y otro personalizado para todos los URL, tendrá que utilizar unas líneas de código similares a éstas:

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

Cuando define en la configuración del servidor principal cualquiera de las directrices TransferLog y CustomLog y también ha definido un host virtual, estas directrices se encargarán de los registros relacionados con este host. Por ejemplo:

```
TransferLog logs/access_log
CustomLog   logs/agents_log      "%{User-agent}i"
<Virtual Host 206.171.50.51>
ServerName reboot.nitec.com
DocumentRoot  /www/reboot/public/htdocs
ScriptAlias /cgi-bin/ /www/reboot/public/cgi-bin/
</VirtualHost>
```

Aquí, el host virtual reboot.nitec.com no ha definido ninguna de las directrices TransferLog o CustomLog dentro de las etiquetas de su contenedor. Toda la información de registro se almacenará en log/access_log y logs/agents_log. Ahora bien, si se añade la siguiente línea en el contenedor del servidor virtual:

```
TransferLog vhost_logs/reboot_access_log
```

entonces todos los registros de reboot.nitec.com se efectuarán en el archivo vhost_logs/reboot_access_log y no se utilizarán ni logs/access_log ni logs/agents_log.

Guardar información específica

Esta sección le presenta unos cuantos módulos específicos de las opciones de registro que proporcionan una serie de propiedades a disposición del módulo de registro por omisión (mod_log_config). La única razón por la que se ofrece esta información es por motivos de compatibilidad.

Guardar la información del agente del usuario

Apache dispone de otro módulo llamado mod_log_agent que le permite registrar la información del agente del usuario (exploradores Web, robots) en un archivo independiente. Este módulo no se compila por defecto en la distribución estándar de Apache, por lo que tendrá que añadirlo al ejecutable httpd por el sistema habitual. El módulo tan sólo dispone de una directriz, AgentLog.

```
Sintaxis: AgentLog nombre_archivo | "  /path/a/programa/externo"
Predeterminado: AgentLog logs/agent_log
Contexto: configuración servidor, host virtual
```

Esta directriz le indica a Apache el nombre del archivo o programa que se tiene que utilizar para enviar la cabecera UserAgent de las peticiones que se reciban. Al igual que ocurre con la directriz TransferLog, puede introducir el nombre del archivo de registro o el path de un programa.

Obsérvese que cuando se utiliza CustomLog, puede añadir el especificador de formato %{User-agent}i para tomar nota de la cabecera UserAgent sin tener que usar este módulo, que no proporciona ninguna funcionalidad que no tenga mod_log_config. Si este módulo tuviese alguna forma de ignorar los accesos locales que efectúa su gente a través de la red, la información que generase sería más importante.

A continuación vamos a ver cómo se puede utilizar este módulo para guardar la información relacionada con los remitentes en un archivo de registro.

Guardar la información sobre los remitentes

Saber quién remite un visitante a su sitio Web es muy útil porque le ayuda a hacerse una idea de qué amigos tiene en la Red. También le puede ayudar a hacerse una idea sobre dónde invertir su dinero a la hora de anunciarse.

Poder precisar quién envía una petición a su sitio Web puede ser una gran ventaja a la hora de establecer relaciones con el resto de habitantes de la Red. Si recibe muchas visitas provenientes de un mismo sistema, conviene que sea cortés y establezca un vínculo en su sitio Web que conduzca a las páginas de dicho sistema. De todas formas, recuerde que antes de nada tiene que pedir permiso. Por otro lado, si recibe muchas visitas provenientes de un sistema que le trae sin cuidado o con el que no quiere colaborar, gracias a sus archivos de registro determinará qué sistemas son esos y podrá eliminar cualquier referencia a ellos que aparezcan en sus páginas Web.

El módulo mod_log_referer le permite registrar las cabeceras Referer de las peticiones recibidas. Por supuesto, la directriz CustomLog (perteneciente al módulo mod_log_config) le permite hacer lo mismo utilizando el especificador %{Referer}I. Entonces, ¿qué ventaja representa utilizarlo? La verdad es que, a diferencia de lo que ocurría con mod_log_config, con este módulo podrá ignorar ciertos host. De esta forma podrá registrar únicamente usuarios reales y no referencias a su host.

Tendrá que compilar este módulo dentro de su ejecutable de Apache. A partir de entonces dispondrá de las siguientes directrices.

RefererLog

```
Sintaxis: RefererLog nombre_archivo | "| /path/a/programa/externo"
Predeterminado: RefererLog logs/referer_log
Contexto: configuración servidor, host virtual
```

Esta directriz determina el nombre del archivo de registro o del programa que recibirá las cabeceras Referer que acompañan a las peticiones.

Si desea más información sobre las restricciones propias del path y sobre cómo utilizar un programa externo, consulte los detalles que aparecen en TransferLog.

RefererIgnore

```
Sintaxis: RefererIgnore cadena cadena ...
Contexto: configuración servidor, host virtual
```

La directriz RefererIgnore añade una lista de cadenas en las que se indican las cabeceras Referer que se han de ignorar. Si una cabecera Referer contiene cualquiera de las cadenas que aparecen en esta lista, no se registrará ninguna información al respecto. Por ejemplo:

```
RefererIgnore suempresa.com
```

De esta forma se evita cualquier referencia a suempresa.com. Si suempresa.com es el nombre del dominio de la red de trabajo interna de su compañía, ignorará todas las referencias que se generen desde ella.

El formato del archivo de registro es muy sencillo. Contiene una línea independiente para cara referencia. Cada una tiene el siguiente formato:

```
uri -> documento
```

Aquí, uri es (%-escaped) el URI del documento solicitado por el cliente y document es (%-decoded) el URL local de dicho documento.

Almacenaje de cookies

La verdad es que hasta este momento hemos visto muy pocas opciones relacionadas con las opciones de registro. Ninguna de ellas le permite identificar a sus visitantes. Y es algo muy importante porque si sabe qué pide cada uno de sus visitantes, podrá hacerse una idea de qué contenido de su sitio Web prefieren sus usuarios. Por ejemplo, supongamos que una de sus páginas Web es realmente atractiva y que dispone de un sistema gracias al cual puede identificar a los usuarios que visitan su sitio Web. Si a la hora de analizar los registros ve que sus visitantes han tenido que saltar de una página a otra hasta que al final han llegado al documento HTML más atractivo de todos, podría plantearse volver a diseñar el sitio Web para que sus visitantes dedicasen menos tiempo a alcanzar dicha página. Apache dispone de un módulo llamado mod_usertrack que le permite registrar los movimientos de sus visitantes gracias a las cookies HTTP.

Cookies HTTP

Una cookie HTTP no tiene nada que ver con las galletas (cookie es galleta en inglés). No es más que un extracto de información que se le entrega al explorador Web. Esta información se suele guardar en un par clave=valor y puede estar asociada a un sitio Web o a un URL. Una vez que se genera y el servidor la acepta, la cookie reside en el sistema del

explorador Web. Cada vez que el explorador solicite el mismo URL, o cualquier otro que se encuentre dentro de su dominio, se le entregará la información sobre ella al servidor. A la hora de configurar la cookie, el servidor le puede indicar al explorador que caduque la información sobre ella pasado cierto tiempo. A partir de entonces no se volverá a usar la cookie nunca más o, en el peor de los casos, durante cierto periodo de tiempo.

Hay mucha controversia sobre la utilización de las cookies. Muchos las consideran como una intrusión en la vida privada. El uso de las cookies para registrar los movimientos de una persona es una práctica muy extendida. De hecho son varias las compañías publicitarias que las utilizan para registrar los movimientos de los usuarios. Conviene destacar que las cookies no pueden hacer ningún daño. Los datos se guardan en un archivo de texto que se coloca dentro del directorio del explorador.

Antes de ver en profundidad el módulo mod_usertrack, me gustaría aclarar unas cuantas cosas. Antes había un módulo llamado mod_cookie que ha cambiado su nombre a mod_usertrack. De todas formas la versión anterior de este módulo podía generar sus propios registros gracias a la directriz CookieLog, que ha dejado de usarse. Ahora, el nuevo módulo dispone de un sistema de registro de cookie bastante más eficaz que el de su antecesor.

Por ejemplo, por medio de la directriz CustomLog, que se encuentra en el módulo de registro estándar, se pueden guardar las cookies en archivos separados.

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

Por razones de compatibilidad, el módulo de registro configurable incorpora la antigua directriz CookieLog, pero conviene actualizarse y utilizar CustomLog. Vamos a ver el nuevo módulo mod_usertrack.

Nota: recuerde que mod_usertrack no guarda los registros de las cookies. Simplemente genera una cookie para cada visitante. Se puede utilizar CustomLog (como ya hemos visto anteriormente) para guardarlas en un archivo de registro para su posterior análisis.

La directriz mod_usertrack no se compila con la distribución estándar de Apache, por lo que tendrá que volver a compilar el ejecutable antes de poder utilizarlo. Este módulo incluye las siguientes directrices.

CookieExpires

```
Sintaxis: CookieExpires caducidad
Contexto: configuración servidor, host virtual
```

Esta directriz se usa para determinar el periodo de expiración de las cookies que genera este módulo. El periodo de expiración se puede definir a partir del número de segundos o bien por medio de un formato como "1 month 2 days 3 hours". Por ejemplo:

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

Aquí, en la primera directriz se define el periodo de expiración en segundos y en la segunda se utiliza el formato especial. Obsérvese que cuando no se usa el formato numérico, se presupone el uso del formato especial. De todas formas, hay que utilizar dobles comillas. Si no se utiliza esta directriz, la cookie únicamente se utilizará con la sesión que tiene abierta el explorador.

CookieTracking

```
Sintaxis: CookieTracking on | off
Contexto: configuración servidor, host virtual, directorio, archivo
de configuración de acceso por directorio (.htaccess)
Override: FileInfo
```

Esta directriz le permite activar o desactivar la generación automática de cookies. Cuando se activa, Apache comienza por enviar una cookie en la que se registran todas las nuevas peticiones.

La directriz CookieTrack se puede utilizar para activar o desactivar la generación de cookies con cada directorio. Por defecto, al compilar mod_usertrack no se activan las cookies.

Uso de los registros de errores

En este capítulo hemos hablado de distintas formas de registrar los datos que pueden parecer interesantes. De todas formas, algo que conviene tener en cuenta es el registro de errores. Sin él sería completamente imposible determinar dónde y cuándo se produce un error.

El registro de errores lo soporta el propio Apache y a través de la directriz ErrorLog podrá tomar nota de todos los fallos que se encuentre Apache.

En esta sección veremos cómo puede incorporar la utilidad de registro de errores syslog (disponible en la mayoría de plataformas UNIX) en Apache.

Syslog es el sistema tradicional que se utiliza para registrar los mensajes que envían los procesos de un demonio (servidor). Es posible que se pregunte: "Apache es un demonio. Entonces, ¿por qué no puede escribir en syslog?" La verdad es que sí puede. Todo lo que tiene que hacer es sustituir la directriz ErrorLog que se encuentra en el archivo de configuración por:

```
ErrorLog syslog
```

y reiniciar Apache. Con un explorador Web acceda a una página inexistente que se encuentre en su servidor y busque en el archivo syslog una entrada httpd. Si quiere saber dónde se encuentran los mensajes httpd, busque en /etc/syslog.conf. Por ejemplo, el listado 11.1 nos muestra el contenido del archivo /etc/syslog.conf para un sistema Linux.

```
# Registre todos los mensajes del kernel en la consola.
# Demasiados registros saturan la pantalla.
#kern.*                         /dev/console

# Registre cualquier cosa (menos el mail) del nivel de información
# o superior.
# No registre los mensajes privados de autenticación.
*.info;mail.none;authpriv.none          /var/log/messages

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

# Guarde todos los archivos de correo electrónico en una misma
# ubicación.
mail.*                          /var/log/maillog

# Todo el mundo tiene acceso a los mensajes urgentes, así que los
# puede guardar en otra máquina.
*.emerg                                 *

# Guarde los errores de mail o news en un archivo de registro de
# errores o en otro especial.
uucp,news.crit                          /var/log/spooler
```

Listado 11.1. /etc/syslog.conf

Hay dos líneas muy importantes (por lo menos en lo que se refiere a Apache) en este listado:

```
*.info;mail.none;authpriv.none      /var/log/messages
*.emerg                             *
```

La primera le dice a syslog que escriba todos los mensajes del tipo info (exceptuando los de correo electrónico y los privados de autenticación) en el

archivo /var/log/messages y la segunda, que los mensajes urgentes se tienen que escribir en todos los archivos de registro. Por ejemplo:

```
ErrorLog syslog
LogLevel debug
```

Aquí, se le dice a Apache que envíe los mensajes de error a syslog. Si quiere almacenar estos mensajes en una ubicación diferente, tendrá que indicárselo a /etc/syslog.conf. Por ejemplo:

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

Si añade esta línea a /etc/syslog.conf y reinicia syslog y Apache, podrá almacenar todos los mensajes de depuración en el archivo /var/log/debug. Hay varias configuraciones para el registro:

- Emerg. Mensajes urgentes.
- Alert. Mensajes de alerta.
- Crit. Mensajes importantes.
- Error. Mensajes de error.
- Warn. Advertencias.
- Notice. Mensajes de avisos.
- Info. Mensajes de información.
- Debug. Mensajes que se generan en el proceso de depuración y que incluyen el archivo fuente y el número de la línea en la que se generó el mensaje. Así se ayuda en el proceso de depuración.

Truco: si quiere acceder a las actualizaciones de syslog o de cualquier otro archivo de registro, puede usar la utilidad tail que tienen la mayoría de los sistemas UNIX. Por ejemplo, si quiere ver las actualizaciones de un archivo llamado /var/log/httpd_errors:

```
tail -f /var/log/httpd_errors
```

Análisis de los archivos de registro

Hasta ahora hemos visto cómo crear los registros estándar con formato CLF y cómo personalizarlos. Ahora necesita una forma de analizar estos

registros para aprovechar su información. Las necesidades del análisis pueden variar considerablemente. Hay casos en los que es necesario generar grandes informes y otros en los que con una simple mirada es suficiente. Para las tareas más sencillas, lo mejor es utilizar lo primero que se tenga a mano. La mayoría de los sistemas UNIX tienen las suficientes utilidades y herramientas como para abarcar este tipo de tareas.

Vamos a fijarnos en una utilidad de UNIX diseñada para hacerse con una lista de los host. Si la utiliza o usa un archivo personalizado que pueda trabajar con CLF, conseguirá la lista requerida. Por ejemplo:

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

imprime todas las direcciones IP del host (si tiene activada la búsqueda DNS, entonces también se mostrarán los alias). La utilidad cat muestra el contenido del archivo access_log y la salida que genera se le entrega al intérprete awk, que tan sólo imprime el primer campo de cada línea por medio de la declaración print. Hemos quedado en que así se imprimen todos los host, pero ¿qué ocurre si se desea excluir los host de su red? En este caso tendría que utilizar:

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

Aquí, 206.171.50 se debería sustituir por las direcciones de su red de trabajo. Recuerde que he supuesto que el usuario tiene una red de clase C. Si es de clase B, sustituya los dos primeros octetos por sus direcciones IP. Esta versión le permite ejecutar su propio host utilizando la utilidad egrep, a la que se le indica que se limite a mostrar (a través de -v) los host cuya dirección de red no comience por 206.171.50. De todas formas, es posible que esta solución tampoco le satisfaga plenamente debido a las numerosas repeticiones. Por eso hemos creado otra línea de código que se puede adaptar mejor a sus necesidades:

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

Aquí, la utilidad uniq filtra las repeticiones y muestra un solo listado por host. Por supuesto, si desea ver el número total de servidores que han accedido a su sitio Web, tendrá que utilizar wc con la opción -l.:

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

De esta manera llevará la cuenta del total de servidores que acceden a su ordenador.

Con las utilidades de UNIX puede recopilar rápidamente la información que necesita, aunque la verdad es que para usar este método tiene que conocer algo del sistema que utiliza UNIX para mostrar la información que generan los programas o herramientas externas de análisis.

También se puede trabajar con las herramientas especializadas en los análisis de los exploradores Web. La mayoría espera que los archivos de registro estén en formato CLF, así que compruebe que todos los ficheros que genere están en este formato. En la tabla 11.2 tiene una lista con algunas herramientas y dónde localizarlas.

Tabla 11.2. Herramientas de análisis externas

Nombre del producto	URL del producto
WebTrends	www.webtrends.com/
Wusage	www.boutell.com/wusage/
wwwstat	www.ics.uci.edu/pub/websoft/wwwstat/
http-analyze	www.netstore.de/Supply/http-analyze/
pwebstats	www.unimelb.edu.au/pwebstats.html
WebStat Explorer	www.webstat.com/
AccessWatch	http://netpressence.com/accesswatch/

La mejor forma de ver cuál es la que mejor se adapta a sus necesidades es probarlas todas, o por lo menos visitar sus páginas Web para poder compararlos. Las dos utilidades que más me gustaron fueron Wusage y wwwstat.

De hecho, Wusage es la aplicación comercial especializada en registros que más me gusta. Sus opciones de configuración son enormes y produce una gran cantidad de informes gráficos utilizando las famosas librerías gráficas de la compañía. Se distribuye en formato binario. Hay copias gratuitas de evaluación para plataformas UNIX y Windows.

wwwstat es uno de los mejores programas de análisis gratuitos. Esta escrito en Perl, por lo que tan sólo habrá que instalar Perl en el sistema en el que se vaya a ejecutar la aplicación. gwstat puede leer la salida generada por esta utilidad y producir una serie de gráficos basándose en su contenido.

De momento, es muy probable que haya asimilado la idea de que crear registros en Apache es muy sencillo y útil. Hacerlo le permitirá conocer qué está ocurriendo en su servidor Apache. Además, los registros le ayudarán en todo momento a localizar errores, identificar los fallos del sistema y ver qué

propiedades son las que más gustan a sus usuarios. Los archivos de registro consumen mucho espacio de disco duro, por lo que conviene dedicar algo de tiempo a su mantenimiento.

Mantenimiento de los registros

Activar el registro de sucesos le puede ahorrar mucho trabajo, pero la verdad es que los archivos de registro por sí solos suponen más trabajo para el administrador del sistema, ya que tiene que dedicar algo de tiempo a su mantenimiento. En los sitios Apache con un alto porcentaje de dominios virtuales, los archivos de registro pueden llegar a alcanzar un gran tamaño en muy poco tiempo, por lo que se puede llegar a provocar un problema con el disco duro. Cuando el tamaño del archivo de registro es demasiado grande, se suele optar por renovarlo.

Dispone de dos opciones para hacerlo: con la utilidad que incorpora Apache, llamada rotatelog; o con logrotate, una facilidad disponible en la mayoría de los archivos Linux.

rotatelog

Apache incluye una utilidad de soporte llamada rotatelog. Para utilizar este programa haga lo siguiente:

```
TransferLog "| /path/to/rotatelogs <logfile> <tiempo de renovación en
segundos>"
```

Por ejemplo, si se quiere renovar el registro de acceso cada 86.400 segundos (es decir, cada 24 horas), tendrá que utilizar la siguiente línea:

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

De este modo, todos los días se generará un nuevo archivo de información que se guarda en /var/logs/httpd.nnn, donde nnn representa un número largo.

logrotate

La utilidad logrotate renueva, comprime y envía por correo electrónico los archivos de registro. Se ha diseñado para facilitarle el trabajo al administrador de dichos ficheros. Permite automatizar estas tareas cada día, semana o mes, o bien en el momento en que alcancen cierto tamaño.

Si su sistema puede trabajar con esta utilidad debería tener una configuración similar a la que le expongo a continuación en su archivo /etc/logrotate.conf:

```
/path/to/httpd/access_log {
compress
rotate 5
mail {hyperlink mailto:webmaster@yourdomain.com}
errors {hyperlink mailto:webmaster@yourdomain.com}
size=1024K
postrotate
        kill -HUP 'cat /path/to/httpd.pid'
endscript
}
```

Esta configuración especifica que se ha de renovar el contenido del archivo /path/to/httpd.pid cada vez que su tamaño supere los 1024 K y que los ficheros antiguos se han de comprimir y enviar por correo a webmaster@yourdomain.com después de llevar a cabo cinco renovaciones. Cualquier error que tenga lugar durante el proceso tendrá que comunicarse a root@yourdomain.com.

12 Seguridad en la Red

¿Ha pasado más de una noche sin dormir pensando en la seguridad de su servidor Web? No ha sido el único. Son muchos los administradores Web que lo hacen. La mayoría comienza a trabajar con pequeñas redes locales (LAN). Y es que hace unos cuantos años la mayoría de las empresas no comprendía el concepto de Internet. Aquellas empresas que trabajan con LAN se han quedado aisladas ya que carecen de protocolos que puedan enviar paquetes a través de Internet.

Pero durante los últimos años el panorama ha cambiado radicalmente. La mayoría de las empresas consideran su conexión a Internet como parte de su diseño de red.

Hoy, la conexión de una red LAN a Internet es una tarea muy sencilla y aun así hay muchos administradores que no llegan a controlar los entresijos de Internet. Y no terminan de sentirse a gusto con la seguridad de Internet (por no decir que le temen).

Por desgracia, un debate detallado sobre la seguridad de Internet está fuera del alcance de este libro. Pero, en este capítulo, trataremos de presentar una imagen global del modelo de seguridad, además de ver también el impacto que tiene en el servicio que se puede prestar a través de la Red. Aprenderá la manera de detectar estos peligros y de cómo implementar medidas de seguridad.

Necesidad de una seguridad Web

A la mayoría de los nuevos administradores Web, la idea de "seguridad necesaria" les resulta algo confusa. Muchos administradores se plantean la idea de seguridad después de que ocurra alguna desgracia. Este acercamiento pasivo no es nada conveniente, pero sí que resulta comprensible. Para un administrador puede llegar a resultar complicado preocuparse de algo que desconoce. Por eso trataré de dar una idea de los riesgos a los que hay que prestar especial atención.

En el momento en que enciende su servidor Web y lo pone a disposición del resto del mundo, ha abierto una ventana para que cualquiera que lo desee pueda acceder a su red. La verdad es que ésta era la idea, ¿verdad? La mayoría de la gente utilizará sus páginas Web, pero algunos buscarán ciertos agujeros a través de los que puedan hacerse con información a la que se supone que no tienen que acceder. También en Internet hay vándalos cuya única misión, además de robar información, es la de provocar situaciones molestas y embarazosas. En cualquier caso, si alguien encuentra un agujero, le saturará el sistema con material obsceno o le sustraerá datos importantes. A veces los ataques no afectan directamente a su sitio Web. Los atacantes pueden usar su sistema y recursos para acceder a otras redes y dejar que todas las pistas le señalen a usted como culpable. La verdad es que ninguna de las situaciones anteriores es nada apetecible.

Para evitar riesgos y demás situaciones embarazosas, asegúrese de crear los requisitos de seguridad que necesitan sus servicios Web. En este capítulo, daré por supuesto que su sitio Web dispone de los siguientes requisitos de seguridad:

- Mantener la integridad de la información que se publica en la Red.
- Evitar que se utilice su servidor Web como punto de infiltración a la red de la organización (esto puede provocar serias pérdidas de material confidencial, integridad de sistemas o fuentes de información).
- Evitar que se utilice el servidor Web como punto de paso de las intrusiones dirigidas a otras redes (su empresa se vería en serias dificultades legales).

La mayoría de los incidentes de seguridad Web ocurren porque la información se hace vulnerable al no configurar correctamente el software. Este software puede ser el propio servidor Web o las aplicaciones (como los programas CGI o las aplicaciones SSI o API) que se ejecutan en el servidor Web.

Por culpa de esto se puede llegar a perder información confidencial. El aprovechamiento de esta información puede dar lugar a otros ataques.

Si está seguro de que la configuración de su servidor es la correcta y de que las aplicaciones que se ejecutan en él son seguras, ¿podrá estar tranquilo? Pues no, porque los ataques contra su servidor Web se pueden dirigir por otras vías. La integridad del servidor Web se puede ver afectada por aplicaciones que no tienen nada que ver con él, por errores en el sistema operativo o por una mala arquitectura de red.

Para poder respirar tranquilo, tendrá que fijarse en todos los detalles. Esto se puede llegar a convertir en una tarea poco menos que imposible, pero recuerde que es imposible tener un entorno de seguridad completamente seguro. Nunca hay una solución global para todos los problemas de seguridad. El objetivo debería ser mejorar la seguridad de su sistema en la medida de lo posible.

Puntos de control relacionados con la seguridad

Para mejorar la seguridad de su servidor público, hay unos cuantos puntos que debería revisar. En la figura 12.1 tiene un diagrama donde aparecen. Este diagrama identifica los puntos relacionados con la seguridad que conviene examinar antes de poder asegurar que la conexión de su red a Internet es segura.

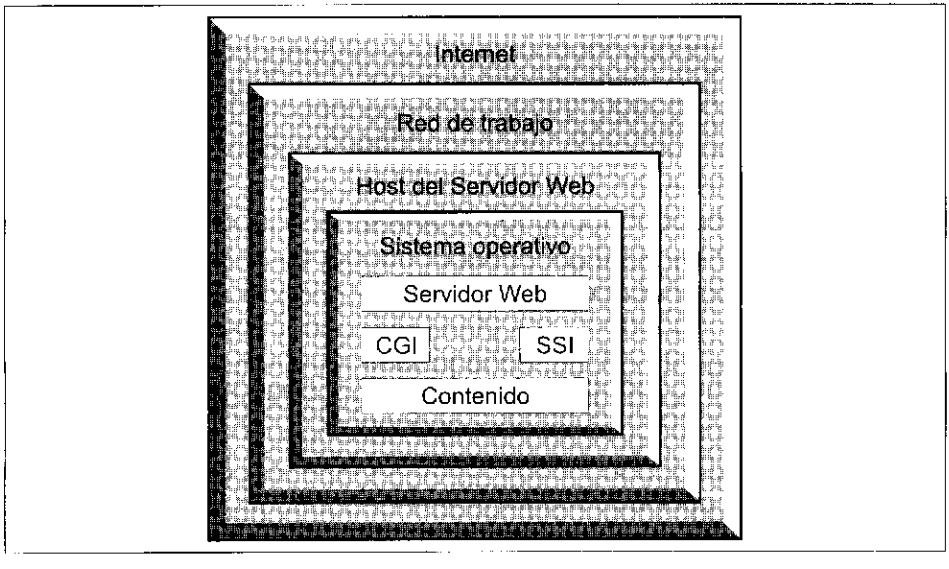


Figura 12.1. Puntos relacionados con la seguridad

Red de trabajo

El primer punto que hay que tener en cuenta es su red de trabajo y cómo se ha de hacer la conexión a Internet. El verdadero problema está relacionado con su servidor Web. Si su red tiene una configuración similar a la que se muestra en la figura 12.2, sepa que está desprotegida.

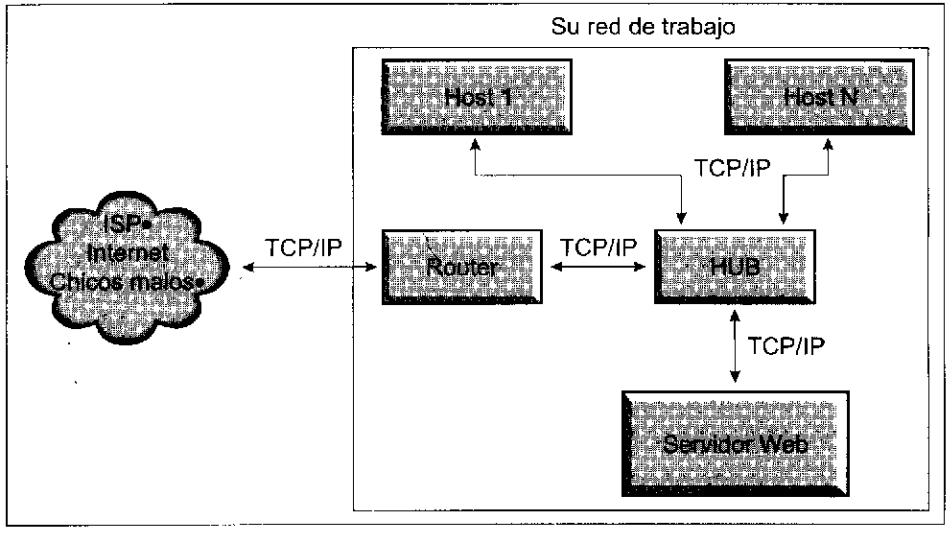


Figura 12.2. Una red sencilla con una conexión a Internet

Nota: para simplificar las cosas supondremos que la red únicamente tiene una conexión a Internet.

En este modelo de red de trabajo, todo el tráfico de Internet se puede enviar hacia y desde la red de trabajo. En otras palabras, cualquiera que se encuentre en Internet puede acceder a su red de trabajo (y viceversa). Es decir, que cualquier hacker puede encontrar, a través de Internet, un punto débil en su red de trabajo que le permita acceder a los host que tiene en su sistema. Obviamente, no todos estos host serán servidores Web con una serie de servicios listos para ser utilizados. Pero incluso una máquina Windows podrá convertirse en un posible objetivo. Como toda la red utiliza el protocolo TCP/IP, una opción de compartición de archivos puede provocar un fallo en las medidas de seguridad. La verdad es que esta red invita a casi todos los criminales a entrar en el sistema.

Si tiene esta configuración, conviene que revise la documentación de router para ver si puede activar las restricciones IP. Le recomiendo que evite por todos los medios la configuración expuesta en el diagrama anterior.

A continuación vamos a ver la configuración que tenemos en la figura 12.3. Recibe el nombre de Sacrificial Lamb (Chivo expiatorio).

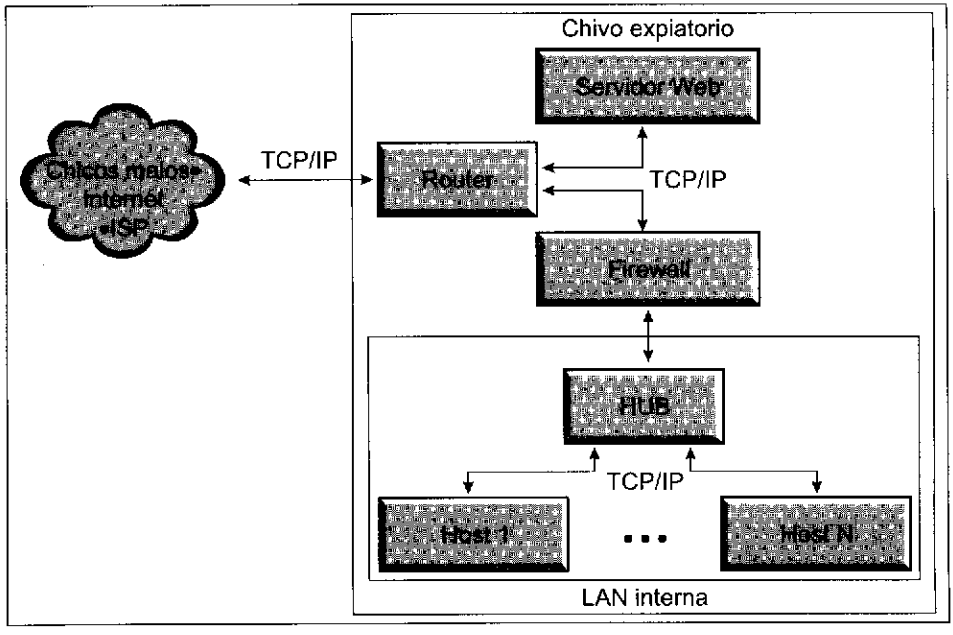


Figura 12.3. Configuración Chivo expiatorio para redes

El servidor Web se encuentra fuera del alcance del firewall y de la red LAN interna. Los host de la red LAN reciben los paquetes de Internet que les llegan a través del host. De esta forma, los host internos están a salvo de cualquier ataque externo (por lo menos en teoría). Por otro lado, el servidor Web está completamente expuesto a cualquier ataque. Esta es la razón por la que esta configuración recibe el nombre de Chivo expiatorio. En este supuesto se cumple lo siguiente:

- Un intruso no puede observar o capturar el tráfico de red que hay entre dos host. En este tráfico tenemos información de autenticación, información sobre los negocios del propietario, datos personales y otros tipos de datos de importancia.
- Los intrusos no pueden acceder al host interno ni obtener cualquier información sobre ellos.

Esta configuración protege contra dos tipos de riesgos y aísla el servidor Web de la red de trabajo y de la información que por allí se mueve.

Truco: en una configuración del tipo Chivo expiatorio conviene desactivar la posibilidad de enviar información al router. De esta forma, no se podrá utilizar el servidor Web para enviar paquetes a los host que se encuentran dentro de la red de trabajo.

Hay gente a la que le gusta colocar el servidor Web detrás de un firewall. Con esa configuración, el firewall se convierte en la puerta de enlace de la red LAN y del tráfico de la Red, pero se complica bastante su configuración. Mi opinión es que semejante complejidad termina por provocar la aparición de agujeros en la seguridad.

La idea principal es aislar la red interna y mantenerla alejada de los ojos curiosos. Conviene que considere la opción de instalar un firewall en la conexión que une su red LAN con Internet. Pero si el firewall es una opción que está fuera de su alcance, también puede pensar en instalar un servidor proxy. Este servidor toma la petición que efectúa una red a otra y se la entrega al destinatario. De esta forma las redes pueden intercambiar información sin llegar a estar conectadas. Es una excelente solución, además de incluir unas excelentes propiedades de registro.

Como se puede ver, es más importante preocuparse de la red interna que del propio servidor Web. Como las redes disponen de un servidor Web que tiene acceso a Internet, es muy posible que para irrumpir en de ellas se aproveche cualquier debilidad del software o de cualquiera de sus aplicaciones (como programas CGI). Si se decide a trabajar con una arquitectura de red que mantenga al servidor Web apartado de la red de trabajo, reducirá considerablemente este tipo de riesgos.

Aunque un servidor Web puede mantenerse apartado de la parte segura de la red, debería protegerlo para evitar cualquier acceso por parte de los hackers. En la siguiente sección estudiaremos la seguridad de los sistemas operativos y el software del servidor Web. Después de eso, veremos las distintas medidas que se pueden implementar para aumentar la seguridad del servidor.

El sistema operativo

Apache puede funcionar en las plataformas UNIX, Windows 95, Windows NT e incluso en los sistemas Macintosh. Pero que pueda funcionar en una plataforma no significa que se pueda ejecutar en cualquier sistema operativo

(OS). De hecho, la correcta elección del sistema operativo es uno de los puntos más importantes relacionados con la seguridad. Los sistemas UNIX y similares, como FreeBSD, SunOS, Solaris, HP-UX, Digital UNIX y Linux son los favoritos de los administradores Web. Aunque su calidad y efectividad pueden variar, a todos les encanta trabajar con ellos.

Si no quiere trabajar con UNIX, puede optar por Windows NT. Esta plataforma es uno de los competidores más importantes de UNIX. Pero hay que tener en cuenta un factor muy importante: Windows NT es nuevo en este juego que se llama Internet. La mayoría de los sistemas UNIX y similares llevan bastante tiempo viéndose las caras con los hackers y sus ataques. Y aún siguen vivos.

Una vez que haya escogido el sistema operativo, debería desactivar cualquier propiedad extra que no tenga la intención de utilizar con el servidor Web que ejecute dicho sistema operativo. Por ejemplo, si no va a utilizar los servicios FTP o SMTP/POP de correo, desactívelos por completo antes de eliminar los programas de correo del sistema. Si no necesita las utilidades más potentes del sistema, bórrelas. Tenga muy presente que su servidor Web no es más que eso, un servidor Web, y los servicios que presta han de estar de acuerdo con su finalidad.

Truco: puede aprender algo más sobre las vulnerabilidades de las distintas plataformas en www.cert.CERT. Esta organización trabaja con la comunidad de Internet para facilitar respuestas relacionadas con la seguridad. También efectúa una serie de investigaciones destinadas a mejorar la seguridad de los sistemas informáticos.

Software del servidor Web

Obviamente, el servidor Web que va a utilizar es Apache, ya que si no, no estaría leyendo este libro. Compruebe que está utilizando la última versión. Siempre es conveniente que compile sus propios archivos binarios en vez de utilizar los distribuidos.

El software de Apache es gratuito, por lo que es conveniente que se lo baje de una fuente de confianza. No se limite a copiarlos de cualquier FTP o página Web que se encuentre en la Red. Compruebe en primer lugar las páginas oficiales de Apache (www.apache.org). En el futuro, el código fuente de Apache contará con una firma PGP. En la actualidad se incluye un fichero PGP en el que aparece la firma digital de todos sus desarrolladores. Puede utilizarlo para autenticar las futuras versiones de Apache que cuenten con

una firma PGP. Los anuncios o avisos que envían los desarrolladores de Apache a través del correo electrónico también cuentan con una garantía de autenticidad PGP.

El grupo de firmas PGP de Apache se encuentra en el nivel más alto de la distribución de Apache. Si se fía del sitio Web del que ha sacado la fuente de la distribución de Apache, también puede copiar los módulos. Si no sabe cómo utilizar PGP en su sistema, aprenderá lo suficiente en:

```
www.pgp.com
```

En algunas de las distribuciones se incluye un archivo que contiene las firmas PGP de dicha distribución. Estas firmas se guardan en los archivos que tienen la extensión .asc (por ejemplo, la firma de apache_1.3.0.tar.gz se guardarán en apache_1.3.0.tar.gz.asc). Se pueden utilizar para verificar la validez de la distribución cuando se agregan los archivos KEYS.

A la última versión de Apache no se le conocen demasiados fallos. Pero esta capacidad para incorporar nuevos módulos ya de por sí supone un riesgo para la seguridad. La mayoría de los módulos están escritos por gente repartida por todo el mundo. Asegúrese de que tan sólo utiliza los módulos que copia de las páginas Web oficiales de Apache o aquéllos que estén registrados, que encontrará en el URL:

```
www.covalent.net/module_registry/
```

Nunca pruebe módulos en sus sistemas de producción. De hecho, siempre es conveniente determinar qué módulos necesita y cuáles no. A continuación compile Apache de acuerdo con estos requisitos. Si siegue estos consejos tendrá un servidor más rápido, pequeño y seguro.

A la hora de configurar Apache tendrá que prestar mucha atención a los puntos que le exponemos a continuación. La idea es desactivar todo lo que no vaya a utilizarse. De esta forma implementará ciertas medidas de seguridad que le ayudarán a reducir los peligros que acechan a su sistema.

Directrices User y Group de Apache

Como ya sabe, Apache puede ejecutarse como un sistema independiente o como un demonio inetd que se encargue de algún servicio. Si opta por ejecutar Apache como un demonio inetd, no tendrá que preocuparse por las directrices User y Group. Pero, si lo ejecuta como un servidor independiente, tendrá que asegurarse de que crea un usuario y un grupo dedicado a Apache. No utilice el usuario nobody (nadie) ni el grupo nogroup (nadie), especialmente si ya se han definido con anterioridad en el sistema. Al existir otros servicios y

ubicaciones con el mismo nombre, los administradores terminarán con un tremendo dolor de cabeza. La solución alternativa es crear un nuevo usuario y grupo y usarlos con las directrices User y Group.

Cuando utilice un usuario y un grupo dedicados para Apache, la administración de permisos del contenido del sitio Web es una tarea más sencilla. Todo lo que tiene que hacer es asegurarse de que el único usuario que tiene acceso a la Red es el administrador de Apache. Si tiene que crear un directorio para que las distintas aplicaciones tengan un espacio en el que puedan escribir datos, bastará con que active los permisos pertinentes de Apache.

Proteger ServerRoot y los directorios de registro

Asegúrese que nadie puede escribir en los directorios ServerRoot (sobre todo en los directorios y archivos de registro). El único que tendrá permiso será el usuario root. El usuario/grupo destinado a Apache no necesita ningún permiso para escribir o leer el contenido de los directorios de registro. Permitir que cualquier otra persona que no sea el usuario root pueda escribir en dichos directorios es un gran peligro desde el punto de vista de la seguridad.

Desactivar el acceso predeterminado

En un modelo estricto de seguridad no puede haber acceso predeterminado. Por eso no se debe caer en la costumbre de permitir el acceso al primero que llegue. Sólo dejará que entren los usuarios procedentes de ciertas localizaciones. Para implementar un acceso distinto al predeterminado, puede utilizar el siguiente segmento de código en uno de los archivos de configuración de Apache:

```
<Directory />
     Order deny,allow
     Deny from all
</Directory>
```

De esta forma anulará el acceso a todo. Si tiene que permitir que se acceda a un directorio, utilice el contenedor <Directory> para abrir dicho directorio. Por ejemplo, si quiere permitir el acceso a /www/mysite/public/htdpcs, añada las siguientes líneas de código:

```
<Directory /www/mysite/public/htdocs>
     Order deny,allow
     Allow from all
</Directory>
```

Este método es una medida de seguridad meramente preventiva y su uso está altamente recomendado.

Desactivar los overrides

Si no quiere que los usuarios anulen los parámetros de configuración utilizando el archivo de control de acceso de cada directorio (.htaccess), desactive la propiedad de la siguiente manera:

```
<Directory />
     AllowOverride None
     Options None
     allow from all
</Directory>
```

De esta forma no se le permite al usuario la posibilidad de usar cualquier directriz que anule la configuración del servidor principal. Además, conseguirá acelerar el servidor, ya que no tiene que detenerse a comprobar en cada uno de los directorios a los que accede si hay o no un archivo de control de acceso (.htaccess).

Sus contenidos

Cualquiera que tenga alguna relación con las medidas de seguridad tendrá que evaluar los distintos sistemas que hay para proteger sus datos cuando sale a navegar por la Red. Cuando vaya a determinar cómo proteger dichos contenidos, básese en los puntos de esta guía:

- Los archivos de contenido de la Red, como HTML, imágenes, sonido y vídeo, solamente tienen que poder leerse desde el servidor Web, pero el único que tendrá permiso de escritura será su dueño.
- Cualquier archivo o directorio que no se pueda mostrar en la Red no deberá encontrarse en ninguno de los directorios destinados a tal fin. Habrá que cambiarlos de sitio, porque, aunque no haya ningún enlace que conduzca hasta ellos, su acceso sigue siendo posible.
- Cualquier archivo temporal que se cree con generadores de contenido dinámico, como aplicaciones CGI, ha de residir en un subdirectorio independiente al que los generadores tengan permiso de acceso. Este directorio se ha de guardar fuera del área de contenidos para asegurarse de que un error en la aplicación no deja a la vista del mundo exterior el contenido de cualquier archivo.
- Para que queden bien claros los derechos del Copyright del contenido de sus documentos, debería colocar un aviso en todas sus paginas Web. Conviene que aparezca al principio del documento. Por ejemplo, en un

archivo HTML puede utilizar un par de etiquetas de comentario para incluir el mensaje del Copyright al principio del archivo. Si tiene pensado actualizar frecuentemente el contenido de sus mensajes, puede optar por una solución alternativa, utilizar la directriz #include.

- Si tiene cierta cantidad de contenido gráfico (imágenes) que desea proteger de los ladrones, debería utilizar la técnica de la marca de agua. Con ella agrega cierta información invisible a la imagen con la que protegerá los derechos del Copyright. La idea es que detecte si hay algún sitio Web que esté utilizando sus imágenes sin su permiso y podrá buscarlas gracias a esta información oculta. Si esta información coincide con el ID de su marca de agua, podrá identificar a la persona que esté violando los derechos del Copyright. Tenga en cuenta que las herramientas que existen hoy en día especializadas en los registros de las marcas de agua no son infalibles. Son muchos los programas que pueden quitar estas marcas. De todas formas merece la pena investigar en esta tecnología si trabaja con los gráficos.

A parte de estas guías conviene que tenga cuidado con los spiders y robots Web. Todos los motores de búsqueda, tales como Yahoo!, AltaVista, Excite e Infoseek utilizan aplicaciones automáticas que buscan en los índices de contenido de los sitios Web. Este sistema es bastante bueno, pero hay veces en las que deseará detener el acceso de estos robots a ciertas partes de su sitio Web.

Si el contenido de alguna parte de su sitio Web caduca con cierta frecuencia (por ejemplo, diariamente), no querrá que los bot de búsqueda lo incluyan en su índice. ¿Por qué? Porque cuando un cliente se apoya en un motor de búsqueda y encuentra un enlace que conduce a un contenido antiguo que ya no existe, dejará de estar contento. Este usuario pasará al vínculo siguiente sin volver a sus páginas Web.

Es posible que haya otros elementos que no quiera incluir en el índice de contenidos. Estos bot tienen un efecto secundario, ya que, al solicitar tantos documentos en tan poco tiempo, a veces pueden llegar a saturar los sitios Web por los que pasan. Se está trabajando mucho en el desarrollo de un comportamiento estándar para estos bot. La última versión de Robot Exclusion Protocol permite que los administradores Web coloquen archivos robots.txt en sus sistemas gracias a los cuales le indicarán a los robots por dónde no pueden ir. Por ejemplo, un archivo bitmap muy grande no tiene ninguna utilidad para los robots que intentan construir un índice con el contenido del sistema. Si puede acceder al directorio en el que se encuentra este archivo, restará recursos tanto al sistema del servidor como al del robot.

Cuando uno de estos robots visita un sitio llamado www.somesite.com, lo primero que hace es verificar la existencia del URL:

```
http://www.somesite.com/robots.txt
```

Si existe, el robot analiza su contenido en busca de las directrices que determinen cómo se ha de comportar. Como administrador de un servidor Web, puede crear directrices personalizadas para su sitio Web. Tenga en cuenta que únicamente puede haber un archivo robot.txt en el sistema. En él encontrará entradas como las siguientes:

```
User-agent: *
Disallow: /cgi-bin/
Disallow: /tmp/
Disallow: /~sam/
```

La primera directriz le indica al robot que las que expone a continuación están pensadas para que las ejecute cualquier robot. Las tres directrices que aparecen a continuación (Disallow) le indican al robot que no acceda a los directorios allí mencionados. Obsérvese que se ha de crear una directriz Disallow para cada URL que se quiera excluir. Es decir, que no podría utilizar la siguiente sentencia:

```
Disallow: /cgi-bin/ /tmp/ /~sam/
```

Tampoco pueden aparecer líneas en blanco en un registro, porque se utilizan para delimitar varios campos. En un código como el que acabamos de ver tampoco se pueden usar las expresiones regulares. El carácter * indica "cualquier robot".

Resumiendo, no se pueden tener líneas como ésta:

```
Disallow: /tmp/*
```

o

```
Disallow: *.gif
```

El robot podrá acceder a todo lo que no esté expresamente prohibido. A continuación le exponemos una serie de ejemplos:

Para que no entre ningún robot en el servidor habrá que utilizar la siguiente configuración:

```
User-agent: *
Disallow: /
```

Para dotar a los robots de un acceso completo, escriba:

```
User-agent: *
Disallow:
```

Puede obtener el mismo efecto borrando el archivo robots.txt. Para excluir a un único robot llamado WebCrawler:

```
User-agent: WebCrawler
Disallow: /
```

Para permitir el acceso de WebCrawler, tendrá utilizar la siguiente configuración:

```
User agent: WebCrawler
Disallow:
User-agent: *
Disallow: /
```

Para desactivar el acceso a un único archivo llamado /daily/changes_to_ often.html y evitar así que los robots lo incluyan en el índice, se pueden utilizar las siguientes líneas de código:

```
User-agent: *
Disallow: /daily/changes_to_often.html
```

Riesgos y Soluciones CGI

El mayor riesgo relacionado con la seguridad de los sitios Web viene de la mano de las aplicaciones CGI. En realidad, no es que este lenguaje de programación sea poco seguro, sino que las aplicaciones que no están bien programadas son un auténtico coladero para la seguridad del sistema. La facilidad de la especificación CGI hace que hasta los programadores más inexpertos puedan programar aplicaciones CGI. Estos nuevos programadores no están informados sobre los problemas relacionados con la seguridad de Internet y crean aplicaciones que funcionan pero que representan una auténtica puerta trasera para acceder a los agujeros del sistema a través de los cuales se pueden ejecutar aplicaciones. A continuación le muestro tres de los riesgos de seguridad que pueden crear las aplicaciones CGI:

- Pérdida de información. Puede ayudar a un hacker a descubrir nuevos métodos para acceder al sistema. Cuanta más información tenga el hacker sobre el sistema, más fácil le será entrar en él.

- Ejecución de los comandos del sistema a través de las aplicaciones CGI. En muchos casos los usuarios pueden manipular los script encargados de controlar los formularios HTML para ejecutar los comandos del sistema y acceder a información confidencial.
- Consumición de los recursos del sistema. Una aplicación CGI que no esté bien programada puede llegar a consumir muchos recursos del sistema, con lo que el servidor puede llegar a quedar inservible.

Por supuesto, deberá tener cuidado a la hora de instalar aplicaciones CGI y cuando el servidor Apache las ejecute como usuario carente de privilegios. De todas formas, no olvide que incluso las aplicaciones que se programan con cuidado también pueden suponer un peligro para la seguridad del sistema, por lo que convendrá protegerse de ellas.

Protegerse de los usuarios internos

La mayoría de los agujeros de seguridad que crean las aplicaciones CGI los provocan las entradas de los usuarios. En las siguientes secciones veremos cómo se pueden crear estos agujeros en la seguridad y cómo se puede proteger de ellos.

La entrada del usuario bloquea la aplicación CGI

El peligro más importante viene de la mano de los usuarios. Uno de estos problemas se produce cuando el tipo de programación utilizada permite que el usuario sature el sistema por medio de sus entradas. Así se puede llegar a crear un agujero en la seguridad del sistema, ya que si un programa capaz de saturar la memoria caché del ordenador cae en manos de hackers, es muy posible que aprovechen esta debilidad para acceder a su sistema. Este tipo de ataques son propios de lenguajes de programación como C o C++.

A continuación le mostramos un código escrito en C en el que se supone que los datos que introduce el usuario no pueden tener más de 1.024 caracteres, algo más que suficiente para declarar una array de 1.024 bytes. Gracias a esta suposición, si el usuario introduce numerosos datos, el sistema se colgará.

```
#include <stdio.h>
#include <stdlib.h>
static char query_string[1024];

char* POST() {
#
# Función para leer la entrada enviada con POST a través de un
# formulario HTML.
```

```
# Variable local entera
int size;

# usa la función getenv() para tomar el valor de la variable de
# entorno CONTENT_LENGTH, que alberga la cuenta de bytes de los
# datos introducidos por el usuario que se encuentran en STDIN.
# Convierte el valor de la cadena resultante de genet() en un
# entero, utilizando la función atoi() y devolviendo el valor al
# tamaño.
size=atoi(getenv("CONTENT_LENGTH"));

# Lee los datos introducidos.
fread(query_string, size,1,stdin);

# Devuelve la cadena de memoria que contiene los datos.
return query_string;
}
```

Cuando un usuario introduce una cantidad de datos que supera los 1.024 bytes, la rutina interrumpe el programa y abre un agujero en la seguridad del sistema. Es tales circunstancias, un hacker puede aprovecharse de este error para ejecutar comandos desde un sistema remoto.

Para solucionar este problema habría que utilizar una distribución de los datos dinámica, que se podría conseguir con las funciones malloc( ) o calloc( ).

La entrada del usuario hace que las llamadas del sistema no sean seguras

Otro de los problemas relacionados con las entradas de los usuarios surge cuando uno de ellos realiza una petición en la que la llamada del sistema abre un subshell que se encarga de procesar el comando. Por ejemplo, en Perl (lenguaje de programación en el que se escriben la mayoría de los programas CGI), este tipo de llamadas pueden efectuarse utilizando las llamadas system( ), exec( ), piped open( ) y eval( ). En C pasa algo parecido con las funciones popen( ) y system( ).

Obsérvese que cada shell script que utiliza las llamadas system( ) o exec( ) abre una puerta a través de la cual los hackers pueden entrar en el sistema. Recuerde que las comillas que se utilizan en Perl y otros lenguajes de interpretación para hacerse con las cadenas de texto también son peligrosas.

Para que veamos lo importante que es utilizar las llamadas del sistema, eche un vistazo a este "inocente" fragmento de código escrito en Perl:

```
#!/usr/local/bin/perl
#
# Propósito: demostrar los riesgos de seguridad que producen los
# script CGI pobremente escritos.
#
```

```
# Toma el nombre de_ dominio de la cadena de la variable de entorno
my $domain = $ENV{ÔQUERY_STRINGÕ};

# Imprime el tipo de contenido apropiado. Como la salida whois se
# encuentra en texto plano, escogemos el tipo de contenido
# text/plain.
#
print "Content-type: text/plain\n\n";

# Aquí tenemos una llamada de sistema errónea.
system("/usr/bin/whois $domain");

# Y ésta es otra que utiliza las comillas.
#
# my $output   `/usr/bin/whois $domain`;
#
# print $output;
exit 0;
```

Este pequeño script se supone que es una puerta de enlace para la Red basada en WHOIS. Si el script whois.pl se encuentra dentro del directorio cgi-bin de un sitio Web llamado www.notsecured.com, un usuario podría utilizar la siguiente entrada para llamar al script:

```
http://domain/cgi-bin/script.pl?domain anydomain.com
```

Este script tomará anydomain.com como la variable $domain que entra a través de QUERY_STRING y abre el programa /usr/bin/whois como argumento de $domain. Así se obtendrá el contenido de la base de datos WHOIS que mantiene InterNIC. Su aspecto es muy inocente, pero el script es un auténtico desastre. Vamos a considerar la siguiente línea:

```
http://domain/cgi bin/script.pl?domain nitec.com;ps
```

En este caso se hace que WHOIS busque un dominio llamado nitec.com y también le suministra la entrada a la utilidad ps de UNIX que se encargará de procesarlo. Así se consigue información sobre el sistema que se supone no se le puede entregar al usuario. Por medio de esta técnica cualquiera puede conseguir mucha información sobre su sistema, situación nada recomendable. Por ejemplo, al sustituir el comando ps por df (una utilidad de UNIX que imprime un resumen sobre el espacio de disco) se permitirá que cualquiera pueda ver las particiones establecidas en el sistema y lo llenas que están. Dejaré para su imaginación la cantidad de cosas que se pueden hacer con esta información.

¿Qué se aprende de esto? La moraleja es que no se puede confiar en la entrada ni utilizar las llamadas del sistema para abusar. A continuación veremos cómo se pueden conseguir estos objetivos.

Hay dos formas de conseguir que la entrada del usuario sea segura. Una de ellas es escanear la entrada en busca de cualquier carácter ilegal y sustituirlo o eliminarlo. Por ejemplo, en el script whois.pl que acabamos de ver, podemos añadir la siguiente línea:

```
$domain =~ s/[\/ ;\[\]\<\>&\t]//g;
```

Se encarga de eliminar todos los metacaracteres. Es una solución muy común pero poco aconsejable, ya que el programador tiene que estar atento ante las posibles combinaciones de caracteres que pueden llegar a causar problemas. Si el usuario utiliza una entrada que no haya previsto el programador, podrá utilizar el script para fines poco recomendables.

Una solución más aceptable es definir una lista de caracteres aceptables y sustituir o eliminar cualquier símbolo que no aparezca en ella. Este tipo de lista es pequeña, manejable y muy útil.

Con esta solución no hará falta que el programador trate de prever todos los caracteres no aceptables, admitiendo la existencia de cierto margen de error. Aquí, todo lo que tiene que hacer es asegurarse de que se identifican los caracteres aceptables y podrá despreocuparse de que los piratas traten de acceder al sistema saltándose las medidas de seguridad.

Basándonos en esta filosofía, podríamos sanear el contenido del programa en Perl que hemos visto anteriormente para que únicamente contenga los caracteres admitidos:

```
#!/usr/local/bin/perl
#
# Esta es una versión mejorada del script whois.pl.
#
# Se le asigna a la variable un grupo de caracteres aceptables para
# los nombres de dominio.
#
my $DOMAIN_CHAR_SET='-a-zA-Z0-9_.';

# Toma el nombre del dominio de la cadena de la variable de entorno
my $domain = $ENV{ÔQUERY_STRING'};

# Ahora elimina todos los caracteres no incluidos en el conjunto
# definido.
$domain =~ s/[^$DOMAIN_CHAR_SET]//g;

# Imprime el tipo de contenido apropiado. Como la salida whois se
# encuentra en texto plano, escogemos el tipo de contenido
# text/plain.
#
print "Content-type: text/plain\n\n";

# Aquí tenemos una llamada de sistema
system("/usr/bin/whois $domain");
```

```
# Y ésta es otra que utiliza las comillas.
#
# my $output = `/usr/bin/whois $domain`;
#
# print $output;
exit 0;
```

En la variable $DOMAIN_CHAR_SET se guardan todos los caracteres aceptables y se comparan con los que ha introducido el usuario, que se encuentran dentro de la variable $domain, para ver si hay algún carácter prohibido. Si los hay, se eliminan.

La mejor forma de analizar la entrada del usuario es establecer una serie de reglas (es decir, cómo se determinará que lo recibido es aceptable). Si espera que se introduzca una dirección de correo electrónico (en vez de escanear todos los posibles shell metacaracteres), utilice una expresión regular como la siguiente, para detectar la validez de la entrada:

```
$email = $input{'email-addr'};
if ($email=~ /^[\w-\.]+\@[\w-\.]+$/) {
       print "Posible dirección válida."
       }
else {
       print " Posible dirección no válida.";
       }
```

Pero no basta con limpiar la entrada del usuario. Tiene que tener cuidado con la forma en que se llama a los programas externos (por ejemplo, se puede invocar de varias formas a los programas en Perl). Algunos de dichos métodos serían:

Utilizando comillas simples puede capturar la salida de un programa externo, como:

```
$list = '/bin/ls -l /etc';
```

que captura el listado del directorio /etc. O bien se puede abrir una conexión (pipe) a un programa, como:

```
open (FP, " | /usr/bin/sort");
```

También puede llamar a un programa externo y esperar a que se le devuelva la salida que genera con system ( ):

```
system "/usr/bin/lpr data.dat";
```

o bien puede llamar a un programa externo y no utilizar nunca exec( ):

```
exec "/usr/bin/sort < data.dat";
```

Todas estas construcciones pueden entrañar cierto riesgo si en la respuesta del usuario aparecen determinados metacaracteres. Para system( ) y exec( ), hay ciertas propiedades sintácticas que le permiten llamar directamente a programas en vez de utilizar el shell. Si le entrega argumentos al programa externo (no sólo cadenas largas, sino elementos separados de una lista), Perl no usará el shell y sus metacaracteres no tendrán ningún efecto secundario:

```
system "/usr/bin/sort","data.dat";
```

Puede aprovecharse de esta ventaja para abrir una conexión sin tener que pasar a través del shell. Para invocar la secuencia de caracteres -|, abre una nueva copia de Perl y establece una conexión con ella. A continuación, el thread se hace cargo de otro programa utilizando el primer argumento de la llamada a la función exec.

Para leer a través de la conexión sin llegar a abrir el shell, puede hacer algo parecido con la secuencia -|:

```
open(GREP,"-|")   | exec "/usr/bin/grep",$userpattern,$filename;
while (<GREP>) {
  print "match: $_";
  }
close GREP;
```

Estas versiones de open( ) son más seguras que los open( ) con los que se han establecido conexiones, por lo que se deben usar siempre que sea posible.

Recuerde que hay muchas propiedades oscuras en Perl que le permiten llamar a programas externos y mentir sobre su nombre. Es realmente útil a la hora de llamar a los programas cuyo comportamiento varía dependiendo del nombre con el que se les llame. La sintaxis será la siguiente:

```
system $real_name "nombre_falso","argumento1","argumento2"
```

Uno de los trucos que utilizan los hackers para modificar el valor de la variable de entorno PATH es dirigirse al script que desean ejecutar (en vez de utilizar el programa que el programador ha previsto para tal fin). Por eso conviene ejecutar los programas utilizando su nombre completo en vez de confiar en el contenido de la variable de entorno PATH. Así, en vez de emplear este fragmento en Perl:

```
system("cat /tmp/shopping.cart.txt");
```

convendrá que use este otro:

```
system "/bin/cat" , "/tmp/shopping.cart.txt ";
```

Pero si tiene que fiarse del contenido de la variable PATH, determínela al principio de la aplicación CGI:

```
$ENV{'PATH'}="bin:/usr/bin:/usr/local/bin";
```

Aun en el caso de que no se fíe de la variable path cuando llame a un programa externo, hay una posibilidad de que el programa al que llama sí que lo haga. Por eso es conveniente que incluya la línea de código anterior al principio del script. Tenga en cuenta que tendrá que ajustar la línea para que aparezcan todos los directorios con los que desea trabajar. No es conveniente utilizar la llamada al directorio general (.).

Perl puede trabajar con una entrada oculta

Uno de los problemas más comunes relacionados con la seguridad de las aplicaciones CGI es que entregan sin darse cuenta variables al script de Perl. Este lenguaje de programación dispone de un mecanismo gracias al cual se pueden buscar entradas ocultas. Cualquier variable que no vaya a utilizar el programa (incluyendo las variables de entorno, STDIN y las procedentes de las líneas de comandos) se considera oculta y no se utilizará. Si utiliza una variable oculta para definir el valor de otra variable, esta última también se convertirá en oculta. Este tipo de variables no pueden utilizar llamadas eval( ), system( ), exec( ) u open( ). Si intenta utilizarlas, Perl le mostrará un mensaje de error. Además finalizará el programa si produce una llamada a un programa que no se encuentre dentro del path. Con Perl 5 se utiliza -T en la línea en la que se llama al intérprete de comandos:

```
#!/usr/local/bin/perl -T
```

Como ya hemos mencionado, una vez que se localiza una variable oculta, Perl no permitirá el uso de las funciones system( ), exec( ), pipcd open eval( ) o del comando backtrick (así como cualquier otra función que afecte a los programas externos, como unlink). No lo podrá utilizar ni siquiera en el caso de que se trate de localizar los metacaracteres del shell o emplear los comandos tr/// o s/// para eliminar dichos caracteres. La única forma de quitarle la condición de oculta a una variable será usar una operación de marcación y extraer las subcadenas localizadas. Por ejemplo, si espera que una variable contenga una dirección de correo electrónico, puede extraer una copia no oculta:

```
$e-mail=~/([\w-\.]+\@[\w\.]+)/;
$untainted_address = $1;
```

Como se puede ver, siempre es conveniente comprobar las variables ocultas cuando se trabaja con Perl.

La regla de oro es evitar pasar directamente la entrada del usuario al programa externo, ya que puede contener comandos ocultos. Evite el uso de funciones que puedan abrir otros programas mientras se ejecutan. Además, busque y elimine los metacaracteres que contengan las cadenas introducidas por el usuario.

El usuario ve datos ocultos

Este es otro de los peligros relacionados con la entrada de los usuarios. Está relacionada con un mal uso de los campos HIDDEN de los formularios HTML. Las aplicaciones CGI le suelen pedir a los usuarios que introduzcan información, la procesan y, a continuación, dependiendo de los resultados obtenidos, le piden que introduzca más datos. Como las aplicaciones CGI se ejecutan bajo petición, son incapaces de recordar los datos que introdujo el cliente la última vez que las llamó. Para solventar este problema, los programadores CGI guardan estos datos en campos HTML ocultos (HIDDEN). El aspecto de estos registros es como sigue:

```
<INPUT TYPE=HIDDEN NAME="variable" VALUE="valor">
```

No hay nada incorrecto en este método, pero hay que tener cuidado ya que, aunque se supone que estas variables son seguras, la realidad es bien distinta. Cualquier usuario puede ver su contenido gracias a la opción View Source (Visualizar Fuente) que incluyen los navegadores Web. Asegúrese de que los datos que guarda en estos campos no sean muy importantes. En otras palabras, no guarde el contenido de la variable PATH, ni nombres de usuario, nombres de archivo o contraseñas en estos campos.

Después de aplicar todas las precauciones vistas con las entradas del usuario y de asegurarse de que suponen el menor riesgo posible, puede que se pregunte si hay que hacer algo más. Pues no. Como comentamos anteriormente, nunca se puede contar con una seguridad completa. Es algo que se va aprendiendo con la experiencia. Mientras aprende a solventar un problema, aparece otro nuevo. En las secciones siguientes veremos algunos de los abusos más comunes relacionados con las aplicaciones CGI.

La entrada del usuario puede causar una negación de servicio

Uno de los ataques más comunes son las negaciones de servicio (DOS). Con uno de estos ataques se consigue anular la capacidad de respuesta del

sistema ante las peticiones de los usuarios honrados. También se pueden utilizar las aplicaciones CGI como bases a partir de las cuales desplegar este tipo de ataques.

La mayoría de las aplicaciones CGI toman los datos que introduce el usuario, efectúan algún proceso y devuelven una respuesta. Como en la Red cualquiera puede ejecutar este tipo de programas, los piratas los usan para montar sus ataques de DOS. Todo lo que tiene que hacer un hacker es utilizar una llamada a una aplicación CGI, a la que añade una entrada que inicie el proceso que desee. A continuación puede volver a llamar a la misma aplicación CGI o a cualquier otra para hacer que el servidor se dedique a abrir muchos procesos CGI.

De esta forma se consume una gran cantidad de recursos de sistema y se reduce la capacidad de respuesta dedicada a los clientes habituales del servidor. Desgraciadamente, estos ataques se pueden iniciar desde el laboratorio informático de cualquier universidad, que suelen contar con un gran número de ordenadores y mucho ancho de banda. Un pequeño script en Perl puede lanzar una petición HTTP al servidor y hacer que se repita una y otra vez. Si para ello se utilizan varias máquinas, se puede llegar a colgar el sistema del servidor Web. ¿Cómo se puede solventar esta situación?

La verdad es que no hay mucho que se pueda hacer al respecto. Pero una vez que ha tenido lugar uno de estos incidentes se puede prohibir cualquier comunicación con el sistema que lo provocó. Puede utilizar el contenedor <Limit ...> para negar el acceso a dicho servidor.

También puede utilizar las directrices RLimitCPU y RLimitMEM para ajustar el uso de la CPU y la memoria de Apache.

Se envían las entradas del usuario a través de POST

Otro de los ejemplos de vandalismo relacionados con CGI que se puede ver a través de los sitios Web es el caso en que los usuarios escriben un texto que más tarde se envía por medio de POST al sistema Web. Por ejemplo, los sitios Web que utilizan grupos de discusión o libros de invitados basados en aplicaciones CGI son propensos a caer víctimas de estas sesiones de vandalismo. Si sus aplicaciones CGI permiten que los usuarios escriban un texto y lo envíen por medio de POST a su sistema, tendrá que asegurarse de desactivar el uso de las etiquetas HTML en cualquiera de los campos de entrada de los usuarios. La forma más sencilla de hacerlo es sustituyendo los símbolos "<" por "<". De esta forma, cuando se dibuja algo en un servidor Web, aparecerá la etiqueta HTML en la pantalla en vez de pasárselo al explorador Web para que lo ejecute. La línea siguiente de Perl hace exactamente eso:

```
$user_input=~ s/</&lt;/g;
```

Muchos vándalos encuentran sitios que permiten escribir en sus libros de invitados y utilizan etiquetas HTML para enlazarlos con sitios obscenos o para cargar imágenes muy grandes, con lo que se sufrirá un retraso bastante grande en la carga de la página. También se puede pegar un JavaScript en la entrada de un libro de invitados. Para evitar este tipo de situaciones conviene utilizar la sustitución que hemos visto anteriormente.

Reducir los riesgos de las aplicaciones CGI

La mejor forma de reducir los riesgos relacionados con CGI es no ejecutar ninguna aplicación CGI. Pero como estamos en una época en la que está de moda el contenido dinámico de las páginas Web, esta solución tan radical no es nada recomendable. Pero sí que se puede centralizar la localización de todas las aplicaciones CGI para poder vigilarlas con más detenimiento.

En muchos casos, sobre todo en los sistemas de los proveedores de servicios de Internet (ISP), todos los usuarios con páginas Web quieren contar con acceso CGI. En estos casos es conveniente ejecutar las aplicaciones CGI bajo el ID de su propietario. Por defecto, las aplicaciones CGI que ejecuta Apache utilizan este sistema. Con este método, el daño que puede realizar un script CGI está limitado por los permisos con los que cuente su propietario. En otras palabras, un script dañino que se ejecute con el ID de un usuario que no sea el administrador del sistema, únicamente podrá dañar a los archivos de su propietario, por lo que tendrá más cuidado porque puede llegar a ser el máximo perjudicado de sus propias aplicaciones CGI. Así se consigue que los usuarios del sistema sean más cuidadosos y se limita el área potencial de daños. Para ejecutar una aplicación utilizando un ID diferente del perteneciente al servidor Apache, necesitará una propiedad de programación llamada wrapper. Un wrapper es un programa que le permite ejecutar una aplicación CGI como si se tratase del propietario de la aplicación. Algunos wrapper CGI efectúan ciertas comprobaciones de seguridad cuando se les solicitan aplicaciones CGI. En las siguientes secciones veremos dos de los wrapper más populares.

suEXEC

Apache puede trabajar con una aplicación llamada suEXEC que permite que sus usuarios ejecuten programas CGI y SSI bajo una identificación ID distinta a la de Apache.

Nota: es posible que suEXEC sólo funcione con plataformas UNIX o similares.

suEXEC es un programa que se encarga de establecer el número ID del usuario (setuid wrapper) que utilizará el programa que se abre cuando se recibe una petición HTTP procedente de un programa CGI o SSI. Es el administrador quien designa el ID de usuario que se utilizará durante la ejecución de un programa. Cuando se efectúa una de estas peticiones, Apache le entrega el nombre del programa y los ID del usuario y de los grupos a los que pertenece suEXEC. El wrapper ejecuta el programa utilizando esta información.

Antes de ejecutar los comandos CGI o SSI, suEXEC efectúa una serie de comprobaciones para asegurarse de que la petición es válida. Entre otras cosas, se comprueba lo siguiente:

- La llamada cuenta con el número apropiado de argumentos.
- La ejecuta un usuario que cuenta con el permiso oportuno. Lo normal es que el único que lo pueda ejecutar sea el administrador de Apache.
- Se le pasa una aplicación CGI o un comando SSI (objetivo del comando exec) que se encuentra dentro del espacio de Apache. La aplicación solicitada ha de encontrarse dentro del DocumentoRoot especificado en ese directorio. Nadie, a excepción del propietario, puede escribir en dicho directorio o en la propia aplicación. Esta aplicación no puede ser un programa setuid o setgid, ya que éstos modifican los ID de los usuarios o de los grupos de usuarios.
- Se le pide que ejecute una aplicación CGI o un comando SSI (objetivo del comando exec) utilizando el ID de un usuario o un grupo. Obsérvese que suEXEC no permite que ni el usuario root ni el grupo root ejecuten ningún programa. Tanto el ID del usuario como el del grupo han de estar por debajo de los números de identificación especificados durante la configuración. De esta forma se permite el bloqueo de las cuentas del sistema y grupos.

Una vez que se han efectuado con éxito las comprobaciones anteriores, el wrapper suEXEC cambia el ID del usuario y del grupo por el número de identificación definido a través de setuid o setgid. También se abre la lista de acceso del grupo para saber a qué grupos pertenece el usuario. suEXEC limpia el proceso de entorno estableciendo un path de ejecución seguro (que se define durante la configuración), a la par que se pasa solamente los nombres de los grupos que aparecen en la lista a través de estas variables (que también se crean durante el proceso de configuración). A continuación, el proceso suEXEC se convierte en el objetivo de la aplicación CGI o comando SSI y lo ejecuta.

A primera vista puede parecer que este método implica una gran cantidad de trabajo. Y es así. Pero la verdad es que también implica un elevado nivel de seguridad.

Configuración e instalación de suEXEC

En los directorios en los que se encuentra el código fuente de Apache encontrará los dos archivos de suEXEC (suexec.h y suexec.c). Para configurar y compilar suEXEC tendrá que seguir los pasos que encontrará en la sección siguiente:

Paso 1: editar el archivo suexec.h

Para editar el archivo suexec.h escriba lo siguiente:

```
#define HTTPD_USER "www"
```

Cambie www por el ID del usuario que quiere utilizar con la directriz User del archivo de configuración de Apache. Este será el único usuario que podrá ejecutar el programa suEXEC.

La siguiente macro define el ID del usuario más pequeño que se puede utilizar como objetivo:

```
#define UID_MIN 100
```

En otras palabras, los números de identificación de usuarios que sean menores que el ID definido en la línea anterior no podrán ejecutar los comandos CGI o SSI a través de suEXEC. Vamos a ver el archivo /etc/passwd para asegurarnos de que el rango de números escogido no se encuentra dentro de las cuentas del sistema, que suelen ser menores de 100.

La macro que aparece a continuación define el ID menor de un grupo que se puede utilizar como objetivo:

```
#define GID_MIN 100
```

En otras palabras, los números de identificación de los grupos que sean menores que el ID definido en la línea anterior no podrán ejecutar los comandos CGI o SSI a través de suEXEC. Vamos a ver el archivo /etc/passwd para asegurarnos de que el rango de números escogido no se encuentra dentro de las cuentas del sistema, que suelen ser menores de 100.

La siguiente macro define el subdirectorio que se encuentra dentro del árbol de directorios del usuario, en el que se guarda el ejecutable suEXEC:

```
#define USERDIR_SUFFIX "public_html"
```

Nota: si la directriz UserDir apunta hacia una ubicación que no se encuentra dentro del directorio principal del usuario referido en el archivo /etc/passwd, no se ejecutará suEXEC.

Si tiene una serie de servidores virtuales con un UserDir diferente para cada uno de ellos, tendrá que definirlos todos para que se encuentren dentro del directorio principal y renombrarlos:

```
#define LOG_EXEC "/usr/local/apache/logs/cgi.log"
```

Así le indica a suEXEC donde ha de guardar los registros de error.

La siguiente macro define DocumentRoot de Apache:

```
#define DOC_ROOT "/usr/local/apache/htdocs"
```

Todas las aplicaciones CGI y los comandos SSI que puede ejecutar suEXEC tendrán que encontrarse dentro de este árbol de directorios, a no ser que sean parte del directorio que se especifica a través de la directriz UserDir.

La macro que le mostramos a continuación define la variable de entorno PATH que ejecuta suEXEC con las aplicaciones CGI y los comandos SSI:

```
#define SAFE_PATH "/usr/local/bin:/usr/bin:/bin"
```

Paso 2: compilar e instalar suEXEC y Apache

Una vez que se ha editado correctamente suexec.h, puede compilar el programa suexec de la siguiente manera:

```
gcc suexec.c -o suexec
```

Así se creará el ejecutable suexec. Si tiene pensado guardar este ejecutable en cualquier otro lugar que no sea /usr/local/apache/sbin, tendrá que volver a compilar Apache. En este caso, edite el archivo httpd.h y modifique la macro para que se dirija a la nueva localización de suexec:

```
#define SUEXEC_BIN "/usr/local/apache/sbin/suexec"
```

Compile Apache y coloque el nuevo ejecutable en su nueva ubicación.

Paso 3: determinar los permisos de suEXEC

Como el único que puede modificar los ID de los usuarios de los programas es el usuario root, tendrá que hacer que sea él el propietario de suexec:

```
chown root /path/to/suexec
```

También ha de utilizar setuserid:

```
chmod 4711 /path/to/suexec
```

Así se completa la instalación de suEXEC. Veamos cómo se puede usar.

Activar y desactivar suEXEC

Una vez que ha instalado suexec y el nuevo ejecutable de Apache en la ubicación adecuada, tendrá que reiniciar el servidor. No utilice la señal HUP (-1) para reiniciarlo, sino TERM (-9). Si la configuración es la correcta, se encontrará con un mensaje similar a éste:

```
Configuring Apache for use with suexec wrapper.
```

Si no ve este mensaje, verifique que ha instalado el wrapper en la ubicación adecuada, que especifica la macro SUEXEC_BIN que se encuentra en el archivo httpd.c. Por otro lado, si le aparece el mensaje, lo tendrá todo listo para ejecutar suEXEC. Si quiere probarlo, coloque una aplicación CGI en el directorio cgi-bin del usuario y acceda a ella a través de un localizador URL:

```
http://www.yourdomain.com/~user/cgi-bin/scriptname
```

Apache usa suEXEC con todos los comandos CGI y SSI, como por ejemplo peticiones exec, que contengan el carácter ~.

El ejemplo que le exponemos a continuación le ayudará a comprender cómo funciona la configuración de suEXEC. Partimos de la base de que en una llamada del servidor Apache a wormhole.nitec.com, el valor de UserDir será public_html y que el usuario kabir tiene como directorio principal /home/kabir, en el que está el directorio public_html. También asumiremos que test.cgi se encuentra en el directorio ~kabir/public_html.

```
#!/usr/local/bin/perl
print "Content-type: text/html\r\n";
print "Prueba de suEXEC<BR>";
print "PATH = $ENV{PATH}";
exit 0;
```

Ahora, cuando accedemos al URL:

```
http://wormhole.nitec.com/~kabir/test.cgi
```

obtenemos la respuesta esperada. El archivo cgi.log (especificado en la macro LOG_EXEC, que se encuentra en suexec.h) mostrará lo siguiente:

```
[00:50:15 12-01-98]: uid: (kabir/kabir) gid: (kabir/kabir) test.cgi
```

Está bien. Vamos a ver ahora la salida que obtenemos al introducir ls -l con ~kabir/test.cgi:

```
-rwxr-x-r-x  1  kabir   kabir 85 Jan 12 00:50 test.cgi
```

Observe qué ocurre cuando cambio el propietario del script y le asigno el usuario root a través del siguiente comando:

```
chown root test.cgi
```

Si intento volver a ejecutar el script, obtendré un error del servidor y el archivo de registro mostrará la siguiente línea:

```
[01:01:30 12-01-98]: target uid/gid (500/500) mismatch with directory
(500/500) or program (0/500)
```

Aquí el programa pertenece al usuario con ID 0, y el grupo sigue siendo kabir (500), por lo que suEXEC no lo ejecuta. Como se puede ver, suEXEC está haciendo exactamente lo contrario de lo que se supone que tiene que hacer. Para asegurarse de que suEXEC va a ejecutar el programa test.cgi en otro directorio, he creado ~kabir/public_html y lo he puesto allí. Después de determinar el usuario y el grupo propietarios del nuevo directorio y archivo, el ID del usuario y del grupo será el de kabir, por lo que accederemos al script a través del siguiente URL:

```
http://wormhole.nildram.com/~kabir/cgi-bin/test.cgi
```

Funciona como estaba previsto. Obsérvese que este pequeño script me permite determinar el valor de la variable PATH con otra comprobación. Como he determinado el valor del path de seguridad con la macro SAFE_PATH que se encuentra en suexec.h, le puedo asegurar que PATH es exactamente lo que habíamos previsto. Debería hacer otras comprobaciones similares para asegurarse de que la configuración de suEXEC es la correcta.

Si tiene host virtuales y quiere ejecutar programas CGI o comandos SSI utilizando suEXEC, tendrá que utilizar las directrices User y Group dentro del contenedor <VirtualHost ...>. Con estas directrices determinará unos números de identificación de usuario y grupo distintos a los que ya está utilizando Apache. Si tan sólo se especifica una de estas directrices (o ninguna), se utilizarán los ID predeterminados por Apache.

Por razones de eficiencia y seguridad, todas las peticiones suexec han de mantenerse dentro del nivel root de los documentos del host virtual, o un nivel por encima del directorio root que se utiliza con las peticiones personales. Por ejemplo, si tiene cuatro host virtuales configurados en su sistema, tendrá

que configurarlos de tal forma que el directorio root de los documentos se encuentre fuera del directorio de documentos de Apache, gracias a lo cual se podrá aprovechar la ejecución de suEXEC.

A continuación vamos a ver otro wrapper CGI muy extendido, que se llama CGIWrap.

CGIWrap

CGIWrap es muy parecido a suEXEC, ya que permite que los usuarios puedan utilizar aplicaciones CGI sin comprometer la seguridad del servidor Web. Los programas CGI se ejecutan con el permiso de su propietario. Además, CGIWrap efectúa una serie de comprobaciones de seguridad sobre la aplicación CGI y no la ejecutará si falla alguna de ellas.

El autor de CGIWrap es Nathan Neulinger y podrá encontrar su última versión en ftp://ftp.cc.umr.edu.

CGIWrap se utiliza a través de un URL que se encuentra dentro de un documento HTML. La versión que se distribuye está configurada de tal forma que un usuario ejecuta los script que se encuentran en el directorio ~/public_html/cgi-bin/.

Configuración e instalación

CGIWrap se distribuye como un archivo tar comprimido con gzip. Puede descomprimirlo con gzip y ejecutarlo con la utilidad tar.

Ejecute el script Configure y responda a todas las preguntas que se le formulen. La mayoría de estas preguntas son autoexplicativas. De todas formas, este wrapper tiene una función que lo hace diferente de suEXEC. Le permite crear archivos allow y deny que se pueden utilizar para permitir o restringir el acceso a sus aplicaciones CGI. Ambos archivos tienen el mismo formato, tal y como se puede ver en la siguiente línea:

```
User ID
mailto:Username@subnet1/mask1,subnet2/mask2...
```

También se puede tener un único nombre de usuario (no un número ID de usuario) o una línea mailto:ID@subnet/mask en la que se puedan definir los pares subnet/mask, por ejemplo:

```
mailto:Myuser@1.2.3.4/255.255.255.255
```

Si esta línea se encuentra dentro del archivo allow (el nombre del fichero lo tendrá que escoger el usuario), el host virtual que pertenece a la red 206.171.50.0 con la subred 255.255.255.0 será el único que pueda ejecutar las aplicaciones CGI de kabir.

Después de ejecutar el script Configure, tendrá que ejecutar la utilidad make para crear el ejecutable CGIWrap.

Activar CGIWrap

Para utilizar la aplicación wrapper, copie el ejecutable CGIWrap en el directorio cgi-bin del usuario. Obsérvese que este directorio tiene que coincidir con el especificado durante el proceso de configuración. El método más sencillo es utilizar la estructura de directorios ~username/public_html/cgi-bin para el directorio de aplicaciones CGI.

Una vez que ha copiado el ejecutable CGIWrap, tendrá que modificar los permisos de propiedad:

```
chown root CGIWrap
chmod 4755 CGIWrap
```

Cree un vínculo nph-cgiwrap, nph-cgiwrapd y cgiwrapd a CGIWrap que se encuentra en el directorio cgi-bin:

```
ln [-s] CGIWrap cgiwrapd
ln [-s] CGI_Wrap nph-cgiwrap
ln [-s] CGIWrap nph-cgiwrapd
```

Obsérvese que en mi servidor Apache tan sólo he especificado la extensión .cgi como aplicación CGI y, por lo tanto, he renombrado el ejecutable CGIWrap como cgiwrap.cgi para que funcione aun en el caso de tener una serie de restricciones similares aplicables a este vínculo.

Ahora bien, para ejecutar la aplicación CGI tendrá que actuar como sigue:

```
www.yourdomain.com/cgi-bin/cgiwrap/username/ nombre del script
```

Por ejemplo, para acceder a la aplicación CGI test.cgi de kabir que se encuentra en wormhole.nitec.com, utilizo el siguiente URL:

```
http://wormhole.nitec.com/cgi-bin/cgiwrap/kabir/test.cgi
```

Truco: si desea analizar la salida de su CGI, especifique cgiwrapd en vez de CGIWrap a través de la siguiente URL:

```
http://www.yourdomain.com/cgi-bin/cgiwrapd/username/scriptname
```

Si el script es del estilo nph-, tendrá que ejecutarlo utilizando este URL:

```
www.yourdomain.com/cgi-bin/nph-cgiwrap/username/scriptname
```

Peligros y soluciones de SSI

Muchos administradores consideran SSI (Server Side Includes) igual de peligroso que las aplicaciones CGI. En cierto sentido es verdad, pero en otros no.

Si ejecuta aplicaciones externas utilizando comandos SSI como exec, el riesgo que supone para la seguridad es el mismo que si ejecuta aplicaciones CGI. De todas formas, desde Apache se puede desactivar este comando sin ninguna dificultad, gracias a la directriz Options:

```
<Directory />
Options IncludesNOEXEC
Order deny,allow
Deny from all
</Directory>
```

Así se desactiva la ejecución de exec y de los comandos SSI desde cualquier parte de su sitio Web. De todas formas, podrá activar estos comandos siempre que sea necesario definiendo un alcance de los directorios bastante más ajustado. Por ejemplo:

```
<Directory />
Options IncludesNOEXEC
Order deny,allow
Deny from all
</Directory>

<Directory /risky>
Options Include
Order deny,allow
Deny from all
</Directory>
```

Este segmento de configuración desactiva la ejecución del comando exec a todo el mundo excepto a aquéllos que se encuentren en el directorio /risky.

Peligros de la autenticación HTTP básica

Otro peligro relacionado con la seguridad lo tenemos con el protocolo HTTP y la autenticación básica. Las contraseñas que se utilizan en este tipo de autenticación viajan por la Red en formato uu-codificado. Como este formato no está codificado, no es nada seguro.

Para que un hacker descodifique una contraseña uu-codificada, tiene que contar con un sniffer que encuentre el paquete exacto en el cual viaja la

contraseña, de entre todos los paquetes IP que viajan por la Red. Aunque las posibilidades son remotas, la verdad es que es posible.

Además, debemos considerar la autenticación HTTP actual como insegura y no es nada recomendable utilizarla con datos de suma importancia. Si necesita un sistema de autenticación potente, deberá encriptar la contraseña antes de que viaje del usuario al servidor. Para eso está el protocolo SSL.

Consideraciones sobre la política de seguridad

Una política de seguridad identifica las prácticas que son vitales para alcanzar una gran robustez en la red del sistema. Si no utiliza ninguna, considere trabajar con algunos de los puntos que le sugerimos en las siguientes secciones.

Registrarlo todo

Los archivos de registro del servidor guardan información sobre su comportamiento en respuesta a lo solicitado en cada petición. El análisis de los archivos de registro puede suministrar información sobre el negocio (por ejemplo, qué páginas Web son las más populares) y sobre la seguridad. Asegúrese de configurar Apache para que tome nota tanto de los errores como de los intentos de acceso. Gracias a los archivos de registro podrá saber quién accede a qué. Acostúmbrese a tomar nota de todo lo que pueda y, si encuentra algo poco corriente, examínelo. Posiblemente a lo que tenga que prestar más atención sea al registro de errores.

Conservar una copia de su sitio Web

Conserve una copia de su sitio Web en un servidor más seguro. Si la integridad de la información pública de su servidor Web llega a peligrar, contará con una copia para restaurar el original. Normalmente, esta copia de seguridad se guarda en un servidor al que sólo puede acceder el administrador del sitio Web (y, quizá, el personal de la empresa responsable de la creación y mantenimiento de las páginas Web). Por eso debería encontrarse en la red de trabajo interna de su empresa.

Para contar con cierto grado de seguridad, tendrá que aplicar robustas tecnologías criptográficas y de comprobación de archivos, que sean capaces de generar para cada fichero un checksum (número que se utiliza para comprobar la integridad de un archivo). Las copias de los archivos y checksum del

medio (protegido contra escritura o que tan sólo cuenta con permiso de lectura) se guardan físicamente en una ubicación segura. Puede utilizar la encriptación MD5 para generar checksum encriptados de cada uno de los archivos.

Administrar el sitio Web desde una consola de un host Web

Conviene administrar su sitio Web desde una consola de un host Web. De esta forma se elimina la existencia del tráfico de datos entre el servidor Web y la estación de trabajo del administrador. De todas formas, hay un par de situaciones en las que no es posible implantar este sistema (como en las organizaciones en las que el administrador no puede acceder directamente al servidor Web). Cuando tenga que administrar el sistema a distancia, asegúrese de que utiliza un sistema de autenticación potente para registrarse en el servidor Web. Si utiliza una herramienta de administración, compruebe que no trabaja con una autenticación HTTP básica. En otras palabras, querrá que las contraseñas no viajen de una estación de trabajo a otra en un formato sin encriptar. Además, tendrá que configurar el servidor Web para que acepte conexiones procedentes de un único host que se encuentre en la misma red de trabajo.

Tenga cuidado con las aplicaciones CGI de dominio público

Siempre que utilice una aplicación CGI de dominio público, tendrá que asegurarse de que alguien en su organización (de dentro o de fuera) comprenda a la perfección el código fuente. Nunca se haga con una copia de una aplicación que proceda de una fuente no fiable. Busque en USENET para ver si alguien ha descubierto problemas en las aplicaciones que tiene previsto instalar en su sistema. Si fuese posible, instale la aplicación en un servidor de prueba. Revise los archivos de registro durante el periodo de prueba. Si la aplicación utiliza servicios de correo, revise todos registro generados.

Compare contenidos

A menudo, los intrusos sustituyen, modifican y dañan los archivos de sistema a los que acceden. Para acceder una y otra vez a los sistemas de forma ilegal, modifican los programas para que parezca que funcionan a la perfección, pero incluyen una puerta trasera por la que se pueden colar y acceder al sistema. También modifican los archivos de registro para eliminar cualquier

rastro de sus actividades. Incluso son capaces de crear nuevos archivos en los sistemas que visitan.

Por lo tanto, es una buena idea comparar los atributos y contenidos de los archivos y directorios de la copia. Si ha creado números checksum encriptados para esos archivos, puede compararlos con los que tiene en su copia actual para determinar si hay alguna diferencia.

Uso de MD5 para verificar la integridad del contenido del archivo

El programa MD5 genera un valor encriptado de 128 bits que se obtiene a partir del contenido de un archivo. Este valor se considera tan fiable como una huella dactilar y se utiliza para verificar la integridad del contenido de un fichero. Incluso si se cambia un solo bit del archivo, el número checksum que genera MD5 cambiará. Es extremadamente complicado modificar un archivo de tal forma que genere el mismo checksum del original.

Una forma de comprobar la integridad de los archivos de un sistema, aplicación y archivos de datos es a través de los números de checksum que genera MD5.

En RFC 1321 encontrará toda la información sobre el sistema checksum encriptado de MD5. En el siguiente ftp encontrará información adicional y el código fuente del programa:

```
ftp://info.cert.org/pub/tools/md5
```

Truco: no utilice el programa sum de UNIX para generar checksum, porque no es fiable.

Si los cambios no se deben a la actuación de una actividad autorizada, debería considerar la posibilidad de que alguien ha entrado en el sistema y tomar las acciones pertinentes. Si el cambio es autorizado, tendrá que actualizar la copia de seguridad.

¿Hay alguna esperanza?

En medio de una discusión sobre los peligros relacionados con la seguridad, es muy fácil deprimirse al considerar que no está nada clara la respuesta

a la pregunta que acabamos de formular. La verdad es que mi respuesta sería que, contando con las medidas necesarias y poniendo el suficiente cuidado, se puede llegar a ganar la batalla.

De acuerdo con Dr. John D. Howard, autor de ciertas publicaciones relacionadas con problemas de seguridad en Internet, a excepción de los ataques de negación de servicio, el número de este tipo de violaciones es cada vez menor. Su estudio se basó en los informes que crea el CERT, en los que se informa sobre los últimos incidentes. También comentó que la probabilidad de que el CERT no informe sobre un incidente está entre el 0 y 4 por ciento. Otra de las conclusiones que se obtenían de su informe es que la media de incidentes que sufre un dominio Web no es más que uno al año.

Tuvo en cuenta un total de 4.567 incidentes que se produjeron en un periodo de 7 años. De todos esos incidentes, el 5.9% corresponde a falsas alarmas. La mayoría (89.3%) eran incidentes de accesos no autorizados, de los cuales 27.7% se correspondían con accesos a cuentas root, el 24.1% a cuentas de usuarios y el 37.6% con intentos de acceso.

Con respecto al crecimiento del número de host en Internet, cada uno de estos tipos de acceso decreció durante el tiempo que duró esta investigación. Para cuando se completó el informe, los accesos a cuentas root eran un 19% menos que el crecimiento de los host en Internet, los accesos a cuentas de usuarios se habían reducido en un 11% y los intentos de acceso en un 17%.

Parte IV
Implementar propiedades avanzadas

13 Perl en Apache

Con el paso de los años, Perl se ha convertido en el lenguaje de programación preferido a la hora de desarrollar soluciones de programación para la Red. Principalmente se utiliza en el desarrollo de script CGI. Pero, por ser un lenguaje de interpretación, tiene una serie de problemas inertes. Su intérprete, perl, se tiene que cargar cada vez que se llama al script en Perl. Por eso el inicio de los script es algo más lento que el de las aplicaciones compiladas como binarios ejecutables.

En capítulos anteriores hemos visto cómo se puede utilizar FastCGI para acelerar los script basados en Perl. Otra forma de solventar el mismo problema podría ser utilizando el módulo mod_perl, fruto de la integración Apache/Perl, que ofrece alguna funcionalidad más que la aceleración de los script CGI. En este capítulo aprenderemos a utilizar mod_perl con el servidor Apache.

mod_perl

Uno de los objetivos principales de la integración Apache/Perl era conseguir que Perl fuese un potente lenguaje de programación dentro del servidor Apache. El resultado fue el desarrollo del módulo mod_perl. Se compila junto a Apache y Perl para que la API de C del servidor tenga una interfaz en Perl

orientada a objetos. Así, los programadores en Perl pueden escribir módulos para Apache. Se puede utilizar uno de estos módulos durante la ejecución del controlador, analizador de cabeceras, traducción de URI, autenticación, autorización, acceso, verificación de tipo, corrección, registro y limpieza de una petición.

Como ya hemos mencionado con anterioridad, cuando se utiliza el módulo mod_perl, tendrá en sus manos toda la potencia de una API en Apache. Vamos a ver cómo llega todo este poder a sus manos.

Cuando actualiza Perl e instala mod_perl en su sistema, crea una interfaz en Perl orientada a objetos para la API del servidor Apache. De acuerdo con ella, las peticiones de objetos contendrán toda la información que necesita el servidor para atender dicha solicitud. ¿Cómo se hace el script con el objeto que tiene que recoger? Tomando PerlHandler como parámetro. Por ejemplo:

```
<Location /perl/>
SetHandler  perl-script
PerlHandler Apache::Registry
Options ExecCGI
</Location>
```

En esta configuración, PerlHandler es el módulo Apache::Registry. Ha de contar con un método llamado handler, que recibirá la petición del objeto como una referencia. Así, el módulo podrá hacer cualquier cosa utilizando la interfaz API del servidor Apache.

Los módulos de Apache escritos en mod_perl pueden hacer lo mismo que los módulos escritos en C. Además, el intérprete persistente incluido dentro del servidor evita tener que llamar a un intérprete externo y que Perl tarde más tiempo en iniciarse. Por ejemplo, no es necesario iniciar un proceso separado con la misma frecuencia como se suele hacer con las extensiones del servidor. Los mecanismos de extensión, como CGI, se pueden sustituir por código Perl que se encarga de controlar la fase de generación de respuestas del proceso de peticiones. El módulo mod_perl incluye un módulo general que se encarga precisamente de esto (Apache::Registry) y que puede ejecutar los script CGI en Perl con total transparencia.

Ahora que ya sabe lo que puede hacer con el módulo mod_perl, le mencionaré los módulos que tiene a su disposición y cómo configurarlos.

Instalación de mod_perl

La mayoría de los módulos de Apache los desarrollan programadores repartidos por todo el mundo. Si quiere conocer cuáles son los últimos módulos

publicados, obtendrá la información pertinente en http://perl.apache.org/src/apache-modlist.html. Aquí tiene una lista con los módulos disponibles en la actualidad en mod_perl.

- Apache::Registry. Ejecuta script CGI sin modificar.
- Apache::Status. Muestra el estado del intérprete.
- Apache::Embperl. Incluye el código en Perl en los documentos HTML.
- Apache::SSI. Implementa SSI en Perl.
- Apache::DBI. Conserva las conexiones DBI persistentes.
- Apache::DCELogin. Obtiene un contexto DCE Login.
- Apache::AuthenDBI. Autentica a través de Perl DBI/DBD::*.
- Apache::AuthzAge. Autoriza basándose en la edad.
- Apache::AccessLimitNum. Limita el acceso del usuario a partir del número de peticiones.
- Apache::Constants. Constantes definidas en httpd.h.
- Apache::MsqlProxy. Transforma los URL en peticiones a las bases de datos mSQL.

El módulo mod_perl no se incluye en la distribución estándar de Apache. Tendrá que instalarlo. Pero antes debe asegurarse de cumplir los siguientes requisitos.

Requisitos de instalación

Necesitará el código fuente de Apache, de la última versión de mod_perl y Perl 5.003 (o superior). Si se quiere obtener la última versión de Perl visite www.perl.com. Si aún no lo tiene instalado, conviene que lo haga antes de seguir con el proceso.

Instalar Perl es tan sencillo como descomprimir el código y ejecutar el script Configure. El script le solicitará bastante información. Si no está seguro de las repuestas, deje la configuración predeterminada. Una vez que haya compilado la última versión de Perl utilizando el comando:

```
make install
```

conviene que compruebe el número de la versión con el comando:

```
perl -v
```

Si el número de la versión coincide con lo que acaba de compilar, ya lo tiene.

El código fuente de Apache lo encontrará en las páginas Web de Apache y el del módulo mod_perl, en http://perl.apache.com.

Cuando lo tenga, instale la fuente en el mismo directorio que el de Apache. Por ejemplo, si éste se encuentra en /usr/local/src/Apache, instálelo entonces en /usr/local/src de tal modo que termine con el directorio /usr/local/src/mod_perl_directory.

A continuación descomprima perl-1_10.tar.gz en el directorio /usr/local/src con el comando:

```
zcat mod_perl_1_10.tar.gz | tar xf -
```

De esta forma crea un directorio /usr/local/src/mod_perl-1.10, que será donde se guarde el módulo mod_perl.

Compilar e instalar mod_perl

Una vez que tiene mod_perl en su ordenador, está listo para compilarlo e instalarlo. Para compilar la fuente mod_perl, tendrá que modificar el directorio mod_perl y ejecutarlo:

```
perl Makefile.PL
```

Si lo ha instalado en el mismo directorio que contiene la fuente de Apache, tendrá que aceptar las respuestas predeterminadas a las preguntas que se le van formulando. De esta forma crea los archivos make tanto para mod_perl como para Apache. El siguiente paso que tiene que dar es ejecutar el comando make:

```
make
```

Una vez que se ha construido el binario de Apache, puede probarlo con el siguiente comando:

```
make test
```

Así se inicia el servidor Apache en el puerto 8529 bajo el UID y GID del proceso Makefile.PL de Perl. Si por cualquier razón el sistema ya estuviese utilizando este puerto, debería ejecutar el comando Makefile.PL indicándole un puerto:

```
perl Makefile.PL PORT=nnnn
```

Aquí, nnn es el número del puerto (mayor que 1024) que no se esté utilizando.

Si las pruebas tienen éxito, verá un mensaje similar a "All tests successful" (Todas las pruebas superadas con éxito). Ahora puede instalar el módulo mod_perl con el comando:

```
make install
```

Tendrá que sustituir su httpd por aquél que incluye la última versión de mod_perl, que se encuentra en el directorio fuente de Apache. Desde el directorio fuente de Apache tendrá que ejecutar el siguiente comando:

```
./httpd -v
```

Debería mostrarle el número de la versión del servidor Apache y del módulo mod_perl. Si no es así, inténtelo con el siguiente comando para ver si aparece mod_perl.c:

```
./httpd -l
```

Si no aparece en la lista de módulos, vuelva entonces a repetir el proceso de instalación.

Puede resultar conveniente volver a nombrar el archivo httpd existente como httpd.old antes de sustituirlo por la nueva versión. De esta forma, si algo fuese mal, podría volver a instalar la versión anterior sin ningún problema.

Ejecutar script CGI en Perl a través del módulo mod_perl

Ahora ya está listo para utilizar el módulo mod_perl. Lo primero que vamos a ver es cómo puede aprovechar mod_perl para acelerar los script CGI basados en Perl.

Es posible que recuerde que para ejecutar los script CGI tiene que definir un alias de un directorio a través de la directriz ScriptAlias. Pero cuando utilice mod_perl no podrá usar ScriptAlias. Tendrá que crear un alias corriente con la directriz Alias. Por ejemplo, para un sitio Web llamado apache.nitec.com en el que los script CGI se encuentren dentro del directorio /www/apache/public/cgi-bin/, puede crear un alias que se llame /perl/ de la siguiente manera:

```
Alias /perl//www/apache/public/cgi-bin/
```

Ahora, utilizando el contenedor <Location ...> tendrá que configurar lo siguiente:

```
<Location /alias-to-cgi-script/>
SetHandler  perl-script
PerlHandler Apache::Registry
Options ExecCGI
</Location>
```

Para nuestro ejemplo, el resultado sería el siguiente:

```
<Location /perl/>
SetHandler  perl-script
PerlHandler Apache::Registry
Options ExecCGI
</Location>
```

También debería tener una directriz ScriptAlias como sigue:

```
ScriptAlias /cgi-bin/   /www/apache/public/cgi-bin/
```

A continuación podría acceder a una aplicación llamada printenv.cgi a través del sistema tradicional con los CGI, es decir, utilizando un URL:

```
http://apache.nitec.com/cgi-bin/printenv.cgi
```

o podría utilizar el módulo Apache::Registry de mod_perl con el URL:

```
http://apache.nitec.com/perl/printenv.cgi
```

Es posible que se pregunte cómo se puede determinar la diferencia entre ambas llamadas. En la versión de mod_perl, la variable CGI estándar GATEWAY_INTERFACE imprime "Perl/1.1" en vez de "CGI/1.1". Además, en la versión de mod_perl se define una variable nueva, MOD_PERL. En ella se guarda el valor de la versión de mod_perl que se está utilizando. Por lo que, si necesita detectar cómo se ejecuta el script, puede utilizar el siguiente segmento de código:

```
If(exists $ENV{'MOD_PERL'}){
      print "Running as a mod_perl application";
      }
else{
      print "Running as a CGI script";
      }
```

Se utiliza el módulo Apache::Registry para ejecutar los script CGI con mod_perl. Cuando se solicita el siguiente URL, este módulo lee en primer

lugar el script printenv.cgi, lo inserta dentro del cuerpo de una subrutina en Perl y lo ejecuta:

```
http://apache.nitec.com/perl/printenv.cgi
```

Cada thread de los procesos de Apache compilará la subrutina una vez y la guardará en la memoria para utilizarla en el futuro. Si en ese transcurso de tiempo la subrutina cambia, se actualizará y la versión compilada se volverá a guardar en la memoria.

De todas formas, puede cargar de antemano sus script en Perl durante el inicio del servidor. mod_perl le ofrece dos directrices: PerlModule y PerlScript. Puede utilizar cualquiera de los dos para indicar qué módulos se cargarán durante el inicio del servidor.

Para cargar el módulo de Perl durante el inicio del servidor, puede utilizar lo siguiente:

```
PerlModule NombreMódulo . . .
```

Por ejemplo:

```
PerlModule Apache::SSI  Foo::Bar Algún::Otro
```

Este ejemplo carga los módulos SSI, Bar y de los módulos Apache, Foo y Algún. Utilizando la directriz PerlModule puede cargar hasta 10 módulos. Si necesita alguno más, utilice la directriz PerlScript.

La sintaxis de PerlScript es la siguiente:

```
PerlScript /path/to/a/Perl/script
```

Así se cargará un script de Perl durante el inicio del servidor. Si tiene que cargar varios módulos, puede escribir un script como éste:

```
use CGI;
use LWP::UserAgent ();
1;
```

Con él se cargan los módulos CGI y UserAgent. Si guarda los módulos en una dirección distinta a la predeterminada para los módulos de Perl, tendrá que especificarlo:

```
use lib qw(/path/to/your/modules);
```

En este caso habrá que sustituir /path/to/your/modules por el nombre completo del directorio en el que se guardan los módulos.

Para utilizar el módulo de Perl tendrá que emplear la directriz PerlHandler, como ya se ha visto. Por ejemplo, supongamos que tiene una línea PerlModule como sigue:

```
PerlModule Apache::Test
```

Y que quiere utilizarlo para atender el URL:

```
http://www.myserver.com/test/
```

Puede definir un alias de la siguiente manera:

```
Alias /test/      /somewhere/some/directory/
```

A continuación definirá el contenedor <Location ...>:

```
<Location /test>
SetHandler perl-script
PerlHandler Apache::Test
</Location>
```

Así le indicará a mod_perl que quiere utilizar el módulo Apache::Test para controlar todas las peticiones dirigidas al URL especificado.

¿Cómo puede saber mod_perl a qué módulo tiene que llamar en Test.pm? Por defecto, la directriz PerlHandler espera que se le entregue el nombre de la subrutina como argumento. Si éste no es el nombre de la subrutina, asume que es el nombre del módulo y llama a un método especial llamado handler. Así que, para la línea PerlHandler, cuando llega una petición URL especial, mod_perl llamará al método Apache::Test::handler. Si no se define este método en el módulo, aparecerá un error.

Se puede utilizar el nombre de un método distinto siempre que se especifique en la línea PerlHandler.

Por ejemplo, si tiene un método llamado doit en el módulo Test.pm y quiere llamarle a él en vez de utilizar handler, tendrá que configurar la línea PerlHandler de la siguiente manera:

```
PerlHandler Apache::Test::doit
```

De esta forma, mod_perl llamará a doit en vez de a handler.

Antes, cuando le he mostrado cómo se puede utilizar Apache::Registry para convertir los script CGI en script basados en mod_perl para cada petición mod_perl, se abría el método Apache::Registry::handler, que ejecuta la versión convertida del script.

Enviar a un puerto los script CGI en Perl

Cuando envía a un puerto los script CGI basados en Perl, tendrá que prestar atención a los siguientes puntos:

Si utiliza el intérprete de la última versión de Perl (5.004), es muy posible que la mayoría de los script CGI tengan que ejecutarse bajo mod_perl sin sufrir demasiadas modificaciones. De todas formas, si utiliza una versión antigua, ppuede que tenga algún problema a la hora de construir funciones de impresión y lectura. Por eso le sugiero que actualice su versión de Perl.

Por defecto, mod_perl no envía las cabeceras HTTP. Sin embargo, puede utilizar la siguiente directriz para modificarla.

```
PerlSendHeader On
```

Ahora se enviará la línea de respuesta y las cabeceras HTTP. Sin embargo, su script aún tendrá que enviar la siguiente cabecera:

```
print "Content-type: text/html\n\n";
```

Si trabaja con CGI.pm o CGI::Switch e imprime la cabecera utilizando el método query, header( ), no tendrá que activar la directriz anterior.

Por otro lado, cuando se convierten las cabeceras de los script en Perl sin analizar (tienen el prefijo nph-) a script mod_perl, tendrá que forzar una salida y limpiar después de escribir o imprimir en la salida del controlador de archivos. Para ello se puede ajustar $| a un valor distinto de cero.

```
local $| = 1;
```

Obsérvese que si PerlSendHeader estuviese activada (On), tendría que desactivar los archivos nph-. Puede utilizar la siguiente configuración para hacerlo.

```
<FilesMatch */nph-*>
PerlSendHeader Off
</FilesMatch>
```

Escribir un módulo en Perl para mod_perl

Ahora que ya está al corriente de los problemas relacionados con el uso de los script CGI en mod_perl, puede comenzar a desarrollar nuevos script en mod_perl. Para tratar de hacerlo lo más sencillo posible, vamos a ver un

módulo en Perl muy sencillo (un script en Perl orientado a objetos) que no hace nada más que imprimir el valor de las variables de entorno. Este módulo en Perl se llama Test.pm y su aspecto es el que aparece en el listado 13.1.

```
# Le dice a Perl que muestre un aviso cuando se utilice una
# variable sin una declaración. Para ello se usa 'use strict.'
use strict;

sub handler {
#
# Propósito: a este método lo llama mod_perl. En realidad es un
# punto de entrada a este módulo. En otras palabras, es una función
# principal.
#

# Le asigna el objeto solicitado (que se pasa como un parámetro) a
# la variable local.
    my $r = shift;

# Variable local temporal
    my $key;

# Asigna una array local asociativa a la conversión que devuelve el
# método cgi_env()
#
    my %ENV = $r->cgi_env;

# Ajusta el content-type de la salida a text/html
#
    $r->content_type("text/html");

# Imprime las cabeceras HTTP
    $r->send_http_header;

# Ahora salta a través de la array %ENV e imprime el par
# clave=valor de cada variable de entorno.
#
    foreach $key (keys %ENV){
      $r->print("$key = $ENV{$key} <BR>.");
      }

# hecho
    return 1;
}
# La siguiente línea la necesita el módulo en Perl.
1;
__FIN__
```

Listado 13.1. Test.pm

En este ejemplo tan sólo hay una subrutina llamada handler. Cuando se especifica una línea de configuración como ésta:

```
<Location /test>
SetHandler perl-script
PerlHandler Apache::Test
</Location>
```

se invocará al método handler siempre que Apache reciba una petición en la que aparezca /test/. Si tiene pensado jugar con el proceso de peticiones de Apache, asegúrese de que comprende el funcionamiento de la interfaz API de Apache. Hay varios estados de petición en los que una API de Apache permite que un módulo entre en funcionamiento y haga algo. En la documentación de Apache aprenderá algo más sobre estos puntos. Por defecto, en el momento de la compilación se desactivan todas estas propiedades. Para activarlas, consulte la información sobre INSTALL.

Los desarrolladores de programas CGI tienen en cuenta una serie de aspectos cuando desarrollan script en mod_perl. Antes de nada vamos a ver por qué requieren tal grado de programación, además de ciertos aspectos relacionados con su desarrollo.

Muchos desarrolladores CGI no pierden el tiempo depurando su código. La razón es muy sencilla: un script CGI se ejecuta durante un espacio de tiempo muy corto y luego desaparece. Pero los script en mod_perl se cargan en la memoria y permanecen allí durante mucho tiempo, por lo que habrá que tener más cuidado a la hora de escribirlos. Aquí tiene unos cuantos consejos que le pueden servir de guía.

A la hora de escribir un script convendrá hacerlo con strict. De esta forma el código resultante no tendrá variables sin definir, que lo único que hacen es consumir memoria y crear cierto grado de confusión. Tampoco olvide utilizar el parámetro -w. Si aún no tiene muy claro por qué hay que utilizarlo, le recomiendo que se lea la página de ayuda de Perl. Verá que no dejan de aconsejar su utilización. Este es un pequeño extracto de dicha página:

> *Siempre que se encuentre con un comportamiento extraño, ¡utilice el parámetro -w! Y siempre que el comportamiento sea correcto, repita de todas formas el proceso utilizando el parámetro -w.*

La verdad es que su uso no es obligatorio, pero sí muy aconsejable. Le ahorrará bastante tiempo durante la ejecución.

Tiene que ser muy cuidadoso a la hora de trabajar con los datos introducidos por el usuario. Puede utilizar el parámetro -T para que Perl efectúe una comprobación, o activar la directriz PerlTaintCheck (añadiendo On) que se encuentra en los archivos de configuración de Apache. Lea la página de ayuda cgi_to_mod_perl que se incluye en la distribución de mod_perl.

Observe que las llamadas del sistema como system( ), exec( ), open PIPE, |program, etc. no funcionan por defecto. Si quiere trabajar con ellas, su versión de Perl tendrá que soportar sifo. Además, la función exit( ) tampoco se puede utilizar. Como alternativa puede usar el método Apache::evit( ), que sustituye automáticamente la llamada exit( ) de los script Apache::Registry.

Aparte de estas anotaciones generales, hay una serie de puntos relacionados con la programación que tendrán que tenerse en cuenta a la hora de desarrollar los script para mod_perl. Los veremos en las próximas secciones.

Uso de los módulos CGI.pm y CGI::*

Si utiliza el módulo CGI.pm en sus script CGI, tendrá que asegurarse de que posee la última versión. Todas las anteriores a la 2.36 no funcionan con mod_perl. Si tiene la última versión de Perl y la de CGI.pm. puede utilizar el comando CGI para servirse como siempre de los módulos desde los script CGI. De todas formas, si utiliza una versión antigua de Perl, tendrá que emplear CGI::Switch( ) para manipular I/O a través de los métodos de Apache.

Los otros módulos CGI (CGI::*) tan sólo se pueden utilizar con la última versión de Perl (5.004 o superior).

Nota: si quiere utilizar el módulo (CGI::*) y el método SendHeaders( ), asegúrese de llamar a $req_obj, cgi, done cuando termine con la petición.

Si utiliza CGI::Switch o CGI.pm y la última versión de Perl, su script sólo trabajará desde la línea de comandos.

Uso de los módulos de autenticación DB/DBM de Apache

Si utiliza los módulos mod_auth_dbm o mod_auth_db se encontrará con problemas, porque las librerías dbm que están dentro del archivo Config.pm de Perl no son las adecuadas. Para determinar el orden de las librerías, tendrá que utilizar el siguiente comando:

```
Perl -V:libs
```

Por defecto, estas librerías están enlazadas con Perl y esta asociación se determina en el módulo Config. Cuando se configura mod_perl con Apache,

el módulo ExtUtils::Embed se encarga de preparar los enlaces con httpd para que funcionen correctamente las extensiones de Perl bajo mod_perl. De todas formas, el orden en el que se almacenan estas librerías en Config.pm puede llegar a confundir a Apache DB/DBM. Si aparece -lbdbm o bien -ldb antes de -lndbm, por ejemplo:

```
libs='-lgdbm  ldb -lndbm -ldl -lm -lc'
```

se modifica el módulo de Config.pm de Perl y se mueve -lgdbm y -ldb al final de la lista. Si no tiene muy claro qué modificación tiene que hacerle a Config.pm, utilice el siguiente script para determinar la ubicación:

```
#!/usr/local/bin/perl
use Config;
print "El directorio de Config.pm es" $Config{archlibexp}\n";
exit 0;
```

La interfaz mod_perl es lo suficientemente inteligente como para detectar los cambios del script después de que se produzca su carga. De esta forma, si modifica un script después de cargarlo a través de mod_perl, no tendrá que volver a iniciar Apache. Pero si los cambios los efectúa en una librería Perl o en uno de los módulos con los que trabaja un script gracias a require, mod_perl no volverá a cargarlo automáticamente. En este caso tendrá que utilizar Apache::StartINC.

Otra nota importante que hay que tener en cuenta es que cuando se incluye la misma librería Perl (no un módulo Perl) en varios script que desea que ejecute mod_perl a través de Apache::Register, tan sólo funcionará uno de ellos. Esto se debe a que cuando la librería requerida se carga en el script, sus subrutinas y variables pasan a formar parte del paquete actual. En un entorno CGI típico, éste pasará a ser el paquete principal bajo el que se compila el script. Es decir que solamente un script será capaz de utilizar la librería requerida. Mod_perl no compila la misma librería para el resto de los script que la solicitan.

La forma más sencilla se resolver este problema es volver a escribir la librería en un módulo en Perl orientado a objetos. Como el módulo en Perl tiene una definición de paquetes que le da su mismo nombre, el módulo solicitado (que se utiliza para convertirlo en una librería en Perl) se puede usar desde cualquier script.

Lea las páginas de ayuda de perlmodlib y perlmod, así como la documentación relacionada con perl, y vuelva a convertir los archivos en el módulo de Perl que define una interfaz. De todas formas, si no se puede volver a escribir la librería para convertirla en un módulo en Perl orientado a objetos, conviene

que mire en las funciones de exportación y las propiedades de las variables de Perl.

Conviene destacar que, si quiere utilizar un módulo que suele estar vinculado con Perl, éste ha de aparecer en static_ext de Config.pm para enlazarlo con httpd durante la construcción del módulo mod_perl.

Este módulo también restringe el uso de objetos especiales (nombres). Como ya hemos comentado, los script pertenecientes a Apache::Registry no se ejecutan en el paquete predeterminado (principal). El espacio del nombre que se utiliza se basa en el URI solicitado. Por lo tanto, estos script no pueden contener objetos especiales como __END__, que se emplea para especificar el final de un script. Tampoco se utilizará __DATA__

Integración de SSI y mod_perl

Hasta ahora hemos hablado de cómo se pueden desarrollar script CGI que aprovechen las ventajas de mod_perl. Pero, ¿qué hay de los script en Perl que utiliza como script SSI?. Ellos también pueden aprovechar las propiedades de mod_perl.

Mod_perl se puede integrar con el módulo SSI mod_include. Todo lo que tiene que hacer es asegurarse de que cuando construye el binario de Apache, el archivo de configuración cuenta con la siguiente línea:

```
EXTRA_CFLAGS=-DUSE_PERL_SSI  I. `perl -MExtUtils::Embed -ccopts`
```

o cuando construye mod_perl, asegúrese de que utiliza el siguiente comando:

```
perl Makefile.PL PERL_SSI=1
```

Una vez que ha construido Apache, puede aprovechar las ventajas de mod_perl con sus script SSI en perl. En vez de usar el comando SSI de la siguiente manera:

```
<!-#exec cmd="/some_directory_alias/your_perlscript" ->
```

tendrá que utilizar el siguiente en su página SSI:

```
<!-#perl sub="Apache::Include" arg="/some_directory_alias/
your_perlscript" ->
```

De todas formas, tendrá que usar la siguiente línea en el archivo de configuración httpd.conf para cargar de antemano el script:

```
PerlScript /path/to/your_perlscript
```

El valor de la subclave que aparece en el ejemplo anterior puede ser el nombre de una subrutina, el nombre de un módulo, el nombre del método de una clase o una llamada anónima a una subrutina (sub {}). Por ejemplo:

```
<!-#perl sub-"Apache::Test" ->
```

Con esta línea se invoca, por defecto, al método Test::handler siempre que Test.pm se cargue de antemano con PerlModule o bien durante el inicio del servidor.

Si quiere llamar a otro método que no sea el handler predeterminado, deberá especificar su nombre. Por ejemplo, si quiere llamar al método doit en ver de handler, cambie la llamada SSI anterior por:

```
<!-#perl sub="Apache::Test::doit" ->
```

Si hay que pasar los argumentos por encima del método, tendrá que usar:

```
<!-#perl sub="Apache::Test::doit" arg="argumento1"
arg-"argumento2" ->
```

para pasar "argumento1" y "argumento2" como los argumentos del método. Obsérvese que el primer argumento siempre se solicitará como referencia al objeto.

Uso de Perl para configurar Apache

Aparte de la ventajas que supone tener mod_perl a disposición de los script CGI o SSI, hay otra característica de la que se pueden aprovechar los programadores de Perl y los administradores de Apache: la capacidad de escribir la configuración del servidor Apache en Perl.

Puede incluir un contenedor <Perl> dentro del archivo de configuración:

```
<Perl>
# Aquí aparecerá el código en Perl
1;
</Perl>
```

La penúltima línea (en la que aparece el 1;) es completamente necesaria. Su código puede ser cualquier script en Perl. El código que aparece dentro del contenedor <Perl> se compila en un paquete especial y mod_transfiere la información sobre la comunicación al módulo de configuración del núcleo de Apache. Antes de que se pueda utilizar esta propiedad, deberá cerciorarse de

que tanto Apache como mod_perl funcionan correctamente. Para configurar esta propiedad para su servidor Apache, tendrá que preparar mod_perl de la siguiente manera:

```
perl Makefile.PL PERL_SECTIONS=1
```

Una vez que ha configurado y compilado de nuevo mod_perl y Apache, está listo para escribir el código Perl dentro de los archivos de configuración del servidor. A continuación veremos la sintaxis que se ha de utilizar para describir la configuración.

Esta directriz toma un único valor que se representa como variables escalares. Por ejemplo:

```
User httpd
```

La directriz User toma el valor de una cadena, por lo que se puede escribir como:

```
<Perl>
$User = "httpd";
1;
</Perl>
```

Aquí tiene un ejemplo de la configuración:

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

Las directrices que necesitan varios valores se pueden representar como listas. Por ejemplo:

PerlModule Apache::TestOne Apache::TestTwo se puede representar como:

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

Los contenedores se representan usando convertidores hash, por ejemplo:

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

se puede representar como sigue:

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

Un ejemplo algo más amplio sería:

```
$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'
        }
 };
```

Ahora puede definir otros contenedores como <Directory>, <Files>, etc. de forma similar. Obsérvese que para que funcionen los contenedores <Perl> tiene que definir la directriz PerlScript en su archivo de configuración. Si no tiene que cargar ningún script en perl, para aprovechar las ventajas de los contenedores <Perl> tendrá que determinar la directriz PerlScript de la siguiente manera:

```
PerlScript /dev/null
```

En mi opinión, convendrá utilizar los contenedores <Perl> en los archivos de configuración sólo si se cumple uno de estos dos requisitos:

- Le gusta tanto trabajar con Perl que no podría imaginarse un archivo de configuración de Apache en el que no apareciesen los contenedores <Perl>.
- Trabaja con varios servidores Apache y quiere crear un único conjunto de archivos de configuración para utilizarlo con todos los sistemas de servidores de Apache. Un script de configuración es muy flexible. Puede leer archivos externos, bucles para automatizar la configuración, etc. Este tipo de configuración tan sólo está limitado por la habilidad que tenga el administrador de Apache para programar.

Si opta por utilizar el contenedor <Perl> por la primera razón, tendría que advertirle que ponga todo su empeño para que los script tengan la mejor apariencia posible. De todas formas, si quiere utilizarlos por la segunda razón y

no termina de tener muy claro cómo puede aprovechar esta propiedad, vea el ejemplo que le mostramos a continuación.

Supongamos que tenemos tres sistemas con servidores Web: host_a, host_b y host_c. Y que los hemos numerado por orden decreciente de potencia. Quiere definir un contenedor <Perl> que permita crear un solo fichero de configuración para los tres host. Este sería su aspecto:

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

Para hacer que el ejemplo sea más interesante, vamos a hacer que cada host sea un servidor virtual distinto y a utilizar una configuración algo más elegante. Observe este ejemplo:

```
<Perl>
# Toma el nombre del host de UNIX y lo guarda en la variable
# $thisHost.

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

my $anyHost;

if ($thisHost =~ /(host_a)/) {
  # aquí aparecería la configuración de host_a
```

```
    @vHosts = qw(gaia, athena, romeo, juliet, shazam);
   }
elsif ($thisHost =~ /host_b/) {
   # aquí aparecería la configuración de host_b
    @vHosts = qw(catbart, ratbart, dilbert);
   }
else {
   # aquí aparecería la configuración de host_c
    @vHosts = qw(lonelyhost);
   }

for $anyHost (@vHosts) {
    ${$VirtualHost{"$anyHost.$domainName"}} = {
      "ServerName" => "$anyHost.$domainName",
      "ServerAdmin"  > "webmaster\@$anyHost.$domainName"
            }
    }
1;
</Perl>
```

Truco: una vez que ha creado una configuración basada en Perl para sus servidores Apache, puede comprobar la sintaxis del código para asegurarse de que es correcta. Utilice el siguiente comando:

```
perl -cx httpd.conf
```

Puntos especiales relacionados con mod_perl

Hasta ahora hemos visto lo maravilloso que es mod_perl y cómo aprovechar sus ventajas. A continuación nos vamos a centrar en un par de aspectos con los que hay que tener cuidado: los recursos y la seguridad del sistema.

Uso de memoria

Mod_perl hace que la ejecución de sus script en Perl sea más rápida, ya que se aprovecha del intérprete que se encuentra presente en la memoria y de los script que se guardan en la caché. Todo esto está muy bien, pero ¿dónde está el problema? Pues que al utilizar mod_perl se emplea una gran cantidad de memoria. Generalmente cuando se ejecuta un script CGI de la forma tradicional, se cierra y desaparece después de cumplir su cometido. Pero en la versión mod_perl el script se queda residiendo en la memoria. Es decir, que cuantos más script quiera utilizar con mod_perl, mayor será la cantidad de memoria que se consumirá.

Si su script en Perl utiliza muchos módulos comunes, puede cargarlos de antemano utilizando un único script en Perl. Por ejemplo, el código que aparece a continuación se corresponde con el del script llamado loader.pl que carga el módulo CGI.pm.

```
#!/usr/local/bin/perl
# startup script
use lib qw(/www/perl/modules);
use CGI;
1;
```

Para cargar de antemano este módulo, tendrá que usar la directriz PerlScript:

```
PerlScript /path/to/loader.pl
```

En la mayoría de los sistemas operativos modernos, el thread del proceso de Apache que ejecuta mod:perl será capaz de compartir CGI.pm.

En cualquier caso, cuantos más script cargue, tanto más memoria necesitará. Esto puede llegar a convertirse en un problema si sus recursos son limitados y tiene que prestar atención a la cantidad de memoria que se emplea. Con ciertas utilidades de UNIX (tales como ps o top) podrá visualizar la cantidad de memoria consumida. La verdad es que yo, con Linux, utilizo el siguiente comando para generar una lista con la siguiente información relacionada con httpd:

```
ps auxw | awk '{print $1, "\t", $2 , "\t", $5 , "\t", $6}' |
egrep '(httpd|SIZE)'
```

Esta sentencia se encarga de imprimir USER, PID, SIZE (tamaño de la imagen virtual, del texto, de los datos y del stack) y RSS (tamaño, medido en kilobytes, que ocupa el programa que se ha cargado en la memoria) de todos los procesos de httpd.

También puede ver qué paquetes y script en Perl ha cargado mod_perl en la memoria caché utilizando Apache::Status. De todas formas, antes de que pueda utilizar este módulo, tendrá que instalar Devel::Symdump. El módulo de Perl proporciona un medio para inspeccionar la tabla de símbolos de Perl y la jerarquía de clases que se utiliza en los programas. Si tiene instalado el módulo CPAN.pm, puede instalar el programa:

```
perl -MCPAN -e 'install "Devel::Symdump"'
```

o bien puede hacerse con el código fuente visitando el URL:

```
www.perl.com/cgi-bin/cpan_mod?module=Devel::Symdump
```

La instalación del módulo es muy sencilla. Todo lo que tiene que hacer es leer el archivo README que se distribuye con él. Una vez que instale el módulo Devel::Symdump, podrá conseguir información sobre los módulos cargados y sus tablas de símbolos. Defina el siguiente segmento en uno de sus archivos de configuración de Apache:

```
<Location /perl-status>
SetHandler  perl-script
PerlHandler Apache::Status
</Location>
```

y reinicie el servidor Apache. Puede hacerse con la información de estado visitando un URL como:

```
http://yourserver/perl-status
```

Seguridad

Los script que ejecuta mod_perl tienen los mismos privilegios que los del usuario y grupo especificados en las directrices User y Group del archivo httpd.conf. Asegúrese de que los permisos de user/group son muy limitados.

Tenga cuidado al desarrollar los script que recogen datos introducidos por los usuarios, para que comprueben las entradas antes de utilizarlas. Si quiere que mod_perl efectúe una comprobación, tendrá que agregar la siguiente directriz en el archivo httpd.conf:

```
PerlTaintCheck On
```

El parámetro -T se suele utilizar para activar la comprobación del módulo mod_perl.

Otro Perl para Apache

Una vez que hemos terminado con mod_perl, me gustaría comentarle un par de formas adicionales con las que puede utilizar Perl en Apache. Hay muchas, pero son dos las que realmente merecen la pena.

De acuerdo con su autor, Ralf S. Engelschall, ePerl es el primer intérprete de Perl incluido en Apache. Es un programa independiente que puede analizar y procesar los script de Perl que se encuentren dentro de un archivo de texto plano en ASCII. Se suele utilizar en la generación de filtros de contenido, aunque es un buen candidato para el lenguaje SSI. El programa ePerl es

un intérprete de Perl completamente funcional que utiliza la librería libperl.a, por lo que, al permanecer el lenguaje sin variar, no hay que aprender nada nuevo. Con ePerl puede utilizar todos los módulos que suele usar con el intérprete corriente de Perl. Incluye los objetos compartidos que se emplean con la facilidad DynaLoader. Se puede bajar el software de la página Web:

```
www.engelschall.com/sw/eperl/
```

Cuando lo tenga, siga las instrucciones que se incluyen en el paquete para instalar ePerl en su sistema. Si únicamente va a utilizar ePerl como SSI, no tendrá que instalar ePerl como un programa independiente. Como alternativa, puede instalar los módulos de Perl 5 dentro de los directorios /path/to/perl/lib/site_perl/{Parse,Apache}/ del sistema de Perl. Ejecute los siguiente comandos desde el directorio fuente de ePerl:

```
perl Makefile.PL
make
make test
make install
```

Instale el módulo Apache::ePerl que hace falta para ejecutar ePerl como SSI. Para comprobar si la instalación ha tenido éxito, copie el ejemplo que se encuentra en el directorio del script (eg) en su árbol de documentos Web y efectúe la siguiente configuración en el archivo httpd.conf:

```
Alias /eperl/  /path/to/eg/directory dentro del árbol de documentos
Web/
<Location /eperl/>
SetHandler perl-script
PerlHandler Apache::ePerl
Options ExecCGI
</Location>
```

Reinicie Apache para asegurarse de que el servidor Web puede leer el directorio en el que se encuentra en el script ejemplo. Ya puede acceder al script ejemplo de ePerl. Por ejemplo, uno de los script ejemplo que suministra el autor aparece en el listado 13.2.

```
<!-
##
##  demo.env - ePerl demonstration webpage
##  Copyright (c) 1996,1997 Ralf S. Engelschall, All Rights
##  Reserved.
##
->
<html>
```

```
<head>
<title>demo.env</title>
</head>
<body>
<blockquote>
<blockquote>
<h1>demo.env</h1>
<h2>Standard CGI Example: Environment</h1>
<p>

This prints out the CGI environment provided by the Web server as
a sorted list consisting of key/value pairs.

<p>
<pre>
<? my $key;
   foreach $key (sort(keys(%ENV))) {
       print "$key=$ENV{$key}\n";
   }
!>
</pre>

</blockquote>
</blockquote>
</body>
</html>
```

Listado 13.2. demo.env.phtml

Con este script obtendrá la salida que se muestra en la figura 13.1.

Como se puede ver en el listado 13.2, el código Perl se encuentra dentro del archivo HTML y contiene un par etiquetas especiales:

```
<?
   # perl code goes here
!>
```

Como ya hemos comentado, puede tener código en Perl válido en las páginas SSI basadas en ePerl. Uno de los archivos ejemplo le muestra cómo usar el módulo Net::FTP para obtener un archivo que se encuentre en un servidor remoto y ofrece una página a sus visitantes. Hay otros muchos ejemplos a los que conviene que eche un vistazo.

Otra de las soluciones que merece la pena destacar es embPerl, de Gerald Richter. Se puede obtener en:

```
ftp://ftp.dev.ecos.de/pub/perl/embperl/
```

EmbPerl y ePerl son muy parecidos. La principal diferencia es que ePerl no es específico de HTML, mientras que embPerl sí lo es. EmbPerl tiene una

serie de metacomandos especializados en el desarrollo de ciertas tareas, como rellenar tablas HTML. También interpreta etiquetas HTML. Puede llegar a ser útil en algunas ocasiones, pero la verdad es que aprender una nueva sintaxis puede ser demasiado. Si tiene que trabajar con este tipo de programas, mi consejo es que se decante por ePerl.

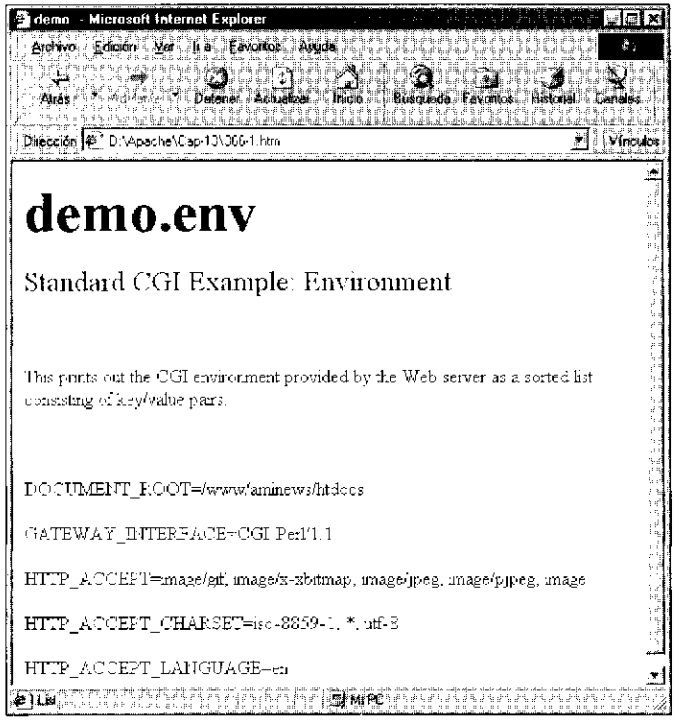


Figura 13.1. Salida de demo.env.phtml

14 Servidor proxy en Apache

Un servidor proxy es un sistema que se encuentra entre el host del cliente y los servidores a los que tiene que acceder. Cuando un cliente solicita un recurso a través de un URL, el servidor proxy recibe la solicitud y busca el recurso solicitado para atender al cliente. En un sentido general, un servidor proxy actúa como un servidor frente al cliente y como un cliente frente a los servidores remotos. En los típicos escenarios proxy, con este proceso se permite que el servidor almacene el material solicitado en la memoria caché, con lo que se ahorra tener que mantener la conexión con el servidor lejano. En vez de eso, los datos que se sirven son los que se encuentran en la memoria del servidor. De esta forma, los servidores proxy alivian el efecto de cuello de botella. De todas formas, un servidor proxy puede hacer más cosas.

En este capítulo aprenderá cómo convertir Apache en un servidor proxy capaz de efectuar una serie de servicios.

Servidores proxy

Antes de hablar sobre la forma de utilizar Apache como un servidor proxy, vamos a ver algo más sobre ellos y cómo utilizarlos. Hay dos tipos de servidores proxy: los directos y los inversos.

Proxy directo

Un servidor proxy directo se encuentra dentro el host del usuario y los recursos remotos a los que se quiere acceder. Los recursos pueden encontrarse en Internet o en una intranet (como se puede apreciar en la figura 14.1). La finalidad de un servidor proxy es localizar los recursos solicitados que se encuentran en un servidor remoto, entregárselos al usuario que lanza la petición y volcarlo a su disco duro. A la siguiente solicitud que pida lo mismo se le entregarán los datos que se encuentran en la memoria caché del sistema.

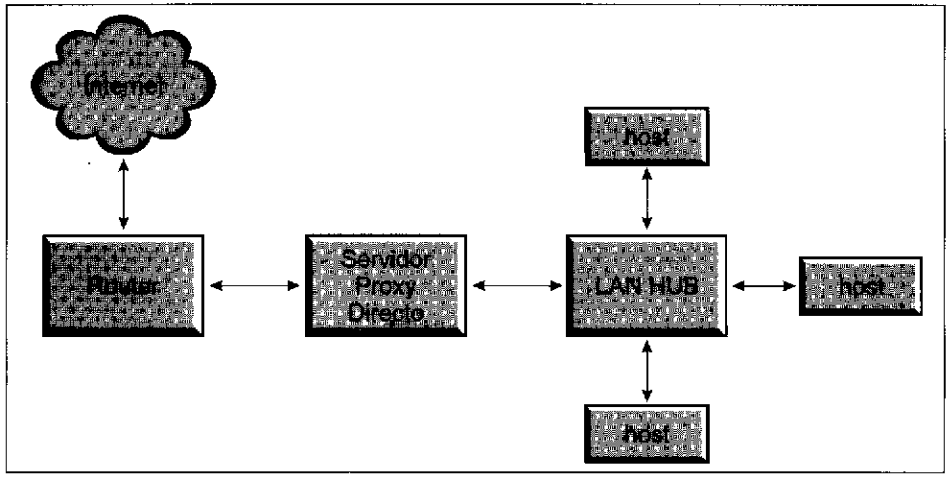


Figura 14.1. Servidor proxy directo

El host del usuario sabe que está trabajando con un servidor proxy porque ha sido configurado para ello. Es decir, que antes de usar un servidor proxy tendrá que instruir al explorador Web para que trabaje con él. Las peticiones remotas se atienden a través del servidor proxy, con lo que se obtiene una solución más barata y se reduce el uso de ancho de banda. También se puede implementar una política de acceso.

A este tipo de servidor también se le conoce como servidor proxy de caché. El inverso también guarda una copia de los datos en la memoria caché, pero funciona exactamente al contrario que el proxy directo.

Proxy inverso

Este tipo de servidor se encuentra frente a un recurso de Internet, tal y como se puede apreciar en la figura 14.2, o frente a una intranet. En este tipo

de configuración, el servidor proxy inverso recupera las peticiones provenientes del servidor original y las devuelve al host del usuario.

Los host del usuario que están conectados con un servidor proxy no saben que están conectados a uno de estos servidores en vez de estarlo directamente con el recurso. Esta es una de las principales diferencias que tienen con los proxy directos. El usuario creerá que está accediendo directamente al recurso solicitado.

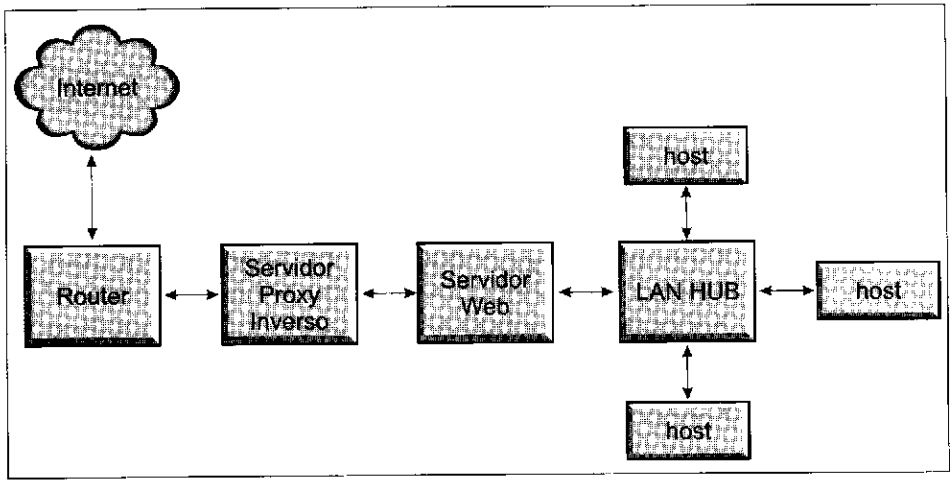


Figura 14.2. Servidor proxy inverso

Por ejemplo, si el sitio Web www.csus.edu utiliza un servidor proxy, los estudiantes tendrán que dirigir su explorador Web hacia www.csus.edu y no tendrán que configurar nada sobre el uso de los servidores proxy. El explorador enviará su petición al servidor www.csus.edu. No sabe que en realidad dicho servidor es un proxy que traduce la petición al servidor que tiene el recurso solicitado. ¿Qué se gana con este tipo de configuración? Como los datos que se envían se guardan en la memoria caché, el rendimiento que pueden llegar a dar los servidores proxy es mayor que el de los servidores normales.

Nota: en la actualidad Apache no puede trabajar con los servicios proxy inversos, pero la situación cambiará en el momento en que se publique la siguiente versión.

Ahora que ya lo sabe todo sobre los servidores proxy, vamos a pasar a considerar si su red necesita o no un servidor proxy.

¿Conviene utilizar un servidor proxy?

Los servicios proxy son escenarios ideales en los que más de un usuario accede a la red. Muchas organizaciones tienen varios ordenadores que acceden a Internet a través de una única conexión, como un router RDSI o una conexión dedicada. Un servidor proxy le puede resultar útil en este tipo de redes.

Al utilizar un proxy se podrá aprovechar de las siguientes ventajas, tanto si trabaja con recursos en Internet o en intranet:

- Proxy. Si la red interna utiliza direcciones IP que no se pueden distribuir con un router, ya sea por razones de seguridad o por cuestiones económicas, puede utilizar un servidor proxy para entregar a los host los recursos de Internet a los que no podrían acceder de cualquier otra forma. En este capítulo aprenderá cómo hacerlo.
- Caché. Con un servidor proxy como Apache (con mod_perl) podrá acceder a los recursos de Internet con más velocidad que a los locales. De esta forma mejora la eficiencia de la red y reduce el uso el ancho de banda.
- Registro y control de acceso. Con un servidor proxy podrá registrar todos los accesos que efectúen, por ejemplo, los estudiantes de una universidad para acceder a Internet (o incluso a una intranet). Se puede bloquear el acceso a ciertos sitios Web para proteger su compañía de problemas legales y evitar que sus empleados pierdan el tiempo. Al analizar el acceso y errores del servidor proxy puede identificar los modelos de uso y basarse en esta información para mejorar sus políticas de acceso.

Como se puede ver, el uso de un servidor Web aporta muchos beneficios. Vamos a ver cómo se puede transformar un servidor Apache en uno proxy.

Apache como servidor proxy

El servidor proxy con el que puede trabajar Apache se encuentra dentro del módulo mod_proxy. Por defecto no se compila. Tendrá que agregarlo al archivo Configuration, que se encuentra en el directorio fuente de Apache, y repetir la compilación del servidor. En la actualidad únicamente incluye un servidor proxy directo. Puede trabajar con los protocolos HTTP (HTTP/0.9 y HTTP/1.0), HTTPS (a través de CONECT para SSL) y FTP. También se

puede configurar el módulo para que se conecte con otros módulos proxy de esos y otros protocolos. Incluye las siguientes directrices.

ProxyRequest

```
Sintaxis: ProxyRequest On | Off
Predeterminado: ProxyRequest Off
Contexto: configuración servidor, host virtual
```

Esta directriz le permite activar o desactivar el servicio de proxy caché. De todas formas, no afecta al funcionamiento de la directriz ProxyPass.

ProxyRemote

```
Sintaxis: ProxyRemote <match> <URL-servidor-proxy-remoto>
Predeterminado: ProxyRequest Off
Contexto: configuración servidor, host virtual
```

Esta directriz permite que el servidor proxy interactúe con otro. El valor de match puede ser uno de los siguientes:

- El nombre de un URL con el que puede trabajar un servidor remoto.
- Una parte de un URL que puede utilizar el servidor remoto.
- Para indicar con qué servidor se ha de contactar para tratar todas las peticiones.

<URL-servidor-proxy-remoto> puede ser http://URL-servidor-proxy-remoto:puerto. Obsérvese que, en la actualidad, únicamente se puede trabajar con el protocolo HTTP. De todas formas, se pueden enviar peticiones FTP desde su servidor proxy a otros que puedan trabajar con HTTP y FTP. Simplemente tendrá que hacer lo siguiente:

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

Con esta sentencia se enviará una petición FTP al servidor proxy local para que la redirija a ftp://ftp.proxy.nitec.com. Como la petición se envía a través del protocolo HTTP, la transacción FTP tendrá lugar en el servidor remoto.

Si todo lo que desea es enviar una petición directamente al servidor Web de cierto sitio Web, puede utilizar esta directriz. Por ejemplo:

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

Así enviará todas las peticiones que coincidan con www.bigisp.com al servidor web-proxy.bigisp.com. Pero si las quiere enviar a otro servidor, tendrá que utilizar el asterisco (*):

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

Esto enviará todas las peticiones proxy locales al servidor proxy que se encuentra en proxy.domain.com.

ProxyPass

```
Sintaxis: ProxyRemote <URL relativo> <URL destino>
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

Esta directriz le permite convertir el árbol de documentos de un servidor Web en el de su servidor proxy. Por ejemplo:

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

Esta directriz se encuentra en archivo httpd.conf de un servidor proxy llamado proxy.nitec.com y permite que todos los usuarios utilicen el servidor proxy para acceder al sitio Web de Microsoft utilizando el URL:

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

Funciona como el mirror de un sistema Web remoto. Cualquier petición que utilice <URL-relativo> se convertirá internamente en el <URL-destino>.

Nota: si el sitio remoto incluye referencias absolutas, es posible que las imágenes no aparezcan y que los enlaces no funcionen. Además, hay que destacar que no se puede usar esta directriz con los servidores SSL.

ProxyBlock

```
Sintaxis: ProxyBlock <nombre parcial o completo del host> ...
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

Esta directriz le permite bloquear el acceso a un host o dominio cuyo nombre se indique. Por ejemplo:

```
ProxyBlock gates
```

bloqueará el acceso a cualquier servidor que tenga gate en su nombre. De esta forma, no se podrá acceder a ninguno de los servidores http://gates.ms.com o http://gates.friendsofbill.com. También se pueden hacer especificaciones múltiples como:

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

Se bloquea el acceso a cualquier host que tenga cualquiera de las palabras especificadas en su nombre. El módulo mod_proxy trata de determinar las direcciones IP de dichos servidores durante el proceso de arranque y los guarda en la memoria para utilizarlos en el futuro.

Para bloquear el acceso a todos los host, deberá utilizar:

```
ProxyBlock *
```

Con ello se desactiva su servidor proxy.

NoProxy

```
Sintaxis: NoProxy <Nombre dominio | Subred | Dirección IP
Nombre host>
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

Esta directriz le proporciona cierto control sobre la directriz ProxyRemote en una intranet. Puede especificar el nombre del dominio, una subred, dirección IP o el nombre de un servidor con el que no trabajará el servidor proxy desde la directriz ProxyRemote. Por ejemplo:

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

Aquí, todas las peticiones de <algo>.yourcompany.com (como www.yourcompany.com) se atenderán en el proxy local, mientras que el resto se mandará a firewall.yourcompany.com.

ProxyDomain

```
Sintaxis: ProxyDomain <Dominio>
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el nombre del dominio predeterminado del servidor proxy. Cuando esta directriz se configura para que trabaje con el nombre

de un dominio local o con una intranet, cualquier petición que no incluya el nombre de un dominio, utilizará el predeterminado. Por ejemplo:

```
ProxyDomain      .nitec.com
```

Cuando un usuario de .nitec.com envía una petición de un URL tal como http://marketing/us.html, la petición se volverá a regenerar con este URL:

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

Obsérvese que el nombre de dominio que especifique ha de comenzar por un periodo.

CacheRoot

```
Sintaxis: CacheRoot <directorio>
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

Esta directriz le permite activar la caché de disco. Puede especificar el nombre del directorio en el que desea guardar la copia de los archivos. El servidor Apache que ejecute un módulo proxy tiene que disponer de permiso de escritura. Por ejemplo:

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

De esta forma le indicará a Apache que escriba los datos de la caché en el directorio /www/proxy/cache. Recuerde que, antes de iniciar el servidor proxy, tiene que especificar el tamaño de la caché utilizando la directriz CacheSize. Es posible que tenga que usar otras directrices relacionadas con la caché (las veremos más tarde) para crear un disco útil para utilizar como memoria caché.

CacheSize

```
Sintaxis: CacheSize <n kilobytes>
Predeterminado: CacheSize 5
Contexto: configuración servidor, host virtual
```

Esta directriz especifica la cantidad de espacio de disco (en K) que se utilizará para las operaciones de caché. Los archivos se guardan en el directorio especificado con la directriz CacheRoot. Es posible que el servidor grabe archivos cuyo tamaño sea mayor que el destinado para la caché, pero en el momento en que los sirva al cliente, los borrará para liberar espacio de disco.

La configuración predeterminada es de 5K, algo completamente nimio. Mi consejo es utilizar entre 10MB y 1GB, dependiendo de la carga con la que suela trabajar.

CacheGcInterval

```
Sintaxis: CacheGcInterval <n horas>
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el tiempo (en horas) que transcurre hasta que el servidor revisa los directorios de la caché para borrar los archivos que hayan caducado. Apache los revisará cuando haya trabajado con un archivo de mayor tamaño que el especificado en CacheSize.

CacheMaxExpire

```
Sintaxis: CacheMaxExpire <n horas>
Predeterminado: CacheMaxExpire 24
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el tiempo (en horas) que transcurre hasta que caducan todos los documentos. En otras palabras, si se especifica como:

```
CacheMaxExpire 48
```

todos los documentos expirarán en un plazo de 48 horas (dos días). Esta directriz anula cualquier fecha de caducidad especificada en el propio documento. Así que, si la fecha de expiración de un documento es posterior que la especificada con esta directriz, se borrará cuando se cumpla la determinada por ésta.

CacheLastModifiedFactor

```
Sintaxis: CacheLastModifiedFactor <número de coma flotante>
Predeterminado: CacheLastModifiedFactor 0.1
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el factor utilizado para calcular la caducidad de un documento determinado cuando el servidor no la especifica. El cálculo se hace mediante la siguiente fórmula:

```
caducidad =  (fecha de última modificación del documento) *
(número de coma flotante)
```

Así que, si un documento se modificó hace 24 horas, al aplicar el factor 0.1 predeterminado de Apache, la caducidad del documento será de 2.4 horas. Si el valor obtenido es mayor que el especificado con CacheMaxExpire, se aplicará este último.

CacheDirLength

```
Sintaxis: CacheDirLength <longitud>
Predeterminado: CacheDirLength 1
Contexto: configuración servidor, host virtual
```

Cuando se activa la caché de disco, Apache crea subdirectorios en los directorios especificados a través de la directriz CacheRoot. Esta directriz especifica el número de caracteres que se pueden utilizar para crear los nombres de los directorios. La verdad es que no hace falta cambiar el valor predeterminado. Para aquellos usuarios curiosos que quieren saber cómo o por qué se crean estos subdirectorios, se lo explico a continuación.

Apache utiliza un sistema de conversión cuando crea el nombre de la ruta y del archivo de los datos procedentes de un URL que se van a guardar en memoria. Por ejemplo, cuando tiene activada la caché del acceso de un URL (como puede ser www.microsoft.com) que va a efectuar a través de un proxy Apache, el servidor convierte el localizador URL para que las posteriores recuperaciones de datos sean más veloces. La conversión puede ser algo así: 1YSRxSmB20Q_HkqkTuXeqvw. Si se utiliza el valor predeterminado de las directrices CacheDirLength y CacheDirLevels, Apache guardará los datos procedentes de www.microsoft.com en un archivo que se llamará:

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

En este caso %CacheRoot% es el directorio que se especifica a través de la directriz CacheRoot. Los directorios /1/Y/S/ se han creado a causa del valor predeterminado de CacheDirLevels. Cuando se vuelva a solicitar otro documento que comparta el mismo URL, todo lo que tendrá que hacer Apache será volver a calcular la conversión de la página a partir de una ruta determinada.

CacheDirLevels

```
Sintaxis: CacheDirLevels <niveles>
Predeterminado: CacheDirLevels 3
Contexto: configuración servidor, host virtual
```

Especifica el número de subdirectorios que creará Apache para almacenar datos. Si desea más información, consulte la sección "CacheDirLength".

CacheDefaultExpire

```
Sintaxis: CacheDefaultExpire <n horas>
Predeterminado: CacheDefaultExpire 1
Contexto: configuración servidor, host virtual
```

Esta directriz proporciona el número predeterminado de horas (n) que se usará para determinar la caducidad de un archivo que se encuentra en la caché, cuando se desconoce la fecha de la última modificación. CachemaxExpire no anula este parámetro.

NoCache

```
Sintaxis: NoCache <Nombre dominio  Subred | Dirección IP |
Nombre host> ...
Predeterminado: Nada
Contexto: configuración servidor, host virtual
```

La directriz NoCache determina una lista de host, nombres de dominio y direcciones IP (separados por espacios en blanco) para los cuales no se empleará la memoria caché. Esta directriz tendrá que utilizarse para desactivar la caché de los servidores Web locales que se encuentran en una intranet. Obsérvese que el servidor proxy también puede trabajar con nombres parciales. Si quiere desactivarlo todo, utilice:

```
NoCache *
```

Configuración del servidor proxy

Ahora que conoce las directrices que puede usar para configurar el servidor proxy en Apache, nos centramos en algunos detalles de la configuración.

Para activar el servidor proxy, tendrá que ajustar el valor de ProxyRequests a On. Luego, la configuración dependerá de lo que desee que haga su servidor proxy. Independientemente de lo que decida, la configuración tendrá que aparecer dentro del contenedor especial <Directory ...>, cuyo aspecto será el siguiente:

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

Cualquier directriz que se quiera utilizar para controlar el comportamiento del servidor proxy tendrá que ir dentro de este contenedor. El asterisco es

el comodín que se emplea para los URL. En otras palabras, cuando se procesa una petición de www.nitec.com en un servidor Web, su aspecto será el siguiente:

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

Nota: también se puede utilizar el contenedor <Directory ~ /RE/>, que usa expresiones regulares.

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

Vamos a ver las configuraciones más usuales.

Conectar una red IP privada a Internet

En este caso, tan sólo un ordenador tendrá la dirección IP de acceso a Internet, tal y como se puede apreciar en la figura 14.3. Esta máquina se utiliza como servidor Apache proxy cuando se activa la directriz ProxyRequests y no hace falta ninguna configuración adicional. Todas las peticiones serán atendidas a través del servidor proxy.

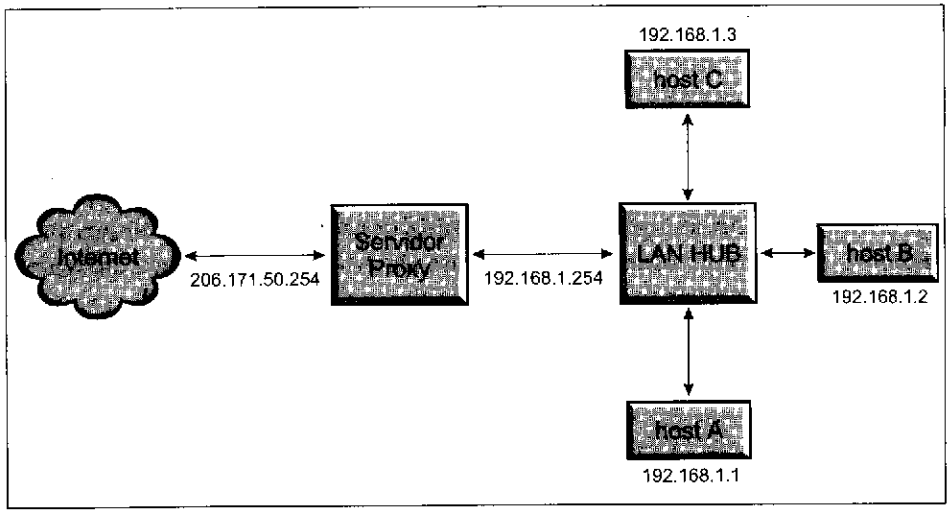


Figura 14.3. Proxy para una red IP privada que se conecta a Internet

Observe que en este tipo de configuración, el servidor proxy debe disponer de varias direcciones. En otras palabras, tiene que acceder a una red privada no routable (192.168.1.0) y a otra IP routable (206.171.50.0). De esta forma, el servidor actúa como un firewall para la red privada, aunque las direcciones IP no routables tenían el mismo efecto. El servidor proxy permite que los host tengan acceso a los servicios de Internet, como Web, FTP, etc.

Vamos a ver un servidor proxy directo.

Guardar en caché los sitios Web

Como gran parte del contenido de Internet y de las intranet es estático, al guardarlo en la memoria caché del servidor proxy se puede ahorrar una cantidad importante del ancho de banda de la red. Un servidor proxy que pueda trabajar con memoria caché buscará los documentos únicamente cuando no tenga una copia en la memoria. Para poder activar esta propiedad tendrá que especificar las directrices de la caché dentro de un contenedor especial. Por ejemplo:

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

Esta configuración define un servidor proxy con caché que guardará una copia de los archivos en el directorio /www/cache. Puede escribir 1.024K de datos (1MB) y los datos se borrarán de la caché cada 24 horas.

Si no quiere que la gente externa abuse de su servidor proxy, puede restringir el acceso por medio de una autenticación en la que se utilice el nombre de usuario y una contraseña.

Para controlar qué host acceden al servidor, puede crear una configuración como la siguiente:

```
<Directory proxy:*>

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

</Directory>
```

Esta configuración negará el acceso a todo el mundo menos a myhost.nitec.com.

Si quiere usar la autenticación nombre de usuario/contraseña, puede utilizar algo similar a:

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

También se puede restringir el acceso de un protocolo:

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

Así puede controlar cómo se procesan las peticiones HTTP a través de su servidor proxy. Del mismo modo, puede utilizar la siguiente configuración para determinar cómo tratará el servidor proxy a cada protocolo:

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

o

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

Puede crear un servidor virtual exclusivo para el servidor proxy. En este caso, las directrices aparecerían dentro del contenedor del servidor virtual:

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

Hacer un mirror de un sitio Web

Un mirror de un sitio Web es una copia de un sistema Web remoto. Por ejemplo, si desea hacer un mirror del sitio Web www.apache.org para que sus usuarios se conecten con el mirror y tarden menos en conseguir la información deseada, puede utilizar el servidor proxy:

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

De esta forma, el servidor proxy se convertirá en un mirror de www.apache.org. Por ejemplo, esta configuración hace que el servidor proxy blackhole.nitec.com se convierta en un mirror de Apache, como se puede apreciar en la figura 14.4.

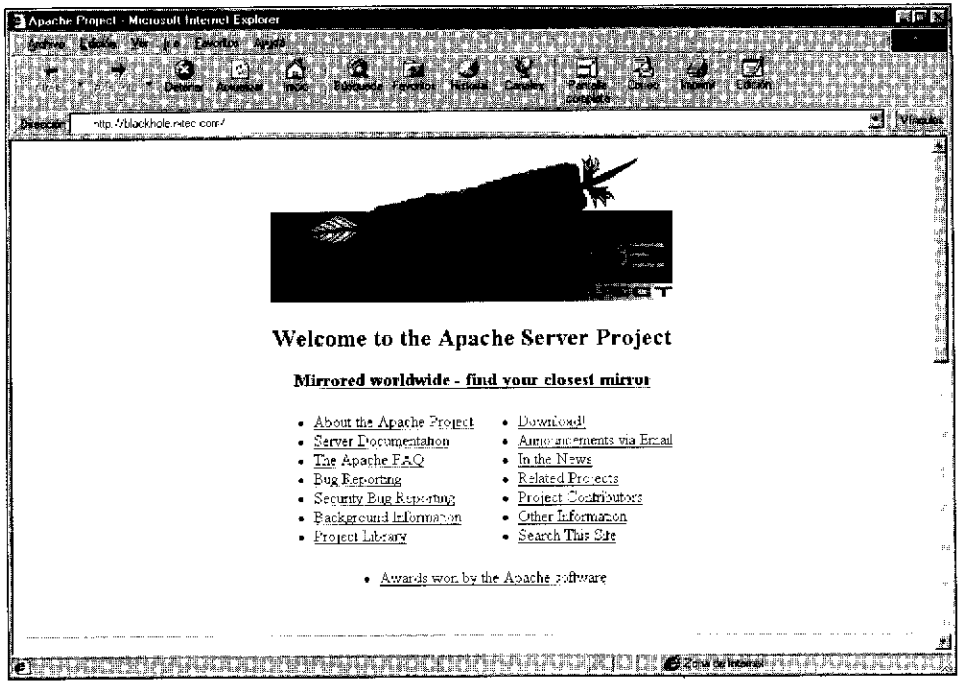


Figura 14.4. Utilice el servidor proxy para crear un mirror del sitio Web de Apache

Cuando un usuario introduce http://blackhole.nitec.com como URL, recibirá la página del mirror que funcionará exactamente igual que si hubiese entrado en www.apache.org.

Advertencia: antes de hacer un mirror de un sitio Web, conviene que pida permiso, ya que se puede encontrar con problemas relacionados con los permisos del Copyright.

Configurar los servidores Web

Ya está listo para configurar su explorador Web en el cliente de su host. Los exploradores Web más populares facilitan el uso de servidores Web. En

las secciones siguientes mostraré como configurar la última versión de Netscape Navigator y de Microsoft Internet Explorer (IE) para que trabajen con servidores proxy. Hay dos tipos de configuraciones: manual y automática.

Optará por las configuraciones manuales cuando tenga muy pocos clientes y la configuración de los servidores proxy no cambie muy a menudo. Si sus necesidades son otras, debería saltarse esta sección y pasar a la que trata de la configuración automática.

Los pasos que le enumeramos a continuación le servirán de guía a través del proceso de configuración manual del servidor proxy:

1. Seleccione Edición, Preferencias, que se encuentra en el menú de comandos de Navigator. Verá un cuadro de diálogo similar al que aparece en la figura 14.5.

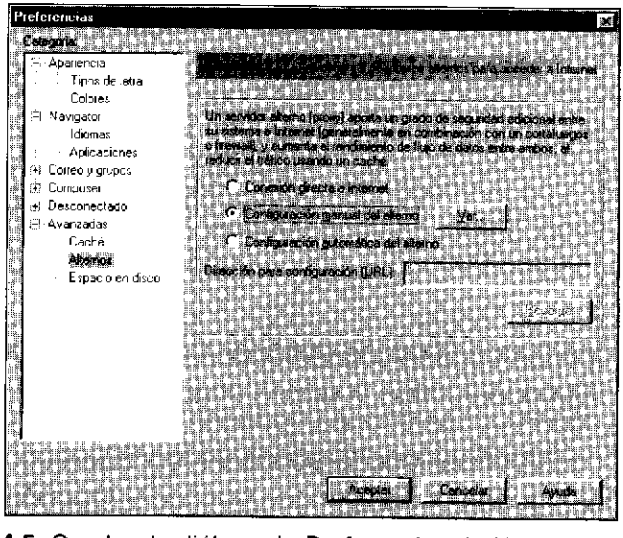


Figura 14.5. Cuadro de diálogo de Preferencias de Netscape Navigator

2. Haga clic en Avanzadas.
3. Haga clic en Alternos.
4. Seleccione configuración manual de alterno y haga clic en el botón Ver. Accederá a la pantalla que aparece en la figura 14.6.
5. Introduzca los URL de HTTP, FTP y Seguridad (HTTPS) en los campos correspondientes. Como está utilizando un único servidor como proxy, el URL y los puertos serán el mismo. Si tiene un servidor distinto para cada servicio, tendrá que especificarlo.

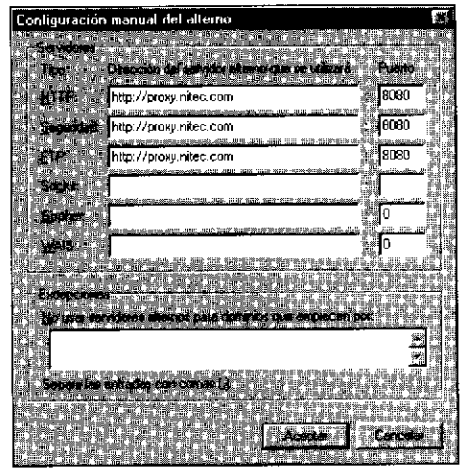


Figura 14.6. Ventana de configuración manual de servidores proxy (alternos) de Netscape Navigator

6. Una vez que haya introducido la información, solicite un documento remoto y compruebe si el servidor proxy funciona correctamente. Una forma de determinarlo será revisando el archivo de registro de actividades y errores del servidor proxy. En la mayoría de los sistemas UNIX se puede utilizar un comando como el siguiente:

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

Ahora vamos a configurar el otro explorador Web. Para configurar manualmente Explorer para que trabaje con servidores proxy, tendrá que hacer lo siguiente:

1. Haga clic en Ver, Opciones de Internet. Se abrirá el cuadro de diálogo que se puede ver en la figura 14.7.
2. Seleccione la pestaña Conexión.
3. Seleccione Usar un servidor proxy para conectar a Internet.
4. Escriba el URL del servidor proxy y su puerto en los campos correspondientes.
5. Haga clic en Aceptar para completar la configuración.

Si quiere especificar un servidor proxy diferente para cada protocolo, puede utilizar el botón Opciones avanzadas, con lo que accederá al cuadro de diálogo que aparece en la figura 14.8.

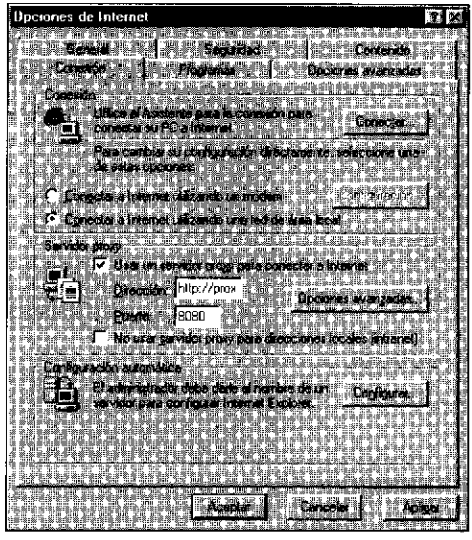


Figura 14.7. Ventana desde la cual se configura manualmente el servidor proxy

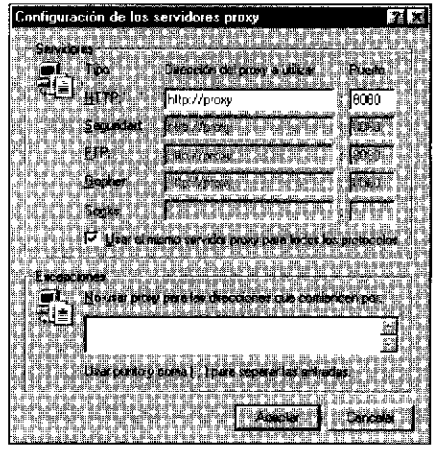


Figura 14.8. Cuadro de diálogo de opciones avanzadas de la configuración de servidores proxy de Microsoft Internet Explorer

Desde esta pantalla, al igual que ocurría con Netscape Navigator, podrá especificar la configuración de servidores proxy diferentes. Para que el explorador utilice la nueva configuración bastará con hacer clic en Aceptar en todas las ventanas. Como puede ver, la configuración manual de los servidores es muy sencilla. De todas formas, si tiene que modificar la configuración de una serie de ordenadores, el proceso le puede llevar bastante tiempo. Aquí es donde se suele utilizar la configuración automática.

Configuración automática de los exploradores para que trabajen con servidores proxy

Nuestros amigos de Netscape se adelantaron a los problemas que podían sufrir sus clientes a la hora de configurar manualmente los servidores proxy e idearon una configuración alternativa. Microsoft ha desarrollado un sistema de configuración automática para su Internet Explorer. Para crear los archivos de configuración automática, Microsoft ha desarrollado un Kit para Administradores de Internet Explorer (IEAK). Como para usar este kit de desarrollo hace falta una licencia y para eso hay que pagar a Microsoft, siento comunicarle que este autor no ha podido trabajar con él. De todas formas, fuentes fidedignas me han confirmado que en este paquete hay un sistema de configuración automático muy parecido al de Netscape. Incluso se pueden utilizar los mismos script.

Esta sección se aplica tanto a IE como a Navigator. La única diferencia es que si quiere trabajar con el primero, tiene que configurarlo para crear los archivos con IEAK.

La configuración de proxy automática se efectúa gracias a un JavaScript especial. Es igual para Netscape Navigator y para IE. El JavaScript especial tiene los siguientes requisitos:

Ha de implementar una función llamada FindProxyForURL. Tiene la siguiente forma:

```
function FindProxyForURL(url, host)
        {
              // Aquí aparecería JavaScript

          return "proxy to use for servicing the URL";

        }
```

Los argumentos de esta función son url y host. Url es el URL completo que se solicita y host es el nombre del servidor que se tiene que extraer del URL. Por ejemplo, cuando el explorador Web detecta una petición, llamará a la función de la siguiente forma:

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

Nota: el argumento host que se encuentra en la llamada de la función, es en realidad la cadena que se encuentra entre :// y los primeros : o /. El número del puerto no se incluye en este parámetro.

La función tiene que devolver una cadena en la que aparezca la configuración del proxy para una petición de un URL determinado. La aceptación de los valores de las cadenas que representan la configuración del proxy son:

NULL	Cuando se devuelve un valor NULL (o la cadena NULL), se le indica al explorador que no hay ningún servidor proxy determinado para esta petición.
DIRECT	Las conexiones se han de establecer directamente, sin proxy.
PROXY host:puerto	Especifica el proxy que hay que utilizar.
SOCKS host:puerto	Especifica el SOCKS que tiene que utilizar el servidor.

Obviamente, los valores más interesantes son DIRECT y PROXY. Cuando tiene varios servidores proxy o SOCKS, puede devolver una lista de pares host:puerto, en vez de uno solo. Por ejemplo:

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

Esta configuración proxy le indicará al explorador que, en primer lugar, trate de efectuar la transacción utilizando best-proxy.nitec.com, luego good-proxy.nitec.com, etc. Observe que cada par host:puerto está separado por un punto y coma y que se repite la palabra PROXY para cada par. Si fallasen todos los servidores, se consultaría al usuario antes de volver a intentar establecer la conexión.

Nota: cuando fallan todos los proxy y se ha especificado ninguna opción DIRECT, el explorador preguntará al usuario si ignora temporalmente a estos servidores y trata de acceder directamente a la dirección solicitada.

Para evitar las interacciones con los usuarios, sustituya la configuración por la siguiente:

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

En este caso se utilizará el proxy basado en SOCKS cuando el proxy principal falle.

Cuando un proxy no responda a la llamada, Netscape Navigator tratará de volver a ponerse en contacto con él cuando transcurran 30 minutos. Siempre que se falle, se volverá a intentar cuando pasen otros 30 minutos.

Para ayudar a los administradores Web (que se supone que saben programar en JavaScript), hay una serie de funciones predefinidas. Las puede ver en la tabla 14.1.

Tabla 14.1. Funciones predefinidas para utilizarlas en el script de configuración automática del proxy

Nombre de la función	Explicación	Ejemplos
isPlainHostName(host)	Devuelve true si no hay un punto en el host. En otras palabras, cuando no se incluye el nombre del host.	`isPlainHostName ("blackhole")` devuelve true. `isPlainHostName ("blackhole.nitec.com")` devuelve false.
dnsDomainIs(host, domain)	Devuelve true si el host pertenece al dominio. El nombre del dominio tiene que empezar por un punto.	`dnsDomainIs ("www.nitec.com",".nitec.com")` devuelve true. `dnsDomainIs("www.apache.com",".nitec.com")` devuelve false.
localHostOrDomainIs (host, fqdnhost)	Devuelve true si la parte del host de fqdnhost coincide con el host.	`localHostOrDomainIs ("a.b.com", "a.b.com")` devuelve true. `localHostOrDomainIs("a.b", "a.b.com")` devuelve true. `localHostOrDomainIs("a.b.org", "a.c.com")` devuelve false.
isResolvable(host)	Si el servidor DNS puede convertir el nombre del host en una dirección IP, devolverá true. En caso contrario, false.	`isResolvable ("{hyperlink}");` devuelve true porque {hyperlink} tiene un registro DNS.

Nombre de la función	Explicación	Ejemplos
	Observe que el uso de esta función puede disminuir la velocidad de los exploradores porque se tendrá que efectuar una prueba con todas las peticiones DNS.	
isInNet(host,IP address pattern, netmask)	Devuelve true si coinciden las direcciones IP con las de la mayoría de las máquinas especificadas en el segundo argumento. La comparación se efectúa utilizando la máscara de red de la siguiente manera: si uno de los octetos de la máscara es un 255, tiene que coincidir con el octeto correspondiente de la dirección IP. Si es 0, se ignorará el octeto correspondiente de la dirección IP. Observe que el uso de esta función puede disminuir la velocidad de los exploradores porque se tendrá que efectuar una prueba con todas las peticiones DNS.	Si el host tiene una dirección IP de 206.171.50.51: `isInNet(host,"206.171.50.50","255.255.255.0");` devuelve true porque, de acuerdo con la máscara de red, los tres primeros octetos tienen que coincidir y el último se ha de ignorar.
dnsResolve(host)	Devuelve la dirección IP del host si tiene éxito. Observe que el uso de esta función puede disminuir la velocidad de los exploradores porque se tendrá que efectuar una prueba con todas las peticiones DNS.	`dnsResolve("proxy.nitec.com")` devuelve "206.171.50.50"

Nombre de la función	Explicación	Ejemplos
myIpAddress()	Devuelve la dirección IP del host en el que se está ejecutando el explorador Web. Observe que el uso de esta función puede disminuir la velocidad de los exploradores porque se tendrá que efectuar una prueba con todas las peticiones DNS.	`var hostIP = myIpAddress()` devuelve el IP del host del explorador Web y lo almacena en una variable llamada hostIP.
dnsDomainLevels (host)	Devuelve el número de los niveles del nombre del host.	`dnsDomainLevels ("www.nitec.com")` devuelve 2.
shExpMatch(string, shellExpression)	Devuelve true si la cadena coincide con la expresión shell.	`shExpMatch ("path/to/dir","*/to/*")` devuelve true. `shExpMatch ("abcdef", "123")` devuelve false.
weekdayRange (weekday1, weekday2, gmt)	Sólo es necesario el primer argumento, weekday1. Devuelve true si el día en el que se ejecuta esta función es igual a weekday1 o se encuentra comprendido entre weekday1 y weekday2. El tercer parámetro gmt es la hora GMT, que se utiliza en vez de la hora local. Los valores aceptables para weekday1 son: SUN, MON, TUE, WED, THU, FRI o SAT.	`weekdayRange ("FRI")` devuelve true si es Viernes. `weekdayRange ("MON", "FRI", "GMT")` devuelve true si el día de la semana está comprendido entre el Lunes y el Viernes.
dateRange(day) dateRange(day1, day2) dateRange(month)	Devuelve true si el día, mes y año están dentro del intervalo. Los valores	`dateRange(31)` devuelve true si el día es el 31.

Nombre de la función	Explicación	Ejemplos
dateRange(month1, month2) dateRange(year) dateRange(year1, year2) dateRange(day1, month1, day2, month2) dateRange(month1, year1, month2, year2) dateRange(day1, month1, year1, day2, month2, year2) dateRange(day1, month1, year1, day2, month2, year2, gmt)	pueden ser: day, 1-31; month, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, o DEC; year, número de cuatro cifras; gmt la hora GMT o nada (hora local).	`dateRange("JAN", "APR")` devuelve true si el mes está comprendido entre Enero y Abril. `dateRange(1995)` devuelve true si el año es 1995.
timeRange(hour) timeRange(hour1, hour2) timeRange(hour1, min1, hour2, min2) timeRange(hour1, min1, sec1, hour2, min2, sec2) timeRange(hour1, min1, sec1, hour2, min2, sec2, gmt)	Devuelve true si la hora, minutos y segundos coinciden con los especificados. Y vuelve a mostrar true si la unidad temporal también coincide con la determinada. Los valo-r es de hour pueden ser 0-23; min, 0-59; y gmt la hora GMT o nada (hora local).	`timeRange(9, 17)` devuelve true si la hora actual está comprendida entre las 9 AM y las 5 PM.

Con la ayuda de las funciones predefinidas y con las funciones personalizadas, puede escribir FindProxyForURL de tal forma que le muestre la cadena de configuración de cada servidor proxy que atienda una petición.

Cuando se inicia el servidor Web, carga una función del archivo JavaScript (más tarde veremos cómo se pone este archivo a disposición del explorador Web) y llama a FindProxyForURL en cada petición de URL. El explorador suministra los argumentos host y URL a la función para que los pueda devolver de acuerdo con la configuración del proxy.

Vamos a ver algunos ejemplos que muestran las distintas formas en las que se puede escribir la función FindProxyForURL.

Uso del servidor proxy únicamente con las peticiones URL remotas

En este ejemplo, la idea es decirle al explorador Web que el proxy sólo atenderá las peticiones de URL remotos, tal como se aprecia en la figura 14.9.

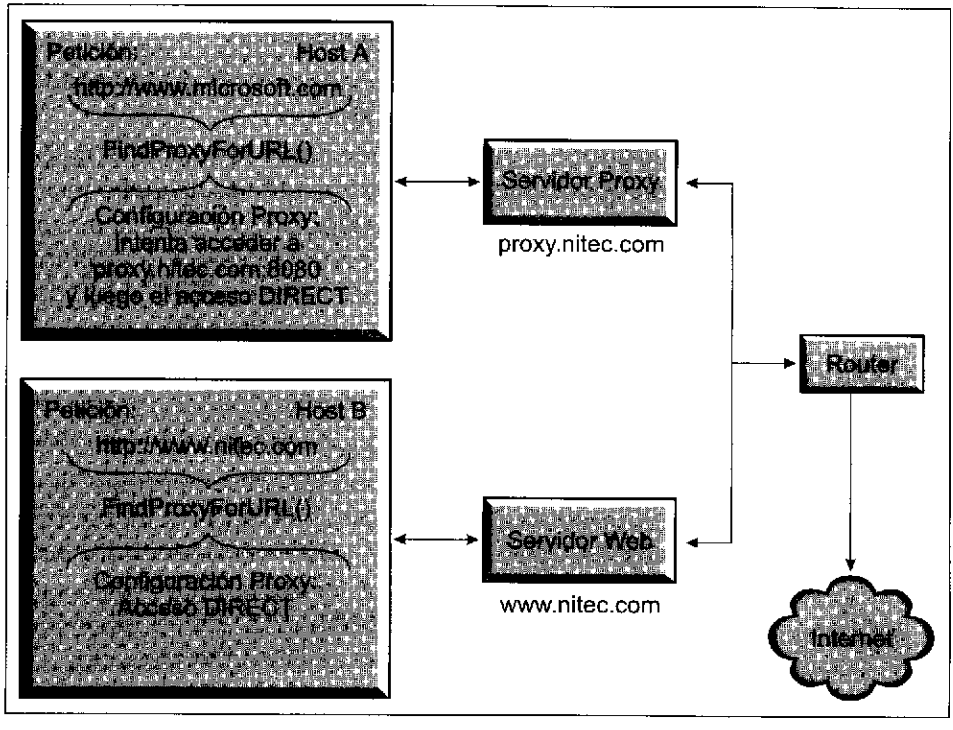


Figura 14.9. Uso del servidor proxy únicamente con las peticiones URL remotas

El listado muestra un ejemplo de la función FindProxyForURL( ).

```
function FindProxyForURL(url, host) {

// Comprueba si el host es local. Si lo es, especifica una
// conexión DIRECT. En caso contrario utilizará el proxy.
if (isPlainHostName(host) ||
    dnsDomainIs(host, ".nitec.com"))
       return "DIRECT";
else
      return "PROXY proxy.nitec.com:8081; DIRECT";
}
```

Listado 14.1. Uso del servidor proxy únicamente con las peticiones URL remotas

Cuando se pide un URL a través del explorador Web del usuario, éste invocará a FindProxyForURL( ) con los argumentos url y host. En primer lugar llama a la función isPlainHostName para ver si el nombre del host está en texto plano (www). Si no lo está, la función devolverá false. Ahora se llama a la función dnsDomainIs para que compruebe si el dominio es .nitec.com. Devuelve false. Como ambas pruebas devuelven false, se ejecuta la otra parte del condicional. En otras palabras, la petición URL para que se le entregue al explorador Web la configuración del proxy:

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

le dice que utilice el proxy llamado proxy.nitec.com a través del puerto 8080. Si no pudiese establecer la conexión a través de dicho puerto, trataría de acceder a él por medio de una conexión directa. Para la mayoría de las instalaciones de los servidores proxy, esta configuración sería más que suficiente. Pero vamos a ver un caso algo más complicado.

Uso de varios servidores proxy

En este caso se utilizan varios servidores proxy. En la figura 14.10 se puede ver una red en la que aparecen varios servidores proxy: http-proxy.nitec.com se utiliza para atender las peticiones URL HTTP remotas, ftp-proxy.nitec.com para atender las peticiones FTP HTTP remotas y ssl-proxy.nitec.com para atender las peticiones SSL HTTP remotas. El resto de las peticiones que utilicen otros protocolos como GOPHER, NEWS, etc. se atienden directamente. Las peticiones locales también se atienden directamente.

Para implementar esta configuración se utiliza un FindProxyForURL algo más complejo, que es el que aparece en el listado 14.2.

```
function FindProxyForURL(url, host) {

  // El URL ¿es local? Si lo es se utiliza una conexión DIRECT
   if (isPlainHostName(host) ||
       dnsDomainIs(host, ".nitec.com")) {
return "DIRECT";
}

  // Si el URL es remoto se verifica el proxy que tiene
  // que utilizar.
  else{
  if (url.substring(0, 5) == "http:") {
        return "PROXY http-proxy.nitec.com:8080";
        }
  else if (url.substring(0, 4) == "ftp:") {
```

```
        return "PROXY ftp proxy.nitec.com:8080";
        }
    else if (url.substring(0, 6) == "https:") {
        return "PROXY ssl proxy.nitec.com:8080";
        }
    else{
        return "DIRECT";
        }
    }
}
```

Listado 14.2. FindProxyForURL para una configuración en la que aparezcan varios servidores proxy

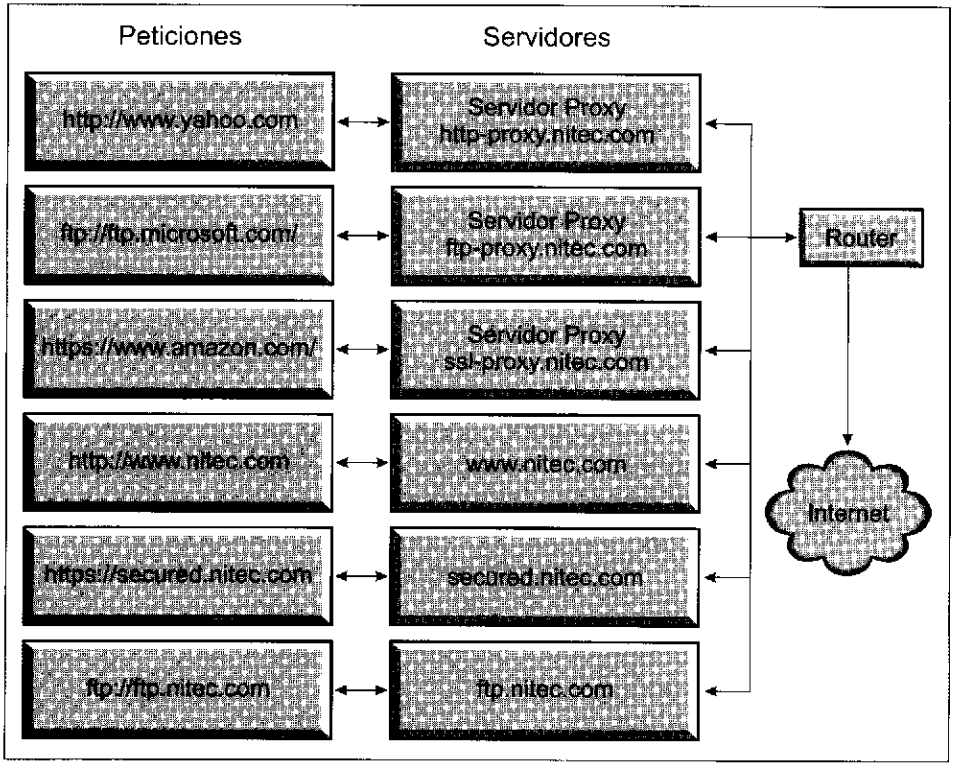


Figura 14.10. Uso de varios servidores proxy

Lo primero que hace esta función es comprobar si la petición URL es local. En caso afirmativo, la atiende directamente. Si proviene de un servidor remoto, se analiza el protocolo del URL para que trabaje con el servidor proxy adecuado. De todas formas, los únicos protocolos que se pueden reconocer son HTTP, FTP y HTTPS y sus peticiones se envían directamente al servidor

proxy adecuado. Cuando se recibe una petición cuyo protocolo no coincide con ninguno de los anteriores, se atiende directamente.

También se puede personalizar la configuración del servidor proxy basándose en el host que accede a él. Para ello se utiliza un script CGI que trabaja con FindProxyForURL de una forma o de otra, dependiendo de REMOTE_HOST (el servidor remoto). En el listado 14.3 puede ver su código fuente.

```
#!/usr/local/bin/perl
#
# Un script en Perl que muestra la configuración del
# servidor proxy.
# $Author$
# $Revision$
# $Id$

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

# Imprime el content type para que el explorador sepa que
# es la configuración de un proxy.
print "Content-type: application/x-ns-proxy-autoconfig\n\n";

# Si la petición proviene de un host con una dirección IP
# 206.171.50.51, entonces se obtiene la configuración del
# proxy de la subrutina &specialClient
#
if ($client =~ /206\.171\.50\.51/){ &specialClient; }

# Si la petición proviene de otro cliente cualquiera, la
# configuración del proxy se obtiene de la subrutina &otherClients
else { &otherClients;   }

exit 0;

sub specialClient{
#
# Esta subrutina devuelve la configuración del servidor proxy
#

print <<FUNC;

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

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

```
        else
            return "DIRECT";
    }
FUNC

}

sub otherClients{
#
# Esta subrutina devuelve la configuración del servidor proxy
#

print <<FUNC;

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

FUNC

}
```

Listado 14.3. script proxy.pl

Este script devuelve la configuración de un servidor proxy a un host que tenga la dirección IP 206.171.50.0 y al resto de los host que tengan una configuración diferente. El resultado es exactamente el mismo que si hubiese creado un archivo .pac. En este caso, se le pide al servidor Web que trate con un script en vez de vérselas con un archivo .pac. Pero como el script envía el content-type dentro de un archivo .pac, el explorador nunca sabrá si está tratando con un archivo .pac de configuración o con un script. Aunque el script del ejemplo no es gran cosa, se pueden utilizar algunos similares para determinar configuraciones más complejas de servidores proxy.

15 SSL para Apache

Hace tan sólo unos años, Internet seguía siendo aquello para lo que se diseñó: una red mundial para científicos e ingenieros. Pero gracias a la Red, Internet se ha convertido en la red de todo el mundo. Parece que hoy todos tenemos algo en Internet. Se han abierto nuevas fronteras comerciales, miles de negocios (grandes y pequeños) tienen abiertas las puertas de sus sitios Web a todos los clientes del mundo. De todas formas, conviene que estos clientes sean cautos porque se sabe que no todas las partes de Internet son seguras.

Para eliminar el sentido de la inseguridad, los sitios Web comerciales tienen que ofrecer un sistema seguro en el que se puedan efectuar las transacciones entre sus páginas Web y los clientes. SSL (Secured Socket Layer) es una de estas tecnologías. En este capítulo veremos el importante papel que juega esta tecnología en el comercio que se efectúa a través de la Red utilizando servidores Apache.

La fundación de SSL: encriptación

Cuando los datos viajan de un punto a otro de Internet, pasan por una gran cantidad de ordenadores como routers, gateways y otros dispositivos de red. Por ejemplo, cuando un visitante de www.nitec.com/ introduce el número de

su tarjeta de crédito en el formulario HTTL que se encuentra en una de sus páginas Web, es posible que la ruta que siguen los datos hasta llegar al su destino sea similar a la que aparece en la figura 15.1.

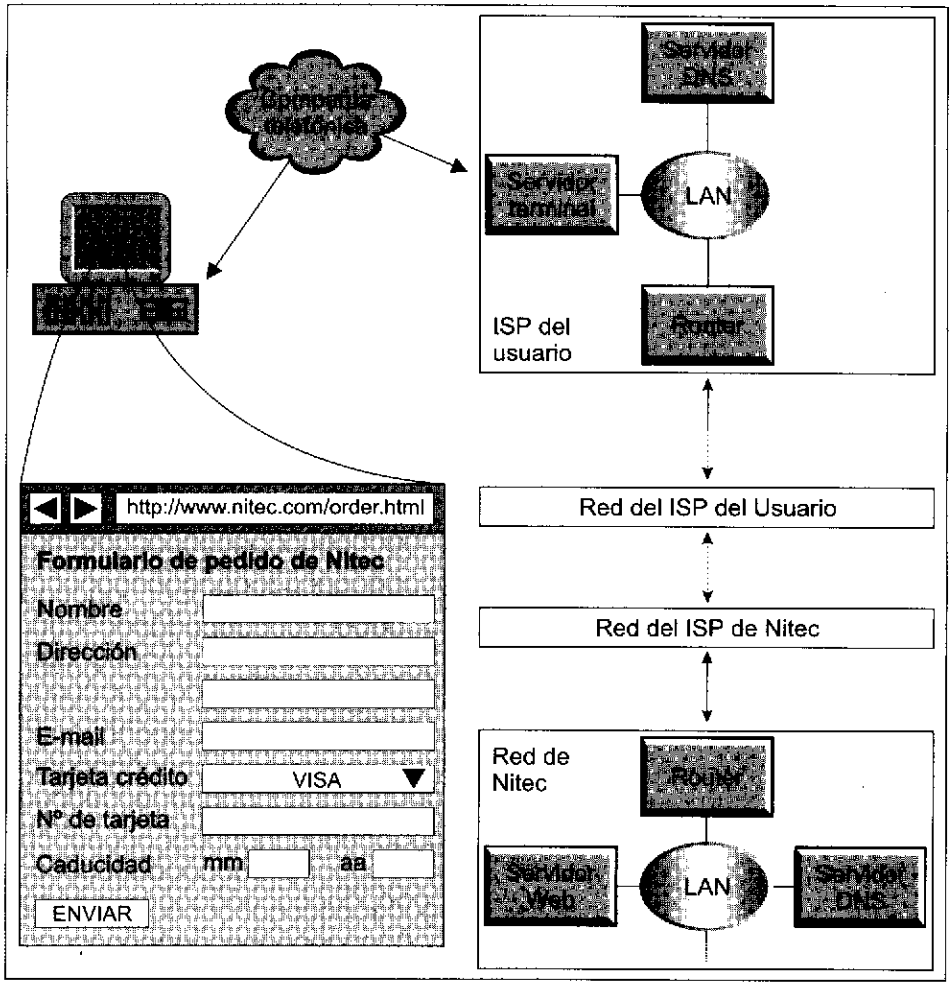


Figura 15.1. Datos viajando de un punto a otro a través de Internet

Como puede ver, los datos tienen que viajar a través de varios nodos, por lo que hay cierta probabilidad de que alguien lo intercepte. Aunque la velocidad que llevan es muy alta (generalmente de milisegundos), siempre se pueden interceptar. Esta es una razón por la que necesitamos un mecanismo seguro para intercambiar datos importantes. Este nivel de seguridad se obtiene a través de la encriptación.

Técnicamente hablando, la encriptación en un sistema de codificación matemático que se asegura de que la única persona que pueda acceder a los datos sea el destinatario. Así se ocultan los datos de ojos curiosos. La encriptación se utiliza para restringir el acceso a las fuentes. Por ejemplo, si se registra en un sistema UNIX o bien Windows NT, la contraseña o las claves que se guardan en el sistema suelen estar encriptadas. En la mayoría de las plataformas UNIX, la contraseña del usuario se encripta y se compara con la que se tiene guardada en el fichero /etc/passwd. Si la comparación tiene éxito, el usuario podrá acceder a la fuente solicitada. Hay dos tipos de encriptación, que veremos en las dos secciones siguientes.

Encriptación simétrica

Este sistema es muy parecido al de claves y bloqueos que se utiliza todos los días. Por ejemplo, tiene una misma llave para abrir y cerrar su coche. De forma similar, en una encriptación simétrica, se utiliza una única llave para bloquear y desbloquear. En la figura 15.2 tiene un diagrama explicativo.

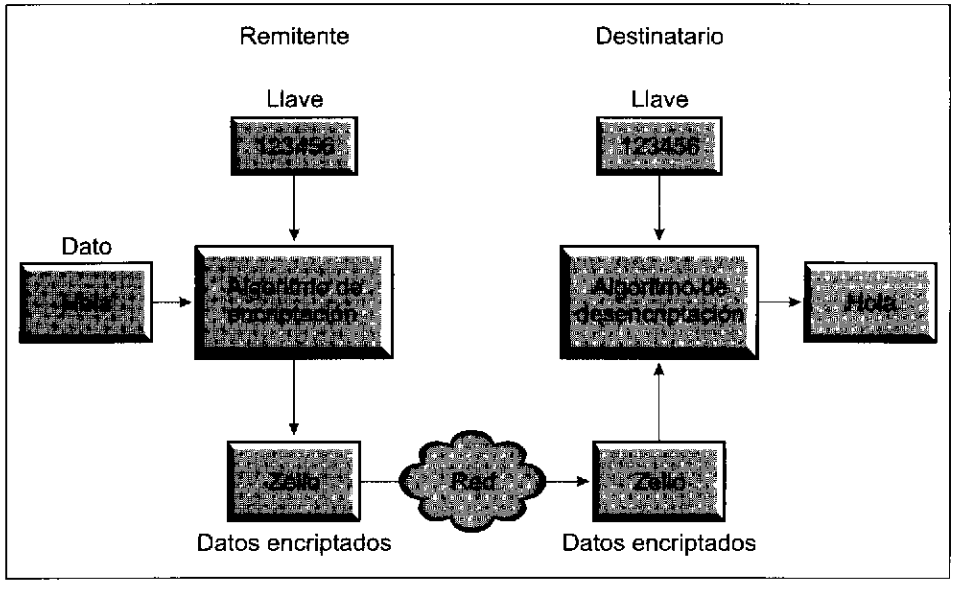


Figura 15.2. Ejemplo de encriptación simétrica

Como en este sistema únicamente se utiliza una llave, todas las partes implicadas tendrán que conocerla. Por otro lado, en una encriptación asimétrica, el funcionamiento es algo diferente.

Encriptación asimétrica

En este caso hay dos llaves: una pública y otra privada. Comparándolo con el sistema anterior, la llave extra es la pública. En la figura 15.3 tenemos un esquema en el que se muestra cómo funciona este sistema.

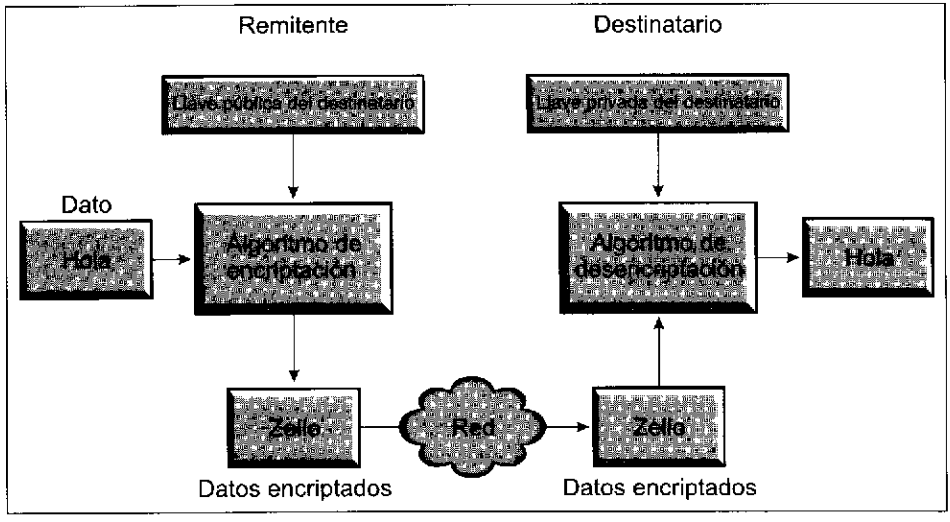


Figura 15.3. Ejemplo de encriptación asimétrica

Cuando se encriptan los datos con una llave pública, sólo se podrán desencriptar con una llave privada, y viceversa. A diferencia de la encriptación simétrica, en este sistema no hace falta que el remitente conozca la llave privada que ha de tener el destinatario para desencriptar los datos. La llave pública se le entrega a todo el mundo, por lo que cualquiera que desee establecer una conexión segura con dicho destinatario podrá utilizarla. La llave privada no se distribuye nunca. Siempre se guarda en secreto.

Utilizando ambos sistemas de encriptación, Netscape desarrolló un protocolo abierto, SSL, que ofrece encriptación de datos, autenticación del servidor y del cliente e integridad de los datos, todo ello a través de una comunicación basada en TCP/IP.

SSL

En una transacción SSL, como la que se puede ver en la figura 15.4, el servidor envía un certificado al sistema del cliente.

Un certificado consiste en un conjunto de datos que contiene la siguiente información:

- Asunto de los datos certificados.
- Entidad que certifica los datos.
- Llave pública de la entidad.
- Cualquier otra información relacionada.

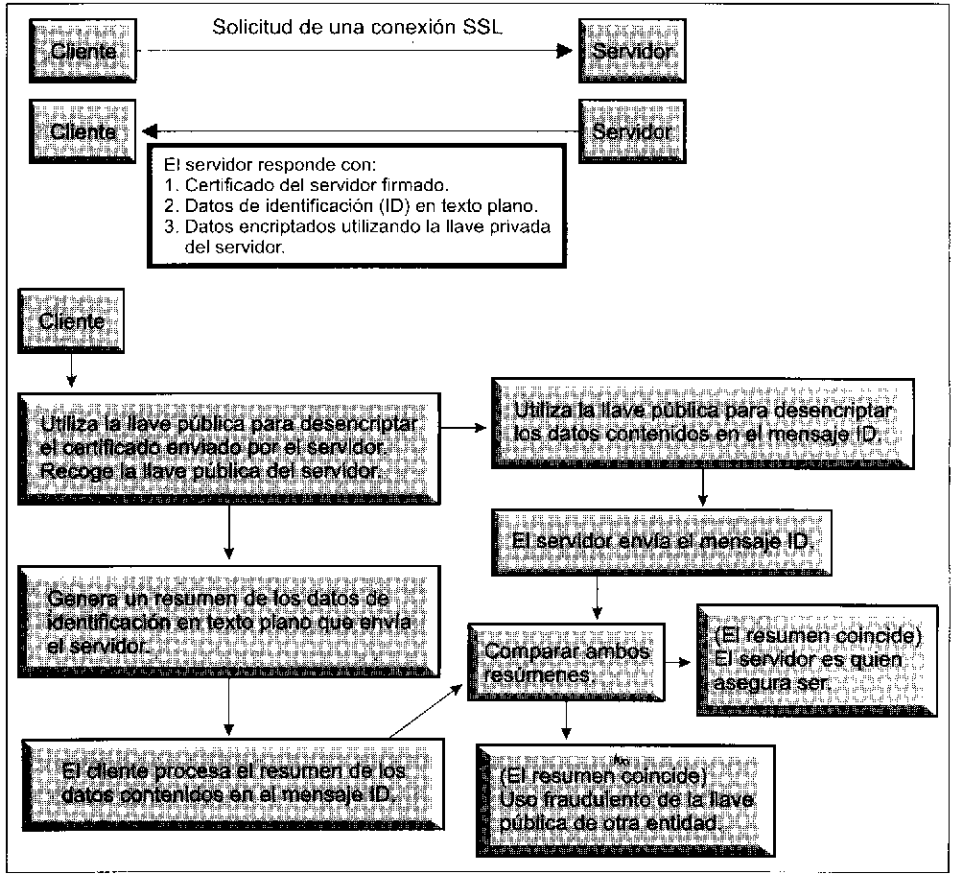


Figura 15.4. Una transacción SSL

Los certificados los suelen utilizar los fabricantes más conocidos y se conocen como Autoridad de Certificación (CA). El certificado se encripta utilizando la llave privada de CA. El cliente desencripta el mensaje utilizando la llave pública de CA.

Como el certificado contiene la llave pública del servidor, el cliente puede desencriptar los datos que éste le envíe. Llegados a este punto, el servidor envía un conjunto de datos con los que se identifica a sí mismo como la identidad mencionada en el certificado. A continuación crea un mensaje resumen con los mismos datos que ha utilizado para identificarse. Este resumen se encripta utilizando la llave privada del servidor. El cliente tiene el certificado procedente del servidor en el que se incluye su llave pública, un mensaje de identificación y un resumen encriptado del mensaje de identificación.

Con la llave pública, el cliente puede desencriptar el resumen del mensaje. Crea un resumen del mensaje de identificación que ha enviado el servidor y lo compara con el resumen que ha recibido. Si coinciden, sabrá que el servidor es quien asegura ser. ¿Por qué? El servidor envió inicialmente un certificado firmado con un CA desconocido y gracias a esta comparación el cliente sabe a quién pertenece esta llave. De todas formas, el cliente tiene que probar que el servidor es quien dice ser, por eso manda un mensaje de identificación junto con un resumen encriptado del mismo que se puede desencriptar utilizando la llave pública.

Si el servidor no tiene la llave privada apropiada, no podrá entonces producir el mismo resumen que el que obtiene el cliente a partir del mensaje de identificación.

Parece complicado, pero la verdad es que esto no acaba aquí. Ahora el cliente puede enviar una llave de encriptación simétrica al servidor, encriptándola con la llave pública del servidor, que puede utilizar esta llave para encriptar los datos que le va a enviar al cliente. ¿Por qué se hace esto? Porque la encriptación simétrica es más rápida que la asimétrica. Mírelo de esta manera: la encriptación asimétrica (llave pública/privada) se usa para transmitir con total seguridad las llaves simétricas de encriptación del cliente al servidor. Esta llave será la que se utilice para establecer canales seguros de comunicación.

Si un impostor se coloca entre los sistemas del cliente y el servidor y consigue interceptar los mensajes que se transmiten las dos partes, ¿podrá causar algún daño? La verdad es que como no tiene la llave de encriptación que utilizan, nunca podrá determinar el contenido de los datos. De todas formas, puede introducir basura en los datos insertando los suyos propios en los paquetes que se envían el cliente y el servidor.

Para evitar esto, el protocolo SSL permite que se trabaje con un código de autenticación de mensajes (MAC). Un MAC es un conjunto de datos que se generan con la llave simétrica y los datos transmitidos. Como el impostor desconoce la llave simétrica, no puede calcular correctamente el valor de MAC. Por ejemplo, se puede utilizar uno de los mejores algoritmos de encriptación,

MD5, desarrollado por RSA Data Security, Inc. Para generar un MAC de 128 bits para cada uno de los datos transmitidos. La potencia informática y el tiempo requerido para producir estos MAC son insignificantes.

Gracias a SSL, el comercio electrónico que se desarrolla en Internet es seguro. Por desgracia, SSL no está listo para funcionar con servidores Apache debido a un problema legal. La implementación actual de SSL utiliza algoritmos patentados por RSA. Como es una empresa que busca su propio beneficio, RSA ha impuesto una serie de restricciones al uso de su tecnología, por lo que los servidores Apache no pueden contar con ella gratuitamente. Otro problema es que el gobierno de los EE.UU. impone serias restricciones relacionadas con la exportación de software especializado en la criptografía. Tan sólo deja que se exporten sistemas de encriptación de 40 bits al resto del mundo. La verdad es que este tipo de encriptación no es válida para encriptar datos de gran importancia. Como Apache lo ha desarrollado un grupo internacional de expertos, el uso de grandes sistemas de encriptación para que pueda trabajar con SSL crea ciertos problemas burocráticos que no sientan nada bien a la hora de desarrollar el proyecto. Por eso se decidió que Apache no podría trabajar directamente con SSL, situación que se mantendría tal cual hasta que se solucionasen todos los problemas.

De todas formas, no se desanime. Hay varias soluciones, tanto comerciales como gratuitas. Si quiere utilizar SSL para propósitos no financieros, puede utilizar Apache-SSL. Si sus necesidades son comerciales, tendrá que hacerse con Stronghold, de C2Net Technology. Los veremos con más detalle en las siguientes secciones.

Apache-SSL

Apache-SSL es un servidor Web seguro, no comercial, que se basa en Apache y que incluye una versión gratuita de la librería de SSL llamada SSLeay. Esta librería puede trabajar con las versiones 2 y 3 de SSL. Antes de entrar a ver los detalles de SSL, conviene que se asegure de que entiende los siguientes puntos legales.

Si tiene pensado trabajar con SSL fuera de los EE.UU., de acuerdo con la documentación de SSLeay no está sujeto a ninguna restricción legal. Pero si utiliza los algoritmos de RSA (los suministra RSAREF) podrá utilizar SSL con fines no comerciales. Si quiere utilizarlo con fines comerciales, tendrá que ponerse en contacto con RSA, ya que tiene sus patentes en los EE.UU. También se puede encontrar con restricciones debidas al uso de algoritmos IDEA en Europa y RC4 en los EE.UU.

De todas formas, si quiere darle una oportunidad a Apache-SSL (y tiene permiso legal para hacerlo), lo primero que debe hacer es configurar SSLeay para que trabaje con su sistema. Lo podrá conseguir en:

```
ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL
```

Configuración de SSLeay

En primer lugar, tendrá que bajarse el archivo tar comprimido del sistema anterior. El sistema más sencillo para construir SSLeay es ejecutar el script en Perl llamado Configure, que se incluye en la distribución. Al ejecutar el script tendrá que especificar el nombre del sistema operativo. Si ejecuta el sistema sin ningún parámetro, verá una lista en la que aparecen los nombres de los sistemas operativos con los que se puede trabajar.

Por ejemplo, para crear los archivos de configuración adecuados para un sistema Linux basado en ELF, como puede ser RedHat 5.0, tendrá que ejecutar el script de la siguiente forma:

```
perl Configure linux elf
```

Cuando se complete la ejecución del script tendrá que hacer lo mismo con los siguientes archivos:

```
make clean
make
make rehash
make test
make install
```

El primero limpia el comando de todos los objetos que quedasen del proceso de compilación y que no tengan utilidad. El segundo compila la librería. El tercero, los certificados de demostración. Y el cuarto lo comprueba todo.

Nota: si trabaja con una plataforma Windows o tiene problemas con la instalación, puede conseguir más información sobre cómo construir SSLeay en las páginas Web de:

```
www.cryptsoft.com/ssleay/doc/Building.html
```

Cuando vea que las comprobaciones de SSLeay se han completado con éxito, tendrá que ejecutar el siguiente comando para instalar el software:

```
make install
```

El ejecutable es ssleay y se instala en el directorio apps. Para acceder a la información de la versión, ejecute el siguiente comando:

```
ssleay version
```

Ahora que su sistema puede trabajar con SSLeay, puede continuar con la configuración de Apache-SSL.

Configuración de Apache-SSL

Hacerse con una copia de los archivos de Apache-SSL es tan sencillo como dirigirse al sistema de www.apache-ssl.org y localizar el sitio FTP que le quede más cerca:

```
ftp://ftp.ox.ac.uk/pub/crypto/SSL/
```

Una vez que se ha hecho con el archivo tar comprimido de Apache-SSL, tendrá que copiarlo en el directorio en el que se encuentra la distribución de Apache (es decir, el directorio superior de Apache) y descomprimirlo allí. Para parchear los archivos de Apache originales, escriba lo siguiente:

```
patch < SSLpatch
```

Cambie al subdirectorio de Apache en el que guarda las aplicaciones y edite el archivo Configuration. Compruebe que el valor de SSL_BASE es el directorio en el que ha descomprimido los archivos fuente de SSLeay. Cuando modifique este valor, tendrá que ejecutar Configure como siempre, con lo que obtendrá el ejecutable de Apache-SSL. Para verificar que la versión del ejecutable es la correcta, escriba:

```
httpsd -v
```

Además de la versión de Apache verá el número de la versión Ben-SSL. Ya está listo para crear un certificado temporal para su servidor Apache seguro.

Crear un certificado temporal

En el proceso de creación de un certificado se utilizan números aleatorios. Para conseguir datos aleatorios para este proceso, cree un archivo de texto con cualquier cosa. Basta con que coja cualquiera de los archivos ya creados y copie su contenido, o que cree uno nuevo. A continuación determine una

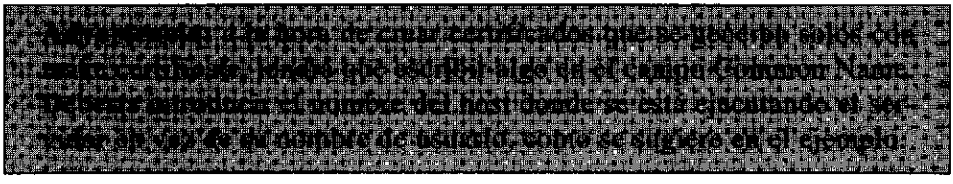
variable a la que llamará RANDFILE y como valor utilice la ruta que conduce al archivo que acaba de crear. Por ejemplo:

```
setenv RANDFILE /tmp/random.txt
```

Esto le indica a la variable RANDFILE que el archivo con el que tiene que trabajar es /tmp/random.txt. Conviene que consulte la documentación de su shell para ver cómo tiene que configurar este tipo de variables de entorno. El comando anterior funciona con el shell tcsh. Una vez que haya completado la configuración de esta variable, ejecute el siguiente comando:

```
make certificate
```

Se le pedirá que introduzca cierta información general, como el nombre de la empresa, el de usuario y la dirección de e-mail.



El certificado se guarda en el archivo SSLconf/conf/httpsd.pem.

Directrices para configurar Apache-SSL

Ahora ya está listo para configurar Apache-SSL. De todas formas, tendrá que estar al tanto de las siguiente directrices de Apache-SSL. Utilizará las siguientes durante el proceso de configuración.

SSLDisable

```
Sintaxis: SSLDisable
Contexto: configuración servidor, host virtual
```

Esta directriz desactiva SSL. No necesita ningún argumento.

SSLCertificateFile

```
Sintaxis: SSLCertificateFile nombre_archivo_certificado
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el nombre del archivo en el que se encuentra el certificado del host del sitio Web. Obsérvese que necesita un certificado

distinto para cada sitio Web, por lo que, si tiene pensado trabajar con una conexión SSL para host virtuales, necesitará un certificado distinto para cada uno de ellos.

SSLCertificateKeyFile

```
Sintaxis: SSLCertificateKeyFile nombre_archivo_certificado
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el archivo en el que se guarda el certificado de la llave privada Si no utiliza esta directriz, se supondrá que se guarda en el archivo que especifique en SSLCertificateKeyFile.

SSLCACertificatePath

```
Sintaxis: SSLCACertificatePath ruta/a/certificados/CA
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el directorio en el que se encuentran los CA de los archivos certificados.

SSLCACertificateFile

```
Sintaxis: SSLCACertificateFile nombre_archivo_certificado_CA
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz especifica el archivo en el cual se encuentran los certificados CA.

SSLVerifyDepth

```
Sintaxis: SSLVerifyDepth número
Predeterminado: SSLVerifyDepth 0
Contexto: configuración servidor, host virtual
```

Como un CA puede certificar a otro, es posible terminar con una cadena de certificados CA. Esta directriz especifica la cantidad de certificados CA que tiene que consultar el servidor cuando compruebe los certificados de los clientes. Si no trabaja con certificados de clientes, no hace falta que modifique el valor predeterminado.

SSLVerifyClient

```
Sintaxis: SSLVerifyClient opción_numérica
Predeterminado: SSLVerifyClient 0
Contexto: configuración servidor, host virtual
```

Esta directriz determina la política de certificación que utilizará el servidor. Si quiere trabajar con la certificación para clientes, utilice el valor 2. Si la certificación de clientes es opcional, utilice el 1. Si no la va a emplear, utilice el 0.

SSLFakeBasicAuth

```
Sintaxis: SSLFakeBasicAuth
Contexto: configuración servidor, host virtual
```

Esta directriz traduce un cliente X509 al nombre del usuario que se tiene que utilizar en el proceso de autenticación. No es aconsejable el uso de esta directriz.

SSLLogFile

```
Sintaxis: SSLLogFile archivo_de_registro
Contexto: configuración servidor, host virtual
```

Esta directriz especifica dónde se tiene que escribir la información relativa a las conexiones SSL.

SSLRequiredCiphers

```
Sintaxis: SSLRequiredCiphers cipher1:cipher2:cipher3:...
Contexto: configuración servidor, host virtual, configuración para
cada directorio (.htaccess)
```

Esta directriz especifica una lista de encriptadores o ciphers en la que cada entrada aparece separada por comas. Un cipher es un algoritmo criptográfico como RC4-MD5 y RC4-SHA.

SSLRequireCipher

```
Sintaxis: SSLRequireCipher cipher
Contexto: configuración para cada directorio (.htaccess)
```

Esta directriz añade un cipher a la lista que fue creada con la directriz anterior.

SSLBanCipher

```
Sintaxis: SSLBanCipher cipher
Contexto: configuración para cada directorio (.htaccess)
```

Esta directriz prohibe que se utilice un cipher con el archivo de configuración de un directorio. Con ella se puede rechazar a los clientes que quieran utilizar el encriptador prohibido.

Configurar Apache para que trabaje con Apache-SSL

Ya está listo para crear archivos de configuración tales como httpd.conf, access.conf o srm.conf de Apache. El autor de Apache-SSL, Ben Laurie, distribuye un ejemplo de httpd.conf cuyo código lo podemos encontrar en el listado 15.1. access.conf y srm.conf están vacíos porque en la configuración basada en SSL que ha empleado el autor en este ejemplo únicamente se utiliza httpd.conf.

En este ejemplo de configuración, Ben nos muestra cómo se puede utilizar httpd.conf con conexiones seguras y con conexiones no seguras. En vez de usar el puerto SSL 443 predeterminado, utiliza el puerto 8887 para establecer las conexiones seguras y 8888 para las normales. El ejemplo aún no está listo para probarlo, ya que tiene que sustituir las rutas de los directorios y quitar el símbolo de comentario de unas cuantas directrices.

```
# Ejemplo de configuración para Apache-SSL.
# Copyright (C) 1995,6,7 Ben Laurie

# Por petición popular, este archivo muestra cómo crear dos sitios
# Web, uno seguro (en el puerto 8887) y otro no (en el puerto 8888).

# Es posible que necesite trabajar con uno de estos usuarios
# Web User

# Los servidores SSL tienen que estar aislados.
ServerType standalone

# El puerto SSL por defecto es 443...
Port 8887
Listen 8887
Listen 8888

# Mi directorio de prueba de documentos es
DocumentRoot /u/ben/www/1/docs

# Observe que todas las opciones SSL son aplicables a host virtuales.

# Desactivar SSL. Util cuando se combina con host virtuales.
#SSLDisable

# Determina la ruta del certificado CA de comprobación
# (tiene que estar codificado PEM).
# (además getenv("SSL_CERT_DIR"), creo).
SSLCACertificatePath /u/ben/apache/apache_1.2.5-ssl/SSLconf/conf

# Determina el archivo del certificado CA de comprobación
# (tiene que estar codificado PEM).
```

```
# (además getenv("SSL_CERT_FILE"), creo).
#SSLCACertificateFile /some/where/somefile
#SSLCACertificatePath /u/ben/apache/apache_1.2.5-ssl/SSLconf/conf/
httpsd.pem

# Dirige SSLCertificateFile al certificado PEM-codificado.
# Si el certificado está encriptado, se le pedirá que introduzca
# la contraseña.
# Gracias a -1 se le volverá a pedir.
# Se puede efectuar una comprobación con "make certificate".
SSLCertificateFile /u/ben/apache/apache_1.2.5-ssl/SSLconf/conf/
httpsd.pem

# Si no se puede combinar la llave con el certificado, utilice
# esta directriz para dirigirse al archivo que tiene la llave. Si
# comienza con "/" querrá decir que es una ruta absoluta. En caso
# contrario es relativo al área de certificación. Es decir,
# "<default>/private/<keyfile>".
#SSLCertificateKeyFile /some/place/with/your.key

# Determina SSLVerifyClient a:
# 0 si no hace falta ninguna certificación
# 1 si es posible que el cliente necesite una certificación
# 2 si el cliente tiene que presentar una certificación
# 3 si el cliente puede presentar una certificación, pero no tiene
# que ser un CA válido
SSLVerifyClient 0
# Determina la intensidad de la verificación, antes de decidir que
# no es válida
SSLVerifyDepth 10

# Transforma el cliente X509 en una autorización Basic.
# Es decir, que se utilizan los métodos estándar Auth/DBMAuth
# para controlar el acceso. El nombre del usuario en una versión
# "de una línea" del certificado X509 del cliente.
# Obsérvese que no se obtiene ninguna contraseña del usuario.
# Cada vez que un usuario entre, necesitará esta contraseña:
# xxj31ZMTZzkVA. Si quiere más información, observe el código.
SSLFakeBasicAuth

# Ubicación para la basura que genera SSL. La gran mayoría son
# duplicados del archivo de registro de errores.
SSLLogFile /tmp/ssl.log

# Registro personalizado
CustomLog logs/ssl_log "%t %{version}c %{cipher}c %{clientcert}c"

<VirtualHost scuzzy:8888>
SSLDisable
</VirtualHost>

# Directrices nuevas y sin documentar
#SSLRequiredCiphers
#SSLRequireCipher
#SSLBanCipher
```

```
# Practique con la autorización...
#<Directory /u/ben/www/1/docs>
#AuthType Basic
#AuthName Experimental
#AuthGroupFile /dev/null
#AuthUserFile /u/ben/www/1/users
#<Limit PUT GET>
#allow from all
#require valid-user
#</Limit>
#</Directory>

ScriptAlias /scripts    /u/ben/www/scripts
```

Listado 15.1. httpd.conf

El listado 15.2 muestra un archivo httpd.conf funcional que se usará con un servidor Web seguro llamado blackhole.nitec.com, cuyo directorio de documentos es /www/nitec/secured/htdocs. Para suministrar un servicio regular a través del puerto 80, he creado un host virtual que desactiva SSL empleando para ello la directriz SSLDisable y además utiliza otro directorio de documentos, /www/nitec/public/htdocs.

```
# User y Group para blackhole.nitec.com
User httpd
Group httpd

# Los servidores SSL tienen que ser independientes.
ServerType standalone

# Utiliza el puerto HTTPS predeterminado
Port 443

# También escucha lo que pasa por el puerto HTTP estándar
Listen 80

# Directorio raíz para los documentos de blackhole.nitec.com
DocumentRoot /www/nitec/secured/htdocs

# Directorio en el que se pueden almacenar los certificados CA
SSLCACertificatePath /usr/local/etc/httpd/SSLconf/conf

# Ruta completa de los archivos de certificación CA
SSLCACertificateFile /usr/local/etc/httpd/SSLconf/
conf/httpsd.pem

# Ruta completa de los certificados SSL
SSLCertificateFile /usr/local/etc/httpd/SSLconf/conf/httpsd.pem

# No hace falta que los clientes tengan certificados
SSLVerifyClient 0
```

```
SSLVerifyDepth 0

SSLLogFile /logs/ssl.log

# Como quiero trabajar con los servicios Web regulares (HTTP) en
# el puerto 80, configuro un host virtual para ello.
<VirtualHost 206.171.50.50:80>
DocumentRoot /www/nitec/public/htdocs
ScriptAlias /cgi-bin/   /www/nitec/public/cgi-bin/
SSLDisable
ServerName blackhole.nitec.com
</VirtualHost>
```

Listado 15.2. httpd.conf

Crea una configuración similar para nuestro nuevo httpsd. Si únicamente está interesado en la creación de servidores virtuales seguros y quiere dejar la conexión principal como está (a través del puerto 80) puede utilizar la siguiente sección de la configuración del servidor principal (es decir, fuera del contenedor del host virtual):

```
Port 80
SSLDisable
```

Para crear host virtuales seguros, tendrá que utilizar una configuración similar a ésta:

```
<VirtualHost host.domain.com:443>
DocumentRoot /path/to/secure/pages
ServerAdmin webmaster@host.domain.com
ServerName host.domain.com
SSLCACertificatePath /usr/local/ssl/certs
SSLCACertificateFile /usr/local/ssl/certs/virtual.host.com.pem
SSLCertificateFile /usr/local/ssl/certs/virtual.host.com.pem
SSLLogFile /path/to/ssl.log
</VirtualHost>
```

La directriz SSLDisable que se encuentra en la sección de configuración del servidor principal desactiva todas las conexiones SSL menos la del host virtual determinado. Ahora ya está listo para comprobar la seguridad de su sitio Web.

Comprobar la seguridad de un servidor

Antes de empezar con las pruebas, asegúrese de que tiene instalada la última versión de Netscape Navigator o Microsoft Internet Explorer. Las versiones antiguas no pueden trabajar con SSL.

Para comprobar la seguridad de un sitio Web tendrá que dirigir su explorador a:

```
https://your.secured.web.site.domain/
```

Recuerde que tiene que utilizar https:// en vez de http:// cuando quiera trabajar con un URL seguro. Con la configuración anterior se utiliza el siguiente acceso:

```
https://blackhole.nitec.com/
```

Para disfrutar de un acceso regular, utilice:

```
http://blackhole.nitec.com/
```

Si ha creado un verificado que se genera automáticamente al ejecutar make certificate e intenta acceder a su sistema seguro, aparecerá un mensaje de advertencia en su explorador. Por ejemplo, cuando se accede a https://blackhole.nitec.com, Netscape Navigator le muestra un mensaje en el que se comenta que el sitio al que se intenta acceder está protegido con un sistema de encriptación, tal y como se puede ver en la figura 15.5.

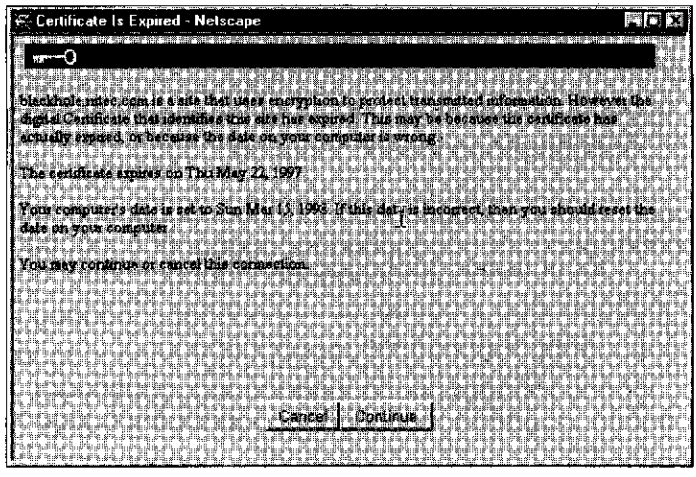


Figura 15.5. Mensaje de advertencia sobre un certificado generado automáticamente

A menos que obtenga un CA comercial, siempre que trate de acceder a un sitio protegido con uno de estos sistemas de encriptación, aparecerá un mensaje de advertencia de este tipo.

Cuando se pasa la ventana en la que se muestra el aviso del que le he hablado, el explorador establecerá entonces un canal seguro de comunicación. La figura 15.6 nos muestra una de estas sesiones para blackhole.nitec.com.

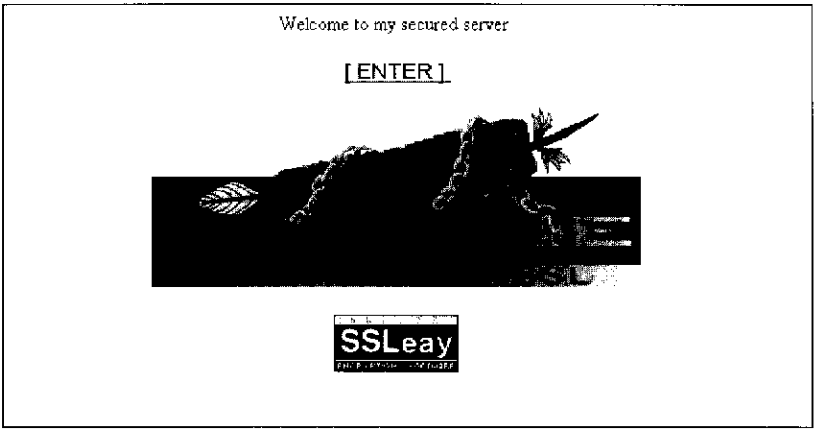


Figura 15.6. Una sesión segura

En Netscape Navigator aparece el icono de un candado en la parte inferior izquierda de la pantalla. Cuando se cierra quiere decir que la conexión establecida es segura.

Si tiene interés por ver qué ocurre cuando un cliente (como un servidor Web) se conecta con un servidor seguro, puede ejecutar la aplicación s_client que se incluye en el paquete de SSLeay Para blackhole.nitec.com. El comando se ejecución sería el siguiente:

```
ssleay s_client -host blackhole.nitec.com -port 443
```

En el listado 15.3 tiene el código de esta aplicación.

```
CONNECTED(00000003)
depth=0 /C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
verify error:num=18:self signed certificate
verify return:1
depth=0 /C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
verify return:1
---
Certificate chain
 0 s:/C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
   i:/C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
```

```
--
Server certificate
---BEGIN CERTIFICATE---
MIICezCCAeQCAQAwDQYJKoZIhvcNAQEEBQAwgYUxCzAJBgNVBAYTAlVTMRMwEQYD
VQQIEwpDYWxpZm9ybmlhMRMwEQYDVQQHEwpTYWNyYW1lbnRvMQ4wDAYDVQQKEwVO
aXRlYzEcMBoGA1UEAxMTYmxhY2tob2xlLm5pdGVjLmNvbTEeMBwGCSqGSIb3DQEJ
ARYPa2FiaXJAbml0ZWMuY29tMB4XDTk3MDQyMjA4NDAwNFoXDTk3MDUyMjA4NDAw
NFowgYUxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRMwEQYDVQQH
EwpTYWNyYW1lbnRvMQ4wDAYDVQQKEwVOaXRlYzEcMBoGA1UEAxMTYmxhY2tob2xl
Lm5pdGVjLmNvbTEeMBwGCSqGSIb3DQEJARYPa2FiaXJAbml0ZWMuY29tMIGfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQDVuX1Kqg7p9eS1xV1RyglaLTSZ+Ge+CESR
HUBUQiGuiU+eo97CaKze4B19AM4ZV3xnloJrS8LKTnLmkJ95A++6ymrhIVIVgmXZ
DgkczjQ2LKInWCJ9QZntoEu/1mPUY2KnXbbQo5v/gtf/J6tbhuRJKtLgz2dK4CWM
/OPQzzrYjQIDAQABMA0GCSqGSIb3DQEBBAUAA4GBAIT2FA3Ce7Q-tnu3KjBj7NLT
VDG5YsEyGXZKBEWJEzpAzOywncndiTwwdUgocYl/EADidHsrgiYDsYPD3oVFkWOw
cvHLwxBE++6F1cLkRty3C/bfgrdJWWPsrd14nZq3Vy8y/xFQkZJ9Ob0cpo6ac0qm
5HrrSatn6DBaWuDXdEI5
---END CERTIFICATE---
subject=/C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
issuer=/C=US/ST=California/L=Sacramento/O=Nitec/
CN=blackhole.nitec.com/Email=kabir@nitec.com
SSL handshake has read 813 bytes and written 317 bytes
--
New, SSLv3, Cipher is RC4-SHA
SSL-Session:
    Cipher    : RC4-SHA
    Session-ID:
C26196B7FC9C2DBE664FC5C55AD4C89887BC9703B98F944136285JE428C4D67F
    Master-Key:
303EBE61E04175F81C605E02A2CC264022F6E55F7714FA32DB3A37852A4AD6C8
5B2748ED3FAD06FDD733CDD098B443D6
    Key-Arg   : None
    Start Time: 861699068
    Timeout   : 7200 (sec)
--
```

Listado 15.3. Salida de s_client que contacta con blackhole.nitec.com a través del puerto 443

La salida anterior contiene mucha información. El servidor proporciona un nombre relacionado con la salida (en formato abreviado). Los nombres que se distinguen se definen con el estándar X.509, que define los campos, los nombres de los campos y las abreviaturas que se utilizan para referirse a ellos. En la tabla 15.1 se muestra una lista con los campos que aparecen en el ejemplo anterior.

CA define qué campos son necesarios y cuáles son opcionales. Incluso puede haber ciertas restricciones en el contenido de cada campo. Por ejemplo, los exploradores de Netscape requieren que el contenido de CN sea una expresión regular.

Tabla 15.1. Nombres de campos distinguidos

Campo	Abreviatura	Descripción	Ejemplo
Nombre Común	CN	Nombre que se está certificando. Nombre completo del dominio del sistema.	CN=blackhole.nitec.com
Organización o compañía	O	Nombre asociado con la organización.	O=Nitec
Unidad de la organización	OU	Nombre asociado con esta unidad organizativa, por ejemplo, un departamento.	OU=no se ha especificado ninguno
Ciudad/ Localidad	L	Ciudad en la que se encuentra CN.	L=Sacramento
Estado/ Provincia	ST	Nombre del estado o la provincia en la que se encuentra CN.	ST=California
País	C	País en el que se encuentra CN.	C=EE.UU. (USA)

El formato binario de un certificado se define utilizando ASN.1. Esta notación define cómo se han de especificar los contenidos y las reglas de codificación definen cómo se traduce la información a formato binario. ASN.1 se puede codificar de varias formas, pero el estándar que se está utilizando, DER, es de lo más sencillo y como resultado ofrece un certificado binario compacto. Cuando se intercambia correo electrónico, el certificado binario se suele codificar con Base64, con lo que se obtienen las líneas ---BEGIN CERTIFICATE--- y ---END CERTIFICATE---.

También aparece la información sobre las cadenas de certificación. Cuando CA se hace cargo de una certificación de otro CA, se crea una cadena y el servidor informará sobre ella al cliente, para que éste pueda decidir si quiere investigar los certificados que se mencionan en dicha cadena. En este ejemplo en particular, no hay ninguna cadena presente.

Después de mostrar la salida que aparece en el listado 15.3, el programa s_client sigue conectado al servidor. Así, si introduce la siguiente petición:

```
GET / HTTP/1.0
```

seguida de una línea en blanco, obtendrá la salida del listado 15.4.

```
HTTP/1.1 200 OK
Date: Tue, 22 Apr 1997 08:51:56 GMT
Server: Apache/1.2.5 Ben-SSL/1.13
Connection: close
Content-Type: text/html

<HTML><HEAD>
<TITLE>Index of /</TITLE>
</HEAD><BODY>
<H1>Index of /</H1>
<UL><LI> <A HREF="/"> Parent Directory</A>
<LI> <A HREF "development"> development</A>
<LI> <A HREF="future"> future</A>
<LI> <A HREF="public"> public</A>
</UL></BODY></HTML>
read:errno=0
```

Listado 15.4. Salida obtenida al efectuar una petición GET a un servidor seguro

No hay ningún archivo índice en el directorio superior del servidor seguro blackhole.nitec.com, por lo que se genera uno dinámicamente.

Si puede hacer pruebas parecidas en su servidor Apache-SSL, creará con éxito un servidor Web seguro.

Obtención de un certificado CA

Ahora, todo lo que tiene que hacer es conseguir un certificado que pueda reconocer el explorador Web de un CA. Por desgracia, no todos los servidores especializados en la distribución de certificados pueden trabajar con Apache-SSL, debido a sus políticas de administración.

Aquí tiene unos cuantos CA que le suministrarán un certificado para Apache-SSL.

- Thawte Consulting, en www.thawte.com/certs/server/request.html
- CertiSign Certificadora Digital Ltda., en www.certisign.com.br
- IKS GmbH, en www.iks-jena.de/produkte/ca/
- Uptime Commerce Ltd., en www.uptimecommerce.com
- ID-Pro GmbH, CA-Projekt, en www.id-pro.de/security/CA

Unicamente Thawte y VeriSign (que no suministra certificados para Apache-SSL) pueden trabajar con todas las versiones de Netscape Navigator/Communicator y Microsoft Internet Explorer. Esta situación puede cambiar, ya que cada vez son más los usuarios que trabajan con Apache-SSL y otros

servidores basados en esta plataforma. Es decir, que cada vez es mayor la presión que se ejerce sobre CA. La mejor aproximación será identificar qué CA puede trabajar con la mayoría de los exploradores y averiguar si alguno de ellos quiere firmar su certificado. En el momento en que escribimos estas líneas, la única opción disponible era Thawte.

Stronghold

Si no quiere tener ningún problema a la hora de configurar SSLeay, parchear los archivos fuente de Apache con Apache-SSL y con fuentes extra y trabajar con certificados de generación automática, lo mejor que puede hacer es conseguir una versión comercial de un servidor Apache seguro, llamado Stronghold.

Stronghold, desarrollado por C2NET Technology, se basa en el último servidor de Apache y SSL. Dispone de un asistente de configuración Web muy aparente que le permite administrar su servidor desde cualquier punto de la Red. Si tiene previsto utilizar la Red para trabajar con comercio electrónico, conviene que tenga en consideración este producto.

Puede obtener una copia de evaluación de este producto de C2NET. Visite el sitio www.c2.net e inscríbase para obtener una copia de evaluación del servidor Stronghold. Al registrar su copia de evaluación se le pedirá que introduzca la dirección IP del host que va a utilizar como servidor Stronghold. Cuando se completa el proceso de registro, se le enviará un mensaje de correo electrónico a su dirección de e-mail (que habrá suministrado durante el proceso anterior). Este mensaje contiene el nombre de usuario/contraseña que le permitirá obtener una copia de Stronghold. También contiene la clave de registro que tendrá que utilizar durante la instalación, así como un URL del sitio desde el cual se puede bajar la copia del servidor.

Creo que Stronghold es una solución ideal para aquellos que quieren pasarse el menor número posible de noches sin dormir, tratando de hacer que SSL funcione con Apache-SSL. La compañía suministra documentación que puede resultar de utilidad. En general, si esta opción le parece demasiado cara, pero tampoco quiere trabajar con Apache-SSL, puede probar el módulo Raven SSL para Apache. Búsquelo en:

```
http://raven.covalent.net/
```

Técnicamente hablando, el módulo Raven SSL, desarrollado por Covalent Technologies, no es exactamente igual que otros módulos de Apache (tal como mod_cgi o mod_rewrite), pero desde el punto de vista del usuario final,

funciona exactamente igual que un módulo corriente. La empresa distribuye una versión precompilada del servidor Apache que incluye este módulo. También incluye un archivo fuente de Apache parcheado de antemano, por lo que volver a compilar un nuevo servidor utilizando los nuevos módulos es algo tan sencillo como quitar un símbolo de comentario del archivo Configuration. Obsérvese que Covalent Technologies no incluye el código fuente de su módulo Raven ni los algoritmos de encriptación RSA debido a las restricciones relacionadas con el código criptográfico impuesto en los EE.UU. En la actualidad, VeriSign no distribuye certificados para los servidores Apache basados en Raven, pero Thawte Consulting sí (y además es más barato que Stronghold).

Instalación de Stronghold

La instalación de Stronghold es muy sencilla. Todo lo que hay que hacer es desempaquetar el software que se baje del sitio Web y ejecutar el script INSTALL.sh. Se le pedirá información como por ejemplo dónde quiere instalar Stronghold, qué puertos desea utilizar con los servicios HTTP y HTTPS y a nombre de qué servidor se extenderá la clave de la licencia. En la mayoría de las preguntas se incluye un valor predeterminado que debería ser válido en la mayoría de los casos.

Nota: no tiene que ejecutar un servidor Apache independiente para los servicios HTTP. Stronghold puede hacerse cargo de los servicios HTTP seguros y no seguros.

Si acepta los valore predeterminados, utilizará los siguientes puertos:

80 Servicio HTTP estándar. Ejemplo: http://suservior/

443 Servicio HTTP basado en SSL. Ejemplo: https://suservidor/

444 Administrador de configuración. Ejemplo: https://suservidor:444/

445 Interfaz Directa (localhost). Ejemplo: https://suservidor:445/

El script de instalación le pide que, si trabaja con shells csh/tcsh, añada lo siguiente al archivo .cshrc:

```
sentenv SSLTOP /path/to/stronghold/dir/ssl
setenv PATH ${SSLTOP}/bin:$PATH
```

Si utiliza los shell sh/bash, añada:

```
$ SSLTOP=/path/to/stronghold/dir/ssl
$ PATH=${SSLTOP}/bin:$PATH
$ export SSLTOP PATH
```

El script de instalación también cuenta con una opción que le permite convertir un certificado/clave de Netscape Commerce o crear un nuevo par. Si nunca ha establecido ninguna transacción comercial con un servidor seguro utilizando Netscape, tendrá que crear un nuevo par certificado/clave. En este caso, se le pedirá que introduzca el tamaño de la clave (o llave) en bits. Puede escoger cualquier valor comprendido entre 512 y 4096 (4K) bits.

Mi consejo es utilizar 1024 bits porque los tamaños más pequeños no son seguros y los más grandes ralentizan mucho el servidor. Además, algunas versiones de los exploradores más populares no pueden trabajar con claves demasiado grandes. Una vez que haya escogido el tamaño de la clave, se le pedirá que escriba unas cuantas letras al azar. No se preocupe sobre lo que escribe. La finalidad de esta petición es medir el tiempo que transcurre entre pulsaciones, independientemente de las teclas que se pulsen. Siga tecleando hasta que oiga un "bip" que le indicará que el contador ha llegado a cero. Al final se encontrará con que se ha generado la llave.

Ahora tiene que tomar una decisión más importante: se le preguntará si desea solicitar un certificado CA. Tenga en cuenta que Stronghold cuenta con un certificado gratuito de Thawte CA. Pero únicamente lo podrá conseguir después de pagar por el software. Si todo lo que quiere es probar el software durante una temporada hasta que decida si lo quiere comprar, mi consejo es que no pida ningún certificado CA hasta que vaya a comprarlo. Siempre puede generar uno con la utilidad genreq que se distribuye con Stronghold. En este caso, conteste No. A continuación se le solicita la información gracias a la cual podrá crear certificados para el periodo de evaluación.

La llave privada que se crea en este proceso se puede proteger por medio de una contraseña. Se le pide que la introduzca antes de comenzar a funcionar. De todas formas, si escoge la opción de uso de contraseña, tendrá que introducirla siempre que inicie el servidor. Es decir, que no podrá utilizar las facilidades rc.d o rc.local para iniciar automáticamente el servidor.

Por último, el script de instalación inicia el servidor seguro y debería ser capaz de acceder a él a través del URL:

```
https://suservidor/
```

Su explorador le comentará que no reconoce el certificado. De todas formas, en el mismo momento en que instale el certificado CA real en su sistema,

desaparecerá este aviso. Lo primero que verá será la pantalla que se muestra en la figura 15.7.

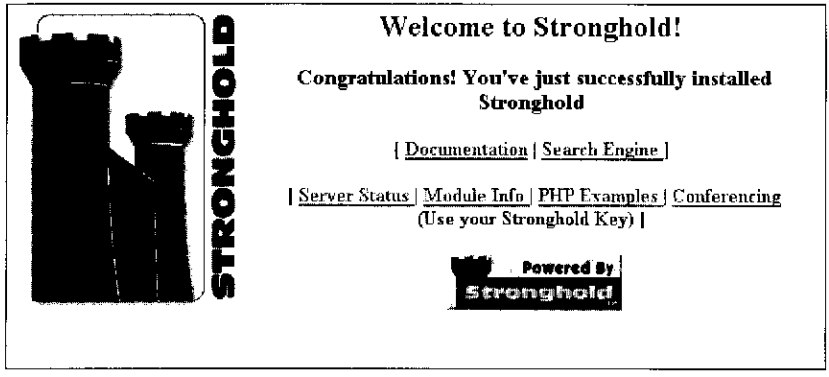


Figura 15.7. Bienvenida a la pantalla de Stronghold

Notar cuando instale Stronghold.con el script INSTALL.sh, también instalará Python, el sistema de conferencia de Focus y el motor de búsqueda SWISH/WWWWAIS.

Ahora que ha instalado Stronghold, puede pasar a crear los contenidos de su espacio seguro. Para administrar el servidor Web asegurado, puede utilizar el administrador Configuration Manager.

Uso de Configuration Manager

Stronghold se puede configurar desde el administrador Configuration Manager que se basa en la Red, o bien a través de los archivos usuales de configuración, como http.conf, access.conf y srm.conf. Para acceder a Configuration Manager desde cualquier punto de la Red, tendrá que dirigir su explorador Web al Url del sitio asegurado y entrar a través del puerto 444 (si no trabaja con los valores predeterminados por el script INSTALL.sh, tendrá que introducir sus valores personalizados).

Por ejemplo, para acceder al administrador Configuration Manager de mi servidor Stronghold tengo que utilizar el URL:

```
https://secured.mitec.com:444/
```

Para protegerme de los abusos, Configuration Manager utiliza un sistema de autenticación basado en nombre de usuario/contraseña. Tiene que utilizar

los que ha introducido durante el proceso de instalación. Una vez que se autentique a sí mismo como administrador de la configuración, accederá a una página similar a la que se muestra en la figura 15.8.

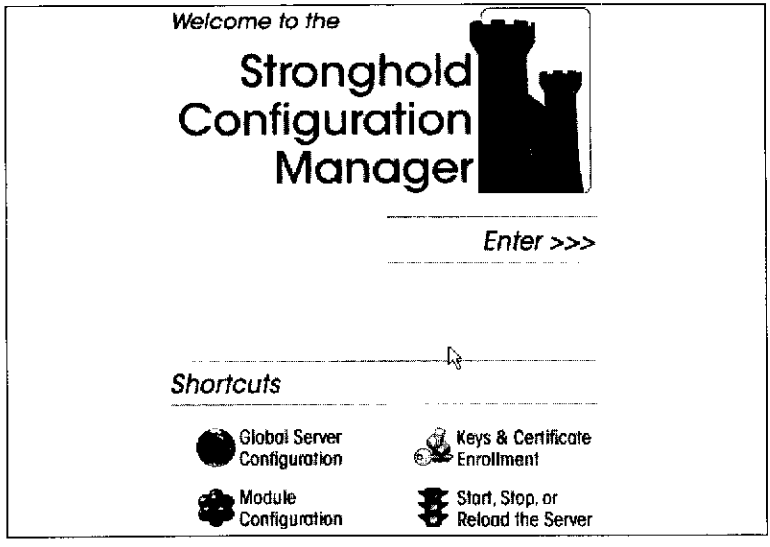


Figura 15.8. Pantalla principal de Configuration Manager

Desde el índice que tiene en esta página podrá acceder rápidamente a:

- Global server configuration (Configuración global del servidor).
- Keys and certificate enrollment (Inscripción de claves y certificado).
- Module configuration (Configuración del módulo).
- Starting, stopping, or reloading the server (Iniciar, parar o volver a cargar el servidor).

Puede hacer clic en Enter (Entrar) y verá una página similar a la que se muestra en la figura 15.9.

Desde aquí, puede acceder a lo siguiente:

- Global server configuration (Configuración global del servidor).
- Virtual host configuration (Configuración del host virtual).
- Per-directory configuration (Configuración para directorio).
- Modules configuration (Configuración de módulos).
- Finalize configuration (Finalizar configuración).

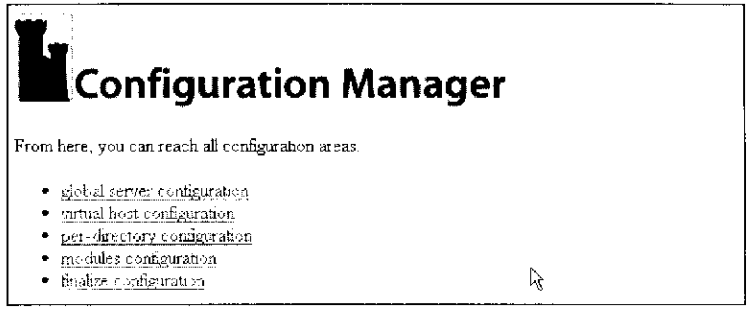


Figura 15.9. Página de configuración global de Configuration Manager

La última opción le permite enviar cualquier añadido o modificación que le haga al servidor. Cuando hace clic sobre esta opción accederá a la página que aparece en la figura 15.10.

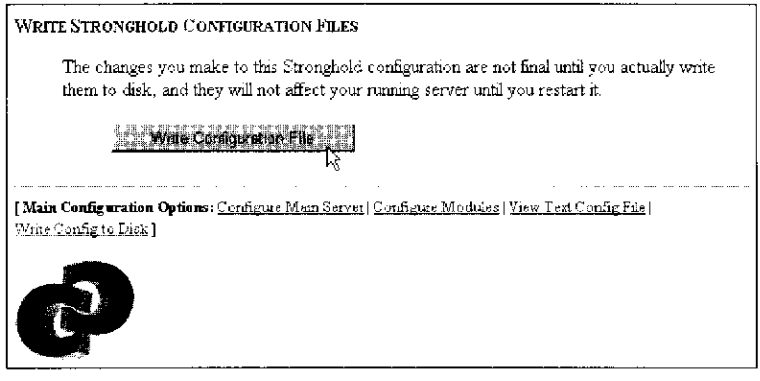


Figura 15.10. Guardar cambios

Cuando esté listo para enviar su nueva configuración, haga clic en el botón que tiene la etiqueta Write Configuration File (Escribir archivo de configuración) para guardar los cambios en los archivos del disco. Para terminar tendrá que reiniciar el servidor y los cambios serán efectivos. Dispone de una página desde la cual puede iniciar (Start), detener (Stop) o volver a cargar (Reload) el servidor, tal y como se puede apreciar en la figura 15.11.

Nota: si tiene un explorador que puede trabajar con SSL, en los sistemas seguros, podrá acceder a Configuration Manager a través de una interfaz directa. Para ello tendrá que entrar a través del URL https://localhost:445/. Si en el proceso de instalación escogió un puerto distinto, tendrá que modificar el número del puerto. Esta solución es muy útil

cuando trabaja con sesiones TELNET y quiere efectuar algún cambio rápidamente. Para estas situaciones, lo conveniente es utilizar un explorador Lynx que pueda trabajar con SSL.

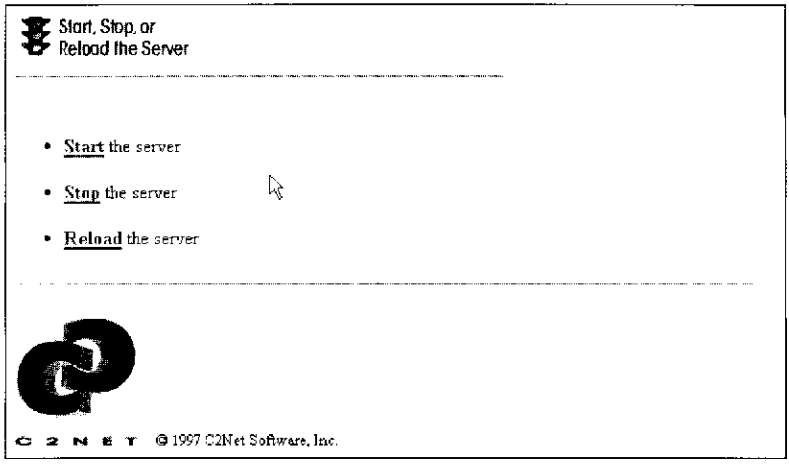


Figura 15.11. Pantalla desde la cual se puede controlar el estado del servidor

Hacerse con un certificado CA

Cuando decida utilizar Stronghold como servidor Web y quiera adquirir un certificado CA, puede hacer una copia de seguridad de la clave que tiene guardada en el directorio privado $SSLTOP/ y ejecutar:

```
genkey <hostname>
```

Esta utilidad genera un par clave para el nombre del host especificado, invoca a genreq para que genere una petición de certificado (CSR) y llama a gencert para que cree un certificado temporal que se puede utilizar hasta que responda CA. El certificado temporal se firma con la llave privada de su sistema. De todas formas, antes de ejecutar esta utilidad, debería determinar qué CA quiere utilizar. Stronghold ofrece gratuitamente un certificado de Thawte Consulting (www.thawtw.com) cuando se compra el paquete del servidor. Este certificado será quien le haga el trabajo, a menos que tenga alguna razón para utilizar otro CA, como VeriSign (www.verisign.com/).

Si desea más información sobre las diferencias de precios que hay entre ambas compañías, también lo puede consultar. La información sobre los precios de VeriSign la encontrará en www.verisign.com/enroll.s/payment.html y

los acuerdos relacionados con su servidor los tiene en www.verisign.com/enroll.s/legal.html. Thawte también publica una tabla de precios en https://www.thawte.com/pricing.html. Si quiere utilizar otro CA, asegúrese de que sea uno con el que puedan trabajar sus clientes.

Después de enviar su CS al CA y de suministrar los papeles que le ayudarán a autenticar su organización, todo lo que puede hacer es esperar a que el CA se ponga en contacto con usted para entregarle el certificado de su sitio Web. Cuando los reciba tendrá que utilizar la utilidad getca de Stronghold para instalarlo.

Para instalar el certificado de un sitio Web, dé los siguientes pasos:

1. Guarde el certificado que se le adjunta con el mensaje de correo electrónico en un archivo que se encuentre en una ubicación temporal, tal como /tmp/01.pem.
2. Ejecute getca, especificando el nombre del host al que pertenece el certificado y el nombre del archivo temporal:

```
getca hostname < /tmp/01.pem
```

3. Reinicie el servidor usando el script reload o introduciendo el comando:

```
kill -HUP `cat /path/to/httpd.pid'
```

Se le pedirá que introduzca un par clave/certificado para cada uno de los host que utilizan SSL. Se incluye el nombre de los host virtuales que vaya a utilizar. Tiene que pagar por cada certificado que solicite, por lo que si tiene muchos sitios con SSL, le puede salir algo caro. En este caso es conveniente volver a organizar sus sitios virtuales para que se dirijan a un único host para acceder a SSL.

Por ejemplo, puede crear un host llamado secured.sudominio.com y tener un único par clave/certificado para él. Si desea que el servidor virtualuno.sudominio.com realice las transacciones de seguridad, todo lo que tiene que hacer es crear un directorio que se encuentre dentro del dominio de secured.sudominio.com y colocar allí todos los archivos del sitio virtual. Los documentos HTML que se encuentran en estos sitios virtuales han de disponer de un enlace que los comunique con un área segura para poder utilizar vínculos como:

```
<A HREF=https://secured.yourdomain.com/virtualhost-directory/
filename>A secured Page</A>
```

Asegúrese de que todos sus visitantes tienen una forma de volver al sitio virtual sin asegurar en el momento en que terminan de interactuar con los

documentos o script asegurados. Para ello se añade un enlace que permita regresar al sitio virtual. Para reducir la posible confusión que se pudiese crear entre los visitantes de un sitio virtual, puede dejar los campos opcionales de su petición de certificado en blanco.

Pero, si utiliza un servidor para interactuar con una red muy amplia, es posible que tenga que crear su propio CA.

Configurar un CA privado

Un CA privado le permitirá firmar sus propios certificados. Hay dos razones por las que puede querer disponer de un CA privado:

- Para satisfacer su curiosidad.
- Su organización tiene una gran necesidad de seguridad y tiene que utilizar SSL entre sus distintos departamentos.

En el segundo caso, asegúrese de que el sistema que decida construir tenga el CA físicamente en una ubicación segura y de que, preferiblemente, no esté conectado con ninguna red. De esta forma se asegurará de que nadie puede violar su CA. Si no toma las precauciones necesarias para proteger su CA y un intruso se hace con él, podrá crear certificados falsos y utilizarlos para asegurar sus documentos o su software.

El servidor Web de Stronghold dispone de una serie de herramientas que permiten configurar un CA básico. Obsérvese que para crearlo necesitará el paquete SSLeay que ha instalado con Apache-SSL. De todas formas, en la siguiente sección utilizaré el Stronghold que se ha instalado con las utilidades SSLeay para crear un CA privado.

Configurar un CA privado

El primer paso en la configuración de un CA privado es crear un par clave/certificado para el CA. Puede utilizar el script makeca que se encuentra en el directorio $SSLTOP/ para crear este par. Utilizará la llave privada que ha generado CA para firmar los certificados de su servidor. Los sistemas de sus clientes pueden usar estos certificados que ha creado para CA con el fin de comprobar la firma del CA que se encuentra en los certificados del servidor.

El script makeca guarda el certificado CA creado en $SSLTOP/CA/cacert.pem. Netscape Navigator requiere que este certificado esté en formato DER. Para convertir el certificado PEM a formato DER, utilice el programa x509:

```
x509 -outform DER < $SSLTOP/CA/cacert.pem > /path/to/document/root/
your_organization.cacert
```

El archivo resultante, your_organization.cacert, se guardará en el directorio raíz de documentos de su servidor Web. Puede guardarlo en otra ubicación si desea que los exploradores de sus clientes puedan acceder a él. La extensión del archivo, .cacert, es importante porque está asociada con el tipo MIME application/x-x509-ca-cert. Asegúrese de comprobar que el archivo mime.type, que se encuentra en el directorio superior de Stronghold, tiene la línea:

```
application/x-x509-ca-cert    cacert
```

Es la encargada de asociar la extensión cacert con el certificado CA y además les dice a los exploradores Web que descarguen e instalen el certificado. Por ejemplo, cuando un usuario de Nitec dirige su explorador Netscape Navigator a:

```
https://blackhole.nitec.com/nitec.cacert
```

se encontrará con el cuadro de diálogo que aparece en la figura 15.12.

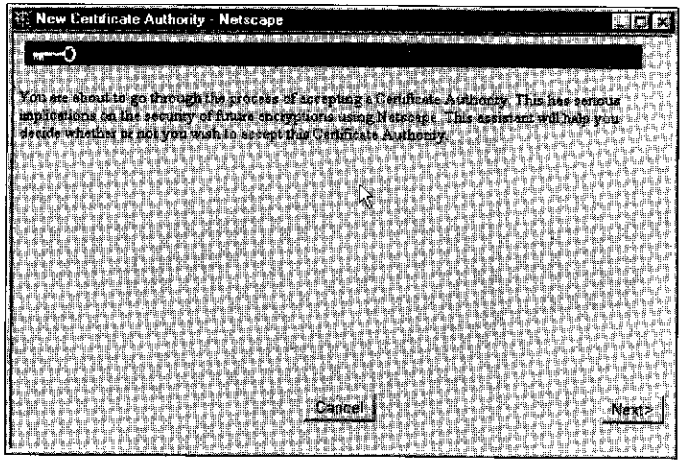


Figura 15.12. Instalación de un certificado CA privado en un explorador

Unas cuantas pantallas más adelante aparece una que da información sobre el certificado. El usuario puede ver los detalles del certificado, tal y como muestra la figura 15.13.

En el proceso, se le da al usuario la oportunidad de decidir qué puede certificar con CA. Por ejemplo, puede decidir permitir que este CA privado certifique otros sitios Web, usuarios de correo electrónico o desarrolladores de software. En la figura 15.14 se puede ver que el usuario ha escogido utilizar el certificado CA para todo.

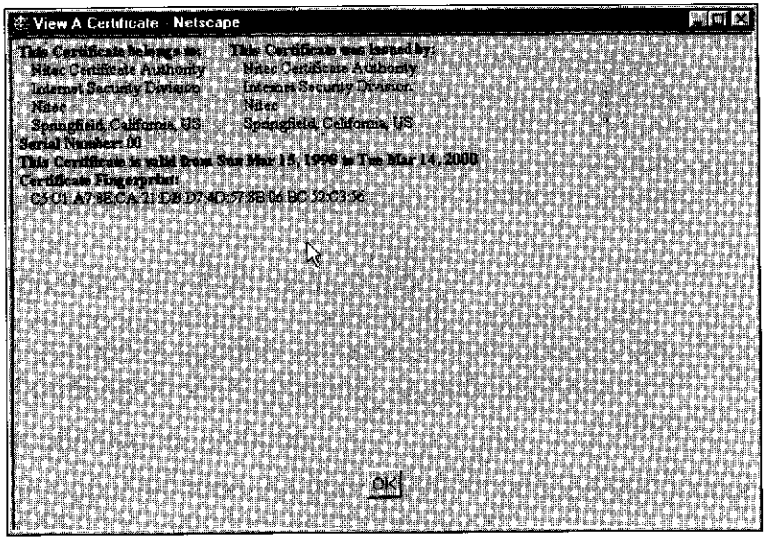


Figura 15.13. Detalles del certificado de Nitec

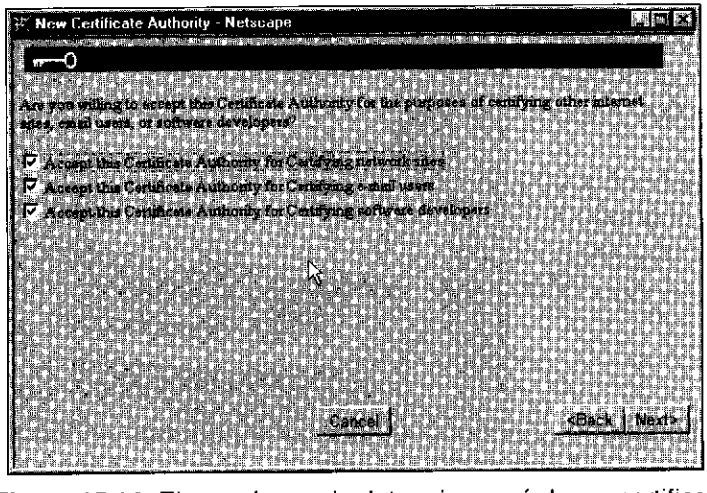


Figura 15.14. El usuario puede determinar qué desea certificar

Por último, cuando el usuario acepta un CA privado, tendrá que asignar un alias para CA. En la figura 15.15 se puede ver que el escogido es Nitec CA.

Una vez que el usuario ha completado estos pasos, el certificado CA aparecerá en la lista de certificados CA oficiales. En la figura 15.16 se puede ver que Nitec CA aparece entre los certificados CA de otras empresas.

El usuario puede decidir si desea editar, borrar o comprobar el certificado CA y utilizarlo.

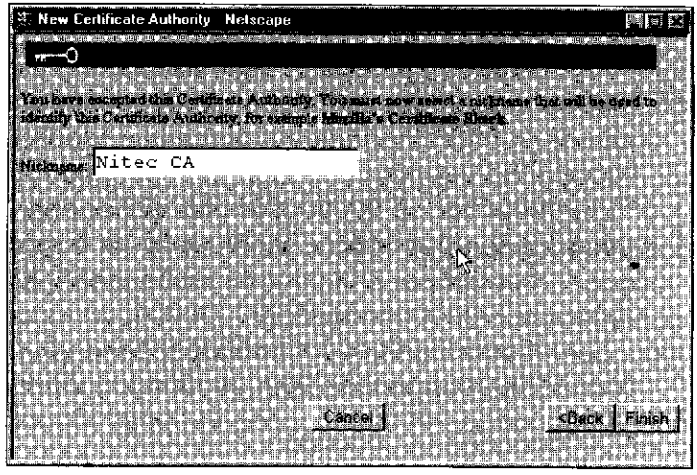


Figura 15.15. Selección de un alias para el CA privado

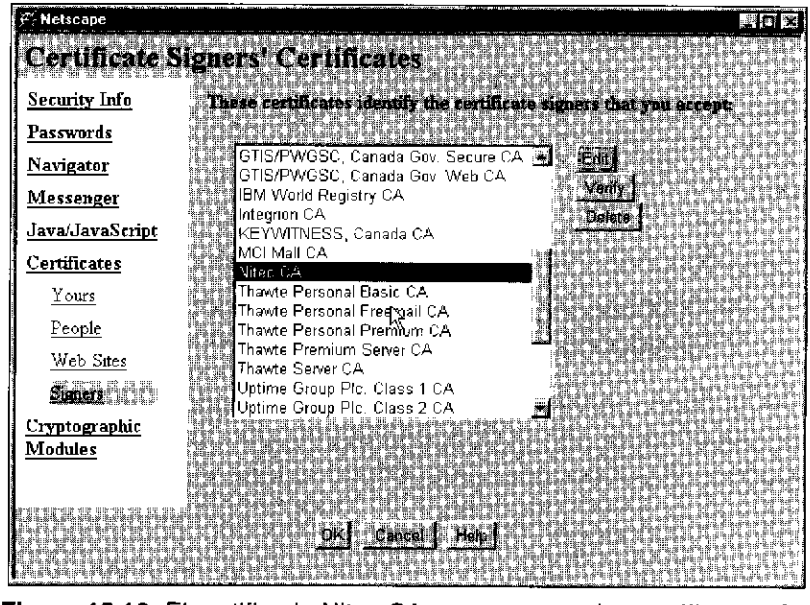


Figura 15.16. El certificado Nitec CA aparece entre los certificados CA de otras empresas

Una vez instalado el certificado en los exploradores Web de los usuarios, éstos no avisarán a sus propictarios si desconocen el CA (a menos que durante el proceso de instalación del certificado se especificase lo contrario).

El siguiente paso en la creación de un CA privado es configurar la librería SSLeay.

Configurar SSLeay

Por defecto, la política de seguridad SSLeay puede ser demasiado holgada para las necesidades de CA privado. Por ejemplo, la configuración predeterminada de SSLeay permite que el servidor certifique las peticiones CommonName (CN), dejando el resto de opciones como alternativas.

Si utiliza con un CA privado para que su organización cuente con un entorno de seguridad estricto, puede modificar la sección de políticas de SSLeay que se encuentra en el archivo de configuración. Este fichero, ssleay.conf, se encuentra en el directorio $SSLTOP/CA.

Localice la línea que determina la política que se va a utilizar. Como la política predeterminada es policy_anything, el aspecto que tiene es el siguiente:

```
policy = policy_anything
```

Puede utilizar dos políticas dentro del archivo de configuración. La predeterminada es la siguiente:

```
# For the 'anything' policy
 [ policy_anything ]
countryName              = optional
stateOrProvinceName      = optional
localityName             = optional
organizationName         = optional
organizationalUnitName   = optional
commonName               = supplied
emailAddress             = optional
```

Como se puede ver, todo es opcional menos CommonName (CN). La otra política, llamada policy_match, tiene la siguiente definición:

```
# For the CA policy
[ policy_match ]
countryName              = match
stateOrProvinceName      = match
organizationName         = match
organizationalUnitName   = optional
commonName               = supplied
emailAddress             = optional
```

Esta política requiere que el servidor certifique las peticiones en las que coincide el nombre del país, estado o provincia y organización. Además, en la petición tiene que aparecer CN. Ahora puede modificar cualquiera de las dos políticas anteriores (policy_anything o policy_match) o crear una nueva.

El método más sencillo para crear una nueva es copiar una de las políticas anteriores y sustituir sus valores por otros nuevos. Los valores que se pueden utilizar son:

- match. Los contenidos del campo tienen que coincidir con los del campo correspondiente de CA.
- optional. No hace falta que aparezca el campo en la petición del certificado.
- supplied. El campo tiene que aparecer en la petición.

Los parámetros de campo de la política se corresponden con los campos de la petición y deben tener uno de estos tres valores. Además, tendrá que asegurarse de que su nueva política tiene un nombre nuevo (que deberá introducir entre paréntesis). Si decide utilizar otra política distinta a la predeterminada, no se olvide hacer las sustituciones pertinentes.

Si su política se llama my_policy, usará la siguiente línea para definirla:

```
policy = my_policy
```

Normalmente, los parámetros de la política son los únicos campos del archivo ssleay.conf que tiene que editar. Si quiere editar otros campos, no olvide leer con atención los comentarios.

Ahora ya está listo para aceptar las peticiones de certificado (CSR) y firmarlas.

Firmar certificados

La verdad es que tiene que establecer otra política antes de comenzar a firmar certificados. Esta política será la que determine qué pruebas, y en qué formatos, pedirá para procesar un certificado. Su finalidad es verificar el origen de la petición CSR.

Una vez que ha recibido la petición CSR a través del correo electrónico o de un script CGI que se encuentre en una página Web, puede guardar su contenido en un archivo temporal y utilizarlo como argumento del script sign_csr. Por ejemplo:

```
sign_csr /tmp/a_csr_request_file
```

Se le pide que introduzca su contraseña CA, que configuró en el proceso de creación de CA, cuando utilizaba el script makeca. La petición sign.csr comprueba la firma que se encuentra en la petición, imprime los campos solicitados y le pide que decida si quiere firmar esta petición. Si cree que la información suministrada es suficiente y coincide con los otros formularios de autenticación (que se suelen encontrar en papel) que le envió el solicitante, puede firmar la petición. El script le pedirá que envíe la firma.

Cuando introduce **Y**, el script imprime el contenido del nuevo certificado y lo guarda en el directorio $SSLTOP/CA/new_certs.

Advertencia: el directorio $SSLTOP/CA/new_certs no se creó con mi versión de Stronghold, por lo que tuve que crearlo a mano.

El nombre del archivo de cada nuevo certificado es un número de una serie formada por todos los certificados firmados que utilizan este CA. Por ejemplo, el primer certificado que firme se guardará en el archivo 01.pem. El siguiente, en el 02.pem, etc. Una vez que ha creado el certificado, se lo envía al usuario para que lo instale.

He creado un CSR para blackhole.nitec.com y lo he firmado utilizando Nitec CA. Instalé el certificado resultante en el servidor Stronghold. Cuando accedí al servidor con un explorador Web, al que ya había instalado el certificado Nitec CA, no me apareció ningún mensaje de aviso en el que se advirtiese que se desconocía el certificado del servidor. La verdad es que me lo esperaba, porque así se lo indiqué a Netscape Navigator. Al pedirle información al explorador sobre el certificado, me mostró la pantalla que aparece en la figura 15.17.

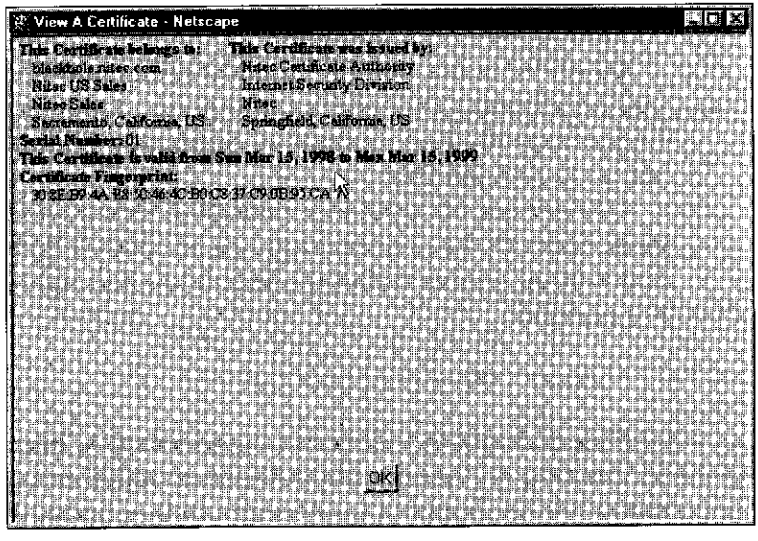


Figura 15.17. Certificado de Nitec CA del servidor blackhole.nitec.com

Para crear la configuración de esta maravillosa firma automática, simplemente tiene que seguir estos pasos:

1. Crear una CSR para blackhole.nitec.com.
2. Crear un CA llamado Nitec CA.
3. Hacer que los exploradores Web de los clientes se bajen e instalen el certificado de Nitec CA.
4. Tomar la CSR que ha firmado Nitec CA.
5. Instalar el certificado que ha firmado el servidor Nitec CA en blackhole.nitec.com.

Como estos pasos se tienen que dar dentro de la misma organización, el proceso es prácticamente plano y se completa en muy poco tiempo. De todas formas, en el mundo real hay que establecer un procedimiento para la petición CSR, es decir, que se encargue de configurar una cuenta especial de correo electrónico en la que se reciban todas las peticiones CSR y otra a la que se envíen los certificados que ha firmado el servidor.

16 Volver a escribir los URL

Gracias a los URL, los visitantes acceden a su sitio Web. Como administrador de Apache, tendrá que asegurarse de que todos los URL funcionan correctamente. ¿Cómo se hace? Revisando los archivos de registro de errores para detectar los URL que no funcionen. Si ve que alguna de las peticiones formuladas a su servidor ha obtenido como respuesta el código de estado "404 Not Found", habrá llegado el momento de investigar los localizadores. A menudo, cuando los diseñadores de un documento HTML actualizan un sitio Web, olvidan que los directorios sobrantes pueden hacer que los bookmark (favoritos) de los clientes dejen de funcionar.

Como administrador, ¿qué tiene que hacer para solventar este tipo de problemas? Las buenas noticias son que hay un módulo llamado mod_rewrite que le permite crear interesantes soluciones utilizando las reglas de los URL. En este capítulo hablaremos de este módulo y veremos una serie de ejemplos prácticos que nos enseñarán cómo volver a escribir los URL.

Entre todos los módulos de Apache disponibles, mod_rewrite destaca por sí solo. Es muy sofisticado, principalmente por dos razones:

- Proporciona a los administradores de Apache cierto grado de flexibilidad a la hora de volver a escribir los URL, algo que resulta muy práctico.
- Su documentación es muy buena.

Motor de Apache para volver a escribir URL

Cuando Apache recibe una petición de un URL, suministra el archivo solicitado al cliente (el explorador Web). ¿Qué pasaría si quisiese intervenir en el proceso y dirigir el URL a otro archivo o a otro URL? Aquí es donde mod_rewrite entra en escena. Suministra un mecanismo flexible para crear las reglas que dirigen los URL. Tendrán el siguiente aspecto:

```
regex-modelo-con-el-que-coincidir          regex-sutituir-modelo
```

De todas formas, también se pueden añadir condiciones (como por ejemplo, más modelos con los que hay que coincidir) para determinar una regla en la que la sustitución únicamente tendrá lugar si se cumplen las condiciones establecidas. Apache considerará el URL sustituido como subpetición interna o bien lo enviará al explorador Web como una redirección externa. Vamos a ver un ejemplo para comprenderlo mejor. En la figura 16.1 tenemos una petición y el resultado obtenido con el módulo mod_rewrite.

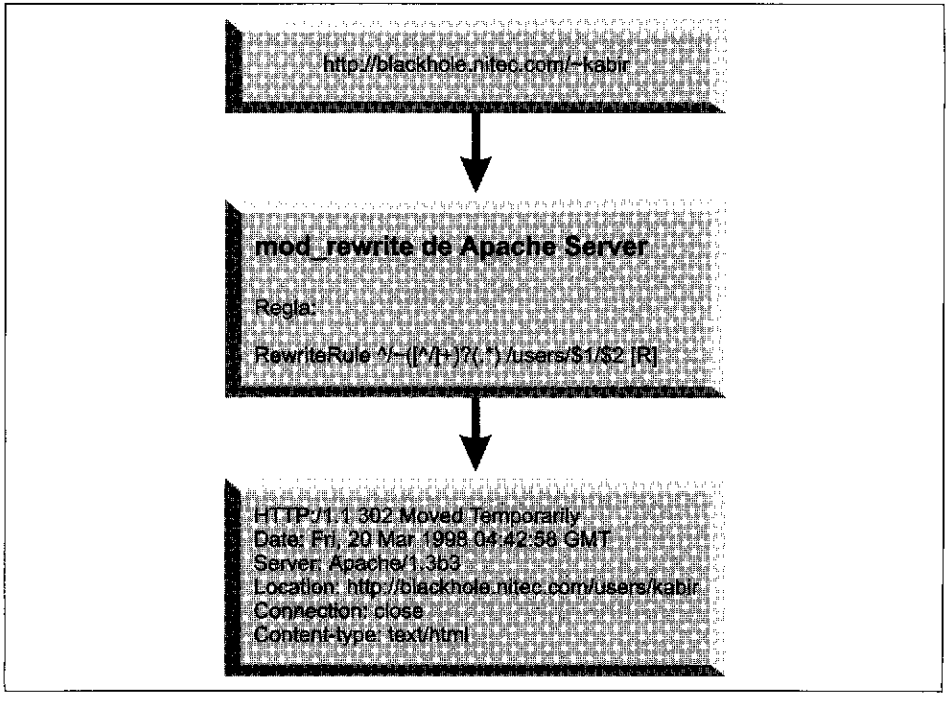


Figura 16.1. Ejemplo de una operación en la que se vuelve a escribir un URL basándose en una regla

Se realiza una petición al servidor Apache para http://blackhole.nitec.com/~kabir. El servidor la recibe y se la entrega al módulo mod_rewrite del URL de traducción, que se tiene que hacer cargo de procesar la petición. El módulo aplica la regla definida en la directriz RewriteRule. En este caso en particular, la regla determina que, si se encuentra el conjunto de símbolos /~([^/]+)?(.*), se sustituirá con /users/$1/$2. Como en la regla aparece la etiqueta [R], se enviará un URL externo al explorador Web. La salida muestra que la nueva dirección será http://blackhole.nitec.com/users/kabir.

Como se puede ver, esta solución puede llegar a ser muy útil en un momento dado.

Vamos a echar un vistazo a las directrices que le permiten aprovechar la capacidad de volver a escribir los URL. Es posible que también esté familiarizado con las variables del servidor que se pueden utilizar para escribir reglas y condiciones:

SERVER_NAME	Nombre del host del servidor Web.
SERVER_ADMIN	Dirección de correo electrónico del administrador del servidor Web.
SERVER_PORT	Dirección del puerto del servidor Web.
SERVER_PROTOCOL	Versión del protocolo HTTP que se usa con el servidor Web.
SERVER_SOFTWARE	Nombre del distribuidor del servidor Web.
SERVER_VERSION	Versión del software del servidor Web.
DOCUMENT_ROOT	Directorio superior de los documentos del sitio Web.
HTTP_ACCEPT	Tipos MIME con los que puede trabajar el cliente Web.
HTTP_COOKIE	Cookie procedente del cliente Web.
HTTP_FORWARDED	URL al que se dirige.
HTTP_HOST	Nombre del host del servidor Web.
HTTP_PROXY_CONNECTION	Información sobre la conexión HTTP proxy.
HTTP_REFERER	URL al que se refiere el localizador actual.
HTTP_USER_AGENT	Información sobre el cliente Web.

REMOTE_ADDR	Dirección IP del cliente Web.
REMOTE_HOST	Nombre del host del cliente Web.
REMOTE_USER	Nombre de usuario del cliente autenticado.
REMOTE_IDENT	Información sobre la identificación de un usuario remoto.
REQUEST_METHOD	Método HTTP que se utiliza para solicitar el URL.
SCRIPT_FILENAME	Ruta física del script solicitado.
PATH_INFO	Ruta del URL solicitado.
QUERY_STRING	Petición de datos que se envía junto con el URL solicitado.
AUTH_TYPE	Tipo de autenticación utilizado.
REQUEST_URI	URI solicitado.
REQUEST_FILENAME	Igual que SCRIPT_FILENAME.
THE_REQUEST	URL solicitado.
TIME_YEAR	Año.
TIME_MON	Mes.
TIME_DAY	Día.
TIME_HOUR	Hora.
TIME_MIN	Minuto.
TIME_SEC	Segundo.
TIME_WDAY	Día de la semana.
TIME	Hora.
API_VERSION	Versión de API utilizada.
IS_SUBREQ	Se utiliza si la petición es una subpetición.

RewriteEngine

```
Sintaxis: RewriteEngine on | off
Predeterminado: RewriteEngine off
Contexto: configuración servidor, host virtual, configuración por
directorio (.htaccess)
```

Esta directriz le permite activar y desactivar el motor de reescritura del módulo mod_rewrite. Por defecto está desactivado. Para utilizarlo, tendrá que activarlo utilizando el valor on.

Si prefiere activarlo desde los archivos de configuración de acceso de cada directorio (.htaccess), tendrá que configurar esta directriz utilizando el contexto adecuado:

```
Options FollowSymLinks
```

En otras palabras, si el directorio pertenece a un host virtual, tendrá que asegurarse de que esta opción se encuentra dentro del contenedor del host correspondiente.

Del mismo modo, si el directorio en cuestión es parte del espacio reservado a los documentos del servidor Web principal, compruebe que esta opción se encuentra dentro de la configuración del servidor.

Obsérvese que, al activar las reglas de reescritura en los archivos de configuración por directorios, puede reducir la efectividad del servidor Apache. Esto es así porque Apache utiliza un truco para poder trabajar con estas reglas, que ralentiza la carga de los procesos del servidor. Por lo tanto, mi consejo es que, siempre que sea posible, trate de evitar el uso de las reglas dentro de los archivos de configuración de cada directorio.

RewriteOptions

```
Sintaxis: RewriteOptions opción1 opción2 ...
Predeterminado: nada
Contexto: configuración servidor, host virtual, configuración por
directorio (.htaccess)
```

Esta directriz le permite especificar las opciones que desea modificar con el motor de reescritura. En la actualidad, la única opción disponible es inherit. Al usarla forzará a que se emplee un nivel de configuración mayor. Por ejemplo, si determina esta directriz en la zona de configuración del servidor principal, el host virtual definido en los archivos de configuración estará influido por las configuraciones de reescritura, como reglas, condiciones, conversiones, etc.

Algo parecido ocurre cuando la directriz se incluye en el archivo de configuración para los directorios (.htaccess), verá que compartirá las reglas, condiciones y conversiones de reescritura del directorio al que pertenece. Por defecto, el motor de reescritura no permite este tipo de configuración, pero con esta directriz se puede cambiar.

RewriteRule

```
Sintaxis: RewriteRule modelo-de-búsqueda cadena-de-sustitución
[lista de etiquetas]
Predeterminado: nada
Contexto: configuración servidor, host virtual, configuración por
directorio (.htaccess)
```

Esta directriz le permite definir una regla de reescritura. Debe tener dos argumentos. El primero es el modelo de búsqueda que se tiene que cumplir para aplicar la cadena de sustitución. El modelo de búsqueda se escribe utilizando una expresión regular. La cadena de sustitución se puede construir a partir de texto plano, referencias a otras cadenas, valores de las variables del servidor o funciones de conversión. La lista de etiquetas puede contener una etiqueta o más, separadas por comas, para informar al motor de reescritura de lo que tiene que hacer en la siguiente sustitución. Volvamos al ejemplo anterior:

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

Aquí, el modelo de búsqueda es /~([^/]+)/?(.*) y la cadena de sustitución, /users/$1/$2. Como se puede ver, en la cadena se utilizan referencias a otras cadenas. La primera, $1, se corresponde con la que se encuentra dentro del primer paréntesis (empezando por la izquierda). De este modo, $1 será ([^/]+) y $2 (.*). Cuando la petición de un URL tiene la siguiente forma:

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

el valor de $1 es kabir y $2 es welcome.html, por lo que la sustitución tendrá el siguiente aspecto:

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

Cuando especifica más de una directriz RewriteRule, la primera actuará sobre el URL original y, si coinciden los cambios, la segunda directriz no llegará a actuar sobre él, dejando que sea la primera RewriteRule quien efectúe los cambios. En caso de que los ajustes tengan lugar en cada paso, el funcionamiento de un conjunto de tres reglas será el siguiente:

```
RewriteRule modelo-de-búsqueda-del-URL-original cadena-de-
sustitución-1  [etiquetas]
RewriteRule modelo-de-búsqueda-de-la-cadena-de-sustitución-1 cadena-
de-sustitución-2  [etiquetas]
RewriteRule modelo-de-búsqueda de la cadena de sustitución-2 cadena-
de-sustitución-3  [etiquetas]
```

¿Se puede aplicar más de una regla al URL original? Sí, se puede utilizar la etiqueta C con el fin de indicarle al motor de búsqueda que encadene varias reglas.

En este caso, es posible que no quiera proceder a la sustitución hasta que se verifiquen todas las comprobaciones. Si el resultado es positivo, puede aplicar una regla de sustitución especial.

Los detalles de las posibles etiquetas son las siguientes:

C | chain	Especifica la regla que se puede encadenar con la siguiente. Cuando se utiliza la opción C, únicamente se accederá a la regla encadenada si se cumple todo lo exigido en su antecesora. Todas las reglas que formen parte de la cadena tiene que tener la etiqueta C. En el momento en que no se cumplan los requisitos de una regla, se ignorarán los del resto.
E=var:value | env=var:value	Con esta directriz puede configurar una variable de entorno, a la que se accede a través de las condiciones rewrite, script CDI, SSL, etc.
F | forbidden	Cuando aparece esta etiqueta en una regla, se le envía una respuesta HTTP al explorador Web con la cabecera FORBIDDEN (código de estado 403). De esta forma se desactiva la petición del URL.
G | gone	Cuando una regla utiliza esta etiqueta, se le envía una respuesta HTTP al explorador Web con la cabecera GONE (código de estado 410). Así se le informa de que el servidor ya no puede atender ese URL.
L | last	Le indica al motor de reescritura que finalice de inmediato la regla que está procesando para que no se aplique ninguna otra al URL.
N | next	Le indica a rewrite que vuelva a empezar desde la primera regla. De todas formas, la primera regla no volverá a trabajar con el URL original, sino con el resultante de la última sustitución. Así se crea un bucle. Se han de establecer unas condiciones de finalización, ya que si no el bucle sería infinito.
NS | nosubreq	Con esta etiqueta se evita la aplicación de reglas a las peticiones URL generadas internamente.

P | proxy	Con esta etiqueta se convierte la petición de un URL en una petición interna dirigida al proxy. Este módulo sólo funciona si se ha compilado Apache con el módulo mod_proxy y se ha configurado para que trabaje con ellos.
PT | passthrough	Es un problema que va a desaparecer en la próxima versión de Apache. La etiqueta obliga a rewrite a modificar la estructura interna del registro de petición para que el URL de un miembro de la estructura pase a ser el nombre de un archivo. Se suele utilizar este módulo con otros que disponen de traductores de URL a nombre de archivo. Un ejemplo podría ser el módulo mod_alias.
QSA | qsappend	Le permite adjuntar datos (como los pares de valores clave=valor) a la parte de la cadena de petición de la URL sustituida.
S=n | skip=n	Salta n reglas.
T=mime-type | type=mime-type	Obliga al MIME-type especificado a que sea el del archivo solicitado.

Obsérvese que se pueden usar todas las condiciones anteriores empleando una o más directrices RewriteCond, que se verán en la siguiente sección.

RewriteCond

```
Sintaxis: RewriteCond cadena-prueba modelo_condición [lista
etiquetas]
Predeterminado: nada
Contexto: configuración servidor, host virtual, configuración por
directorio (.htaccess)
```

Esta directriz se utiliza cuando se quiere añadir una condición extra para una regla de reescritura especificada por la directriz RewriteRule. Puede tener varias directrices RewriteCond por cada RewriteRule. Todas las condiciones rewrite tienen que definirse antes que la propia regla.

La cadena de prueba tiene que construirse en texto plano, variables de servidor o referencias tanto a actual regla de reescritura como a la última condición. Para acceder a la primera referencia de la directriz RewriteRule actual, se utiliza $1 y para acceder a la de la directriz RewriteCond, %1.

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

Hay unos cuantos accesos a datos especiales:

%{ENV:variable}	Utilícelo para acceder a la variable de entorno que esté a disposición del proceso de Apache.
%{HTTP:header}	Utilícelo para acceder a la cabecera HTTP que se emplea en la petición.
%{LA-U:variable}	Utilícelo para acceder al valor de la variable que no está disponible en el estado actual de proceso. Por ejemplo, si tiene que utilizar la variable REMOTE_USER en una condición de reescritura incluida en el archivo de configuración del directorio (.htaccess), no puede utilizar %{REMOTE_USER} porque esta variable únicamente se define cuando el servidor ha completado la fase de autenticación que sigue al método mod_rewrite del proceso de un URL. Para ver cuál es el nombre del usuario que se ha autenticado, puede utilizar %{LA-U:REMOTE_USER}. De todas formas, si va a acceder a los datos de REMOTE_USER desde la directriz RewriteCond que se encuentra en el archivo de configuración para cada directorio, podrá utilizar %{REMOTE_USER} porque la fase de autorización y la variable del servidor ya están a disposición de quien la solicite. La mejora se puede generar a partir una petición basada en una subpetición interna de un URL.
%{LA-F:variable}	En la mayoría de los casos es igual que %{LA-U:REMOTE_USER}, pero la mejora se efectúa a través de una petición interna que se basa en el nombre del archivo.

El modelo de las condiciones se puede utilizar con ciertas anotaciones y tiene que comenzar por una expresión regular. Por ejemplo, puede efectuar ciertas comparaciones léxicas entre la cadena de prueba y el modelo, pero tendrá que utilizar un prefijo que puede ser cualquiera de los caracteres <, > o =. Con este último, el modelo se compara con la cadena considerándolo texto plano.

Hay ocasiones en las que se quiere comprobar si la cadena de prueba es un archivo, un directorio o un enlace simbólico. En este caso se puede sustituir el modelo condicionante por las siguientes cadenas especiales:

-d Verifica la existencia de la cadena de comprobación especificada en el directorio.

-f Verifica la existencia de la cadena de comprobación especificada en el archivo.

-s Verifica la existencia de la cadena de comprobación especificada en el archivo distinto de cero.

-l Verifica la existencia de la cadena de comprobación especificada en el vínculo simbólico.

-F Verifica la existencia y accesibilidad de la cadena de comprobación especificada en el archivo.

-U Verifica la validez y el acceso de la cadena de comprobación especificada en el URL.

La lista de etiquetas opcionales puede consistir en una o más cadenas separadas por comas:

NC | nocase En la prueba no tienen que coincidir mayúsculas y minúsculas.

OR | ornext Normalmente, cuando tiene que visualizar más de una RewriteCond para la directriz RewriteRule, estas condiciones se unen por medio de AND para que se efectúe la última sustitución. De todas formas, si tiene que crear una relación OR entre dos condiciones, utilizará esta etiqueta.

RewriteMap

```
Sintaxis: RewriteMap nombre-de-la-conversión tipo-de-
conversión:fuente
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Esta directriz facilita un valor clave que se utilizará durante la conversión. Piense en la conversión como en una tabla de datos donde cada fila tiene una clave y un valor. Generalmente la conversión se guarda en un archivo. Pero

también se puede tratar de un archivo de texto, un DBM, funciones internas de Apache o un programa externo. El tipo de conversión se corresponde con la fuente de Apache. Los tipos que se pueden aplicar son los siguientes:

txt Archivo de texto plano que tiene una líneas con pares clave/valor que se encuentran en una misma línea y que están separados, por lo menos, por un carácter en blanco. Los archivos contienen líneas de comentarios que comienzan por caracteres #. También pueden tener líneas en blanco. Estas líneas y las de comentarios se ignorarán. Por ejemplo:

```
Key1    valor1
Key2    valor2
```

define dos tipos de pares de valores. Obsérvese que el archivo de texto que se basa en las conversiones se lee durante el inicio de Apache y cuando se ha actualizado con el servidor en marcha.

rnd Un archivo de texto plano especial que tiene todas las restricciones del tipo txt, pero que permite el grado de flexibilidad definido en el valor. El valor de cada clave se puede definir como un conjunto de ORed utilizando la barra vertical (|). Por ejemplo:

```
Key1 primer_valor_para_clave1 | segundo_valor_para_clave1
Key2 primer_valor_para_clave2 | segundo_valor_para_clave2
```

define los pares principales para las claves que tengan varios valores. El valor seleccionado se escoge al azar.

int Las funciones internas de Apache toupper(key) o tolower(key) se pueden utilizar como fuentes de conversión. La primera transforma todos los caracteres de la clave en mayúsculas, mientras que la segunda en minúsculas.

dbm Se puede utilizar un archivo DBM como fuente de conversión. Puede ser muy útil y rápido (comparado con los archivos de texto), sobre todo cuando hay que trabajar con grandes números de pares. Los archivos DBM que se emplean durante las conversiones se leen durante el arranque de Apache o, si el servidor ya está funcionando, únicamente si se actualiza su contenido.

prg Programa externo que se puede utilizar para generar el valor. Cuando se utiliza un programa, se inicia junto con Apache y se procede a la transferencia de datos (entre estos dos nodos) a través de STDIN y STDOUT. Recuerde que tiene que usar la directriz RewriteLock

para definir el archivo de bloqueo que se utiliza con los programas externos. A la hora de construir uno de estos programas, asegúrese de que lee la entrada de stdin y de que escribe la salida en stdout en modo I/O sin utilizar el buffer.

RewriteBase

```
Sintaxis: RewriteBase <URL base>
Predeterminado: ruta del directorio en el que se encuentra el archivo
de configuración de acceso (.htaccess)
Contexto: configuración de acceso por directorio (.htaccess)
```

Esta directriz sólo se utilizará con las reglas de reescritura que se definen en los archivos de configuración de cada directorio. Unicamente será obligatorio su uso con las rutas de los URL que no apunten directamente al archivo solicitado. Utilice con esta directriz el alias que tenga asignado al directorio. De esta forma se asegurará de que se emplee el módulo mod_rewrite en vez de la ruta física del URL final (ya sustituido). Por ejemplo, cuando se configura un alias de la siguiente manera:

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

y las reglas de reescritura se activan en el archivo /www/nitec/htdocs/icons/.htaccess, la directriz RewriteBase debería ser de la siguiente manera:

```
RewriteBase /icons/
```

RewriteLog

```
Sintaxis: RewriteLog ruta/al/archivo/de/registro
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Si quiere registrar las aplicaciones de sus reglas de reescritura, utilizará esta directriz para determinar el nombre del archivo de registro. Al igual que otras directrices, supone que cuando una ruta no comienza con una barra inclinada (/), está referida al directorio raíz del servidor principal. Por ejemplo:

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

escribirá un archivo de registro de errores que se guardará en el subdirectorio de registro que se encuentra dentro del árbol de directorios del servidor principal. Como ya hemos mencionado, el único que puede escribir en un archivo de registro es el servidor.

RewriteLogLevel

```
Sintaxis: RewriteLogLevel  nivel
Predeterminado: RewriteLogLevel 0
Contexto: configuración servidor, host virtual
```

Esta directriz le permite especificar qué se registra en el archivo. El valor predeterminado, 0, indica que no se ha de registrar nada. De hecho, con este número se suspenden los procesos de registro del módulo. Así que, si quiere desactivar el registro, deje el valor predeterminado de esta directriz.

Si configura la directriz RewriteLog a /dev/null y RewriteLogLevel a un valor distinto de cero, se estará ejecutando el proceso interno de registro, pero no se generará ningún archivo. Tiene diez niveles, comprendidos entre el 0 y el 9. Cuanto más alto sea el número, mayor cantidad de datos se incluirá en el registro.

RewriteLock

```
Sintaxis: RewriteLock nombre de archivo
Predeterminado: nada
Contexto: configuración servidor, host virtual
```

Si utiliza un programa externo para crear la conversión de rewrite, empleará esta directriz para especificar el nombre del archivo. Este fichero se usa como archivo de bloqueo para sincronizar la comunicación con los programas externos de conversión.

URL

En esta sección veremos algunos ejemplos relacionados con la reescritura de URL que coincide con la composición de los URL. A menudo hay que redirigir o ampliar una petición URL a otra. En los ejemplos que le mostramos a continuación veremos cómo puede utilizar mod_rewrite para este fin.

Redirigir el directorio principal de un usuario a un nuevo servidor Web

Si tiene demasiadas páginas de sus usuarios en el servidor Web y quiere moverlas a una nueva máquina, necesitará una regla para redirigir, tal y como se puede ver en la figura 16.2.

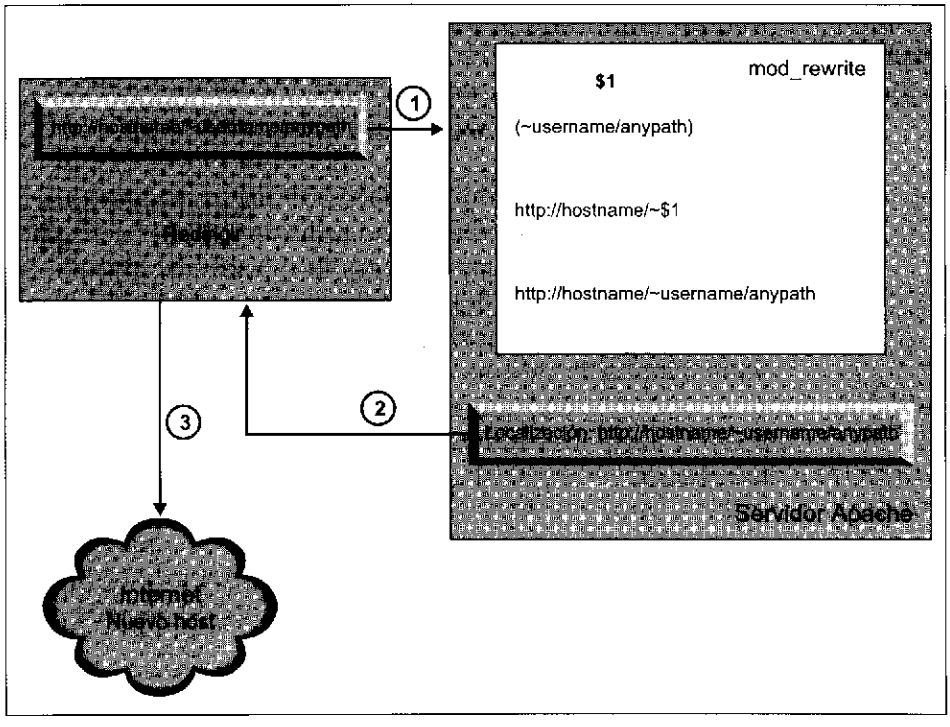


Figura 16.2. Redirigir el directorio principal de un usuario a un nuevo servidor Web

La solución con mod_rewrite es muy simple. Tendrá que redirigir todos los URL de /~username/anypath a http://hostname/~username/anypath como sigue:

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

Observe que se emplea la etiqueta L para especificar que no se utilizará ninguna otra regla al URL final.

Búsqueda de una página en varios directorios

A veces es necesario dejar que el servidor Web busque las páginas en más de un directorio. Y en este caso, técnicas como MultiViews no sirven para nada. Por ejemplo, supongamos que quiere atender una petición dirigida a http://hostname/filename.html, de modo que si el archivo no se encuentra en el primer directorio (dir1), el servidor lo busque en otro (dir2). En la figura 16.3 se puede ver qué es lo que ocurre.

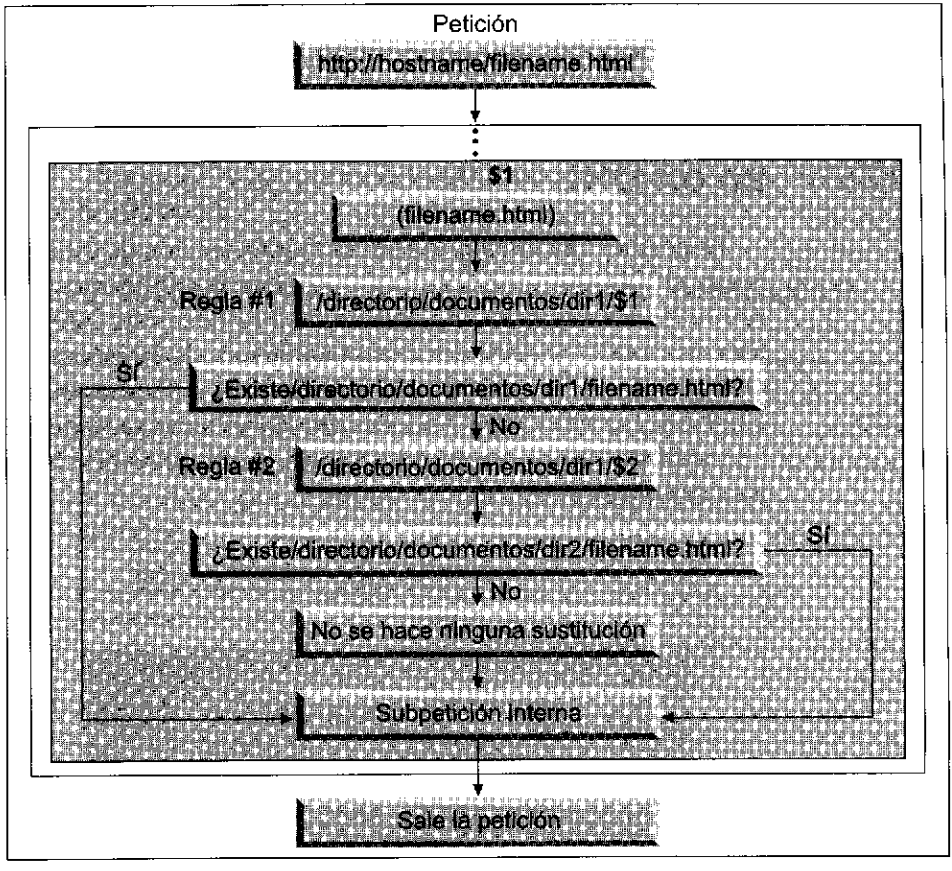


Figura 16.3. Búsqueda de una página en varios directorios

Las reglas que hay que implementar son las siguientes:

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

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

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

La primera regla sustituye al URL por /directorio/documentos/dir1/$1 (donde $1 es el archivo solicitado), si el archivo solicitado se encuentra en el subdirectorio /directorio/documentos/dir1/. Si se cumple la condición, ésta será la última regla que se aplique. De todas formas, si no se encuentra ninguna marca, se aplicará la siguiente regla. Hace lo mismo que la primera, salvo que busca en el directorio dir2. Si tampoco se cumple lo establecido con esta

regla, no se efectuará ninguna sustitución y se le entregará a Apache, siguiendo el proceso habitual.

Configurar una variable de entorno basada en un URL

Es posible que intente conservar la información de estado entre peticiones y usar el URL para codificarla. Pero no querrá emplear un script CGI wrapper con todas las páginas simplemente para saltarse esta información. Se puede usar una regla de reescritura para saltarse toda la información de estado y almacenarla en las variables de entorno para utilizarlas más tarde con CGI o con XSS. De esta forma, un URL /foo/S=java/bar se traducirá como /foo/bar y la variable de entorno STATUS toma el valor java. En la figura 16.4 podemos ver qué está ocurriendo.

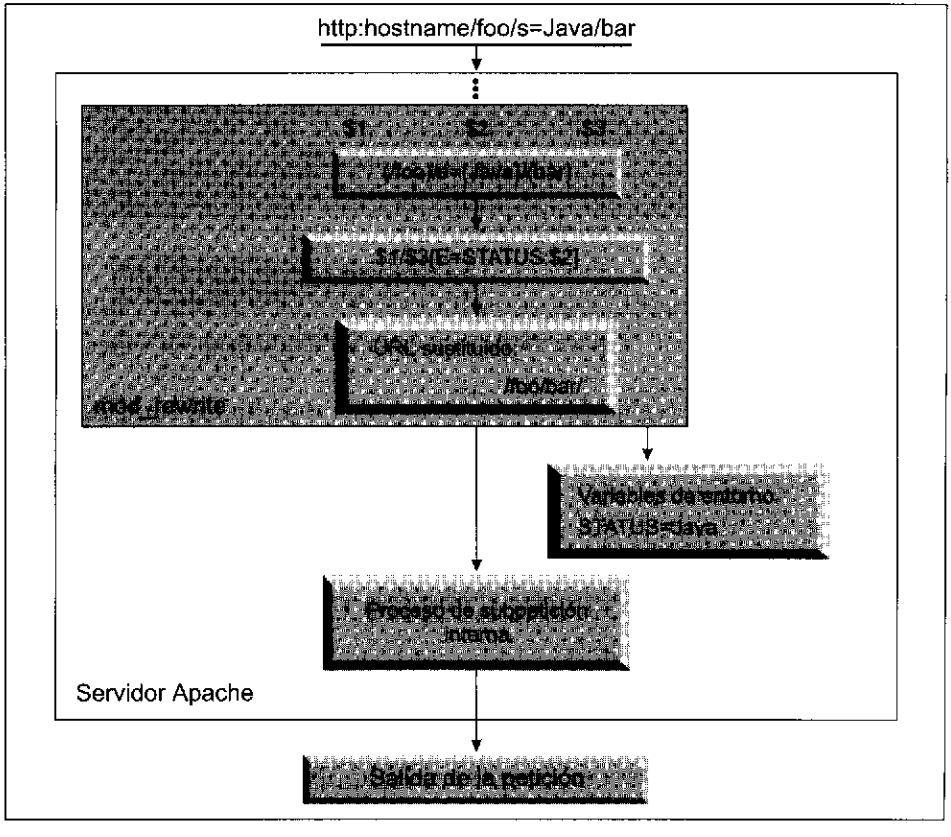


Figura 16.4. Configurar una variable de entorno desde un URL

Los pasos que se dan en la figura 16.4 para reescribir el URL se pueden implementar usando la siguiente regla:

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

El valor de $3 se guarda en la variable de entorno llamada STATUS utilizando la etiqueta E.

Crear sitios www.nombreusuario.host.com

Supongamos que queremos acceder a la página principal de www.nombreusuario.host.com a través de los registros de una dirección DNS (A) de dicha máquina y sin tener que pasar por ningún host virtual. Para las peticiones HTTP/1.0 no hay ninguna solución, pero para las HTTP/1.1 que contengan una cabecera Host:, puede utilizar la siguiente regla para convertir www.nombreusuario.host.com/cualquierruta en /principal/nombreusuario/cualquierruta:

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

En la figura 16.5 se puede cómo funciona.

Este es un ejemplo de una serie de reglas encadenadas. La primera actúa como una condición que comprueba si la variable de entorno HTTP_HOST coincide con el modelo www.nombreusuario.host.com. En caso afirmativo, se aplica la regla. En otras palabras, una petición como www.nombreusuario.host.com/cualquierruta se sustituirá por www.nombreusuario.host.com/cualquierruta. La verdad es que parece algo confuso porque la sustitución no está nada clara. Pero es necesaria para que se pueda obtener el nombre de usuario por medio de la segunda regla. Lo obtiene del URL resultante de la primera sustitución, lo procesa y crea uno nuevo (/principal/nombreusuario/cualquierruta), que será el que se utilice con las subpeticiones internas.

Redireccionar un URL fallido a otro servidor Web

Si trabaja con una red con varios servidores Web y suele mover los contenidos de un servidor a otro, es posible que se encuentre con un problema a la hora de redireccionar las peticiones URL fallidas dirigidas de un servidor A a otro B. Hay muchas formas de hacerlo: a través de la directriz ErrorDocument, a través de un script CGI que se haga cargo del cambio o utilizando el módulo

mod_rewrite para volver a escribir los URL fallidos y dirigirlos a otro servidor. Tenga en cuenta que la peor solución de todas es usar el módulo. La primera solución ofrece la mayor eficacia, pero es la menos flexible y la menos segura ante errores:

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

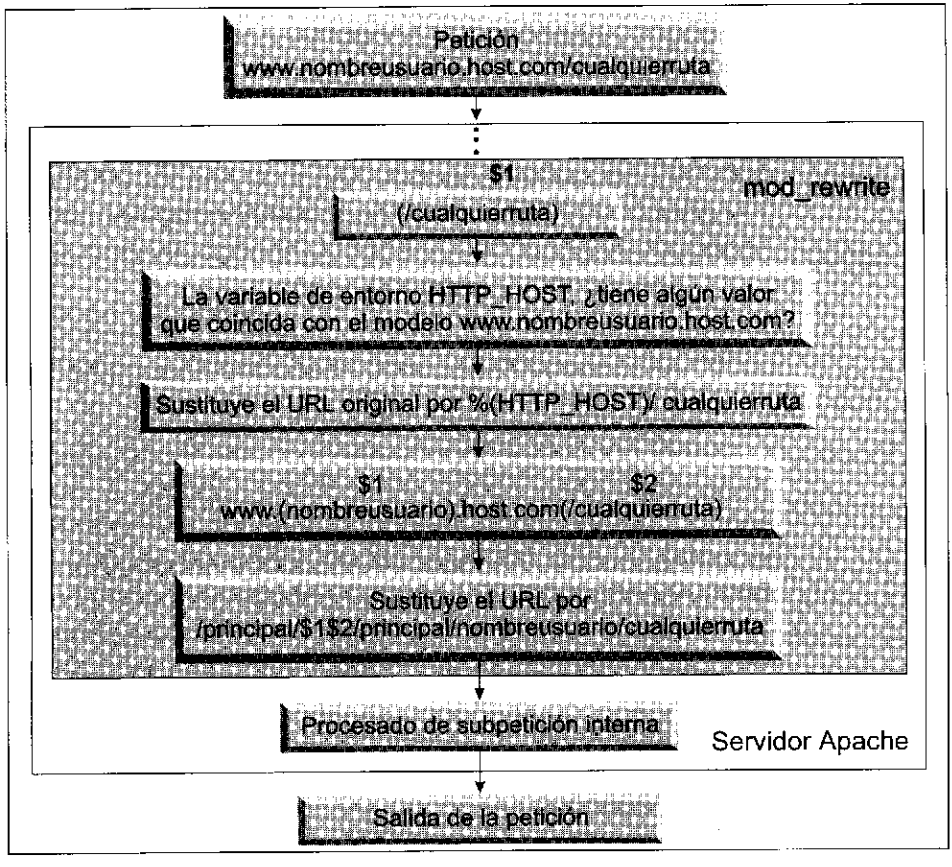


Figura 16.5. Nombre de usuario para cada host virtual

El problema es que sólo trabaja con las páginas que se encuentran dentro de DocumentRoot. Se pueden añadir más condiciones (con las que controlar los directorios principales, por ejemplo), pero hay una variante mejor:

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

Aquí se utiliza una de las propiedades de mod_rewrite que funciona con todo tipo de URL. Pero tiene una clara desventaja y es que actúa negativamente sobre la velocidad del servidor, porque para cada petición que se recibe se genera más de una subpetición interna. Si trabaja en una máquina con una CPU potente, utílice esta solución. En caso contrario, use la primera o, mejor aún, un script CGI ErrorDocument.

Crear un acceso múltiple

Este ejemplo le mostrará cómo crear una regla que dirija las peticiones que se basen en un tipo de dominio, como .com, .net, .edu, .org, .uk, .de, etc. La idea es redirigir al visitante al sitio Web geográficamente más cercano. Esta técnica la utilizan las grandes empresas para redirigir a sus clientes internacionales a los sitios Web o a los servidores FTP apropiados.

El primer paso para crear este tipo de soluciones es construir un archivo de conversión. Por ejemplo, el siguiente muestra uno basado en texto, llamado site.redirect.map:

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

Cuando se recibe una petición para http://www.mydomain.com/download/ cualquierruta procedente de un servidor llamado dialup001.demon.uk, habrá que redirigir la petición al sitio Web http://www.mydomain.uk/download/ y, de igual forma, los servidores que pertenezcan a otros dominios como .com, .net, .edu y .org se redirigen a http://www.mydomain.com/download/.

Las reglas necesarias serán las siguientes:

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

Como se puede observar en la figura 16.6, cuando un host como dialup001.demon.uk solicita la página http://www.mydomain.uk/download/index.html, la primera regla reescribe la petición utilizando el nombre del servidor del host que efectúa la petición:

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

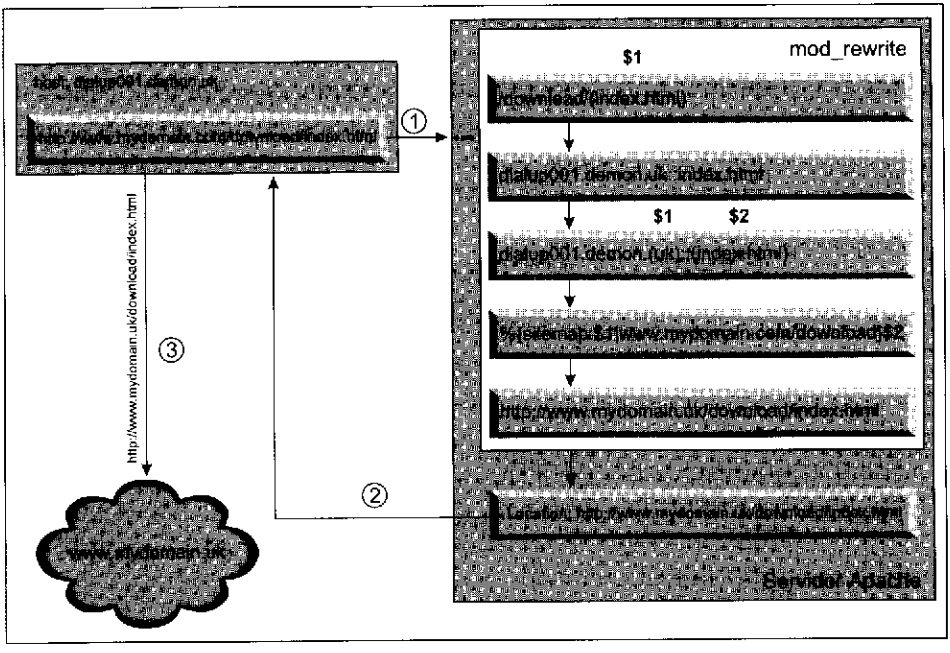


Figura 16.6. URL basado en un acceso múltiple

A continuación se aplica la siguiente regla de la cadena. Esta regla se empleará cuando el original coincide con el modelo de comparación y se crea el URL de sustitución, para lo cual se observa el archivo de conversión. Si no coincide la comparación, se usará entonces el URL predeterminado www.mydomain.com. Para ello se utiliza el operador | (o) en la cadena del URL sustituido. Quizá facilitemos la comprensión de esta segunda regla utilizando el algoritmo que se encuentra en el listado 16.1.

```
# sustituye el actual URL utilizando la información del tipo de
# dominio almacenada en $1 y busca en el archivo de conversión.

If  (el archivo de conversión que tiene la clave coincide con el
tipo de dominio) then
# utiliza el valor de la clave de la siguiente forma:

      Substituted URL = valor-de-key$2    # donde $2 es cualquier
                                          # cosa que se encuentre
                                          # detrás del nombre
                                          # completo del host::
                                          # pattern


Else
 # Utiliza el valor predeterminado www.mydomain.com/download/$2
```

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

Endif
```

Listado 16.1. Algoritmo para la segunda regla de reescritura

La etiqueta R indica que se redirige un URL externo y L que es la última regla que se va a aplicar.

Crear URL sensibles al paso del tiempo

¿Se ha preguntado alguna vez si un mismo URL puede apuntar a distintos archivos, dependiendo del paso del tiempo? El módulo mod_rewrite lo hace posible. Hay una serie de variables llamadas TIME-xxx que se utilizan para especificar las condiciones de la sustitución. Use los siguientes modelos de comparación, <STRING, >STRING y =STRING para marcar las redirecciones dependientes del tiempo. Por ejemplo:

```
RewriteCond    %{TIME_HOUR}%{TIME_MIN} >0700
RewriteCond    %{TIME_HOUR}%{TIME_MIN} <1900
RewriteRule    ^foo\.html$             foo.day.html
RewriteRule    ^foo\.html$             foo.night.html
```

De esta forma, cuando se accede a foo.html entre las 07:00 y las 19:00 se muestra el archivo foo.day.html. En el resto de la franja horaria, se muestra foo.night.html.

Control de contenidos

Los ejemplos que se verán en esta sección le mostrarán las relacionadas con el contenido. Aprenderá a crear URL compatibles con versiones anteriores, basadas en el tipo de explorador o bien en redirecciones transparentes de HTML o CGI.

Agregar compatibilidad con versiones anteriores en los URL

Vamos a suponer que recientemente ha renombrado la página bar.html a foo.html y que desea trabajar con el URL antiguo por cuestiones de compatibilidad. Además, quiere que los usuarios que utilizan la dirección anterior no

se den cuenta de que la página ha cambiado de nombre. ¿Cómo se puede hacer? De la siguiente manera:

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

Si quiere informar al explorador de los cambios, puede efectuar una reescritura externa para que el explorador muestre el nuevo URL. Todo lo que tiene que hacer es añadir el etiqueta R:

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

Crear URL cuyo contenido coincida con el explorador

Se puede utilizar una serie de reglas para mostrar diferentes contenidos (por medio de subpeticiones internas) a los distintos exploradores. No se puede emplear la negociación del contenido porque los exploradores no usan el mismo formato que los URL para mostrar esta información. En vez de eso tendrá que emplear una cabecera HTTP User-Agent. Por ejemplo, si la cabecera User-Agent de un explorador Web coincide con Mozilla/3, podrá enviar una página que aproveche las características de Netscape Navigator 3. Si el explorador es una versión anterior, podrá utilizar otra página.

Si la cabecera HTTP User-Agent comienza por Mozilla/3, entonces la página foo.html se volverá a escribir a foo.NS.html y se detiene el proceso de reescritura. Si el explorador es Lynx o Mozilla/1 ó 2, el URL pasará a ser foo.20.html. El resto de los exploradores recibirán la página foo.32.html. Para ello se utilizan las siguientes reglas:

```
RewriteCond %{HTTP_USER_AGENT}  ^Mozilla/3.*
RewriteRule ^foo\.html$         foo.3x.html          [L]

RewriteCond %{HTTP_USER_AGENT}  ^Lynx/.*          [OR]
RewriteCond %{HTTP_USER_AGENT}  ^Mozilla/[12].*
RewriteRule ^foo\.html$         foo.2x.html          [L]
```

Cuando se recibe una petición para un URL como http://hostname/foo.html, se verifica la primera condición para ver si la variable de entorno HTTP_USER_AGENT tiene un valor que contiene la cadena Mozilla/3. Si se cumple, se aplica la primera regla. Se sustituye foo.3x.html con el URL original y se completa el proceso de reescritura. De todas formas, cuando no se aplica la primera, se pasa a la segunda. Hay dos condiciones que son ORed. Es decir, se tiene que cumplir una de estas condiciones antes de aplicar la regla.

La primera condición es verificar la misma variable de entorno para la subcadena Lynx/ y la segunda comprueba la misma variable de entorno para la subcadena Mozilla/1 o Mozilla/2. Si se cumple cualquiera de estas condiciones, se aplica la regla. Se sustituye foo.2x.html por el URL original. El URL resultante se convierte en una subpetición y se procesa como una petición usual de Apache.

Crear un HTML para una puerta de enlace CGI

Dispone de varias formas para transformar las páginas estáticas foo.html en una variante dinámica foo.cgi sin informar al explorador del usuario:

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

La regla reescribe una petición para foo.html en una dirigida a foo.cgi. Fuerza la corrección de los tipos MIME para ejecutar un script CGI. Una solicitud, como http://hostname/foo.html, se traduce internamente a una petición para el script CGI. El explorador no sabe cómo se ha redirigido la petición.

Restricciones de acceso

Con estos ejemplos aprenderemos todo lo relacionado con el control de acceso. Aprenderá como administrar el acceso a ciertas áreas de su sitio Web utilizando los módulos de reescritura de URL.

Robots de bloqueo

Es muy sencillo bloquear el acceso de los robots molestos que se dedican a recuperar páginas para un sitio Web. Puede trabajar con un archivo llamado /robots.txt que contiene las entradas de REP (Robot Exclusion Protocol) pero no es suficiente para controlar a ciertos robots. Una solución podría ser la siguiente:

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

La regla tiene dos condiciones:

```
if  (HTTP_USER_AGENT of the robot matches a pattern "NameOfBadRobot")
and
```

```
    (REMOTE_ADDR of the requesting host is 123.45.67.8 to
123.45.67.9) then
      No substitution but send a HTTP "Forbidden" header (status
code 403)
endif
```

Como se puede observar, coincide la cabecera User-Agent, junto con las direcciones IP que utiliza el host. Las condiciones anteriores le permiten trabajar con varias direcciones IP (123.45.67.8 y 123.45.67.9).

Crear un desvío HTTP basado en un URL

Puede programar un desvío flexible para los URL que actúe sobre la cabecera HTTP Referer y que la configure para que trabaje con tantas páginas como desee. Se hace de la siguiente forma:

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

Se utiliza junto con el archivo de conversión correspondiente:

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

Si el URL coincide con el valor de DEFLECTED que se encuentra en el archivo de conversión, automáticamente se redirigirán todas las peticiones de vuelta a la página de referencia. En cualquier otro caso, las peticiones de enviarán a los URL especificados.

Parte V
Uso de Apache hoy y mañana

17 Trucos de eficacia

Para obtener la máxima eficacia de su servidor Apache tendrá que visualizar una serie de puntos: el ordenador, la red, el contenido y la gente implicada. En este capítulo veremos distintas cuestiones que pueden reducir la efectividad y le mostraremos cómo solucionarlos.

Tengo que advertirle que, si suele modificar los núcleos de los sistemas operativos para obtener una mayor eficacia, es muy probable entonces que este capítulo no le enseñe nada nuevo. La finalidad de estas páginas es mostrar a todos aquéllos que están empezando cómo mejorar la efectividad de sus sistemas.

También es posible que tenga la sensación de que estoy dando mis primeros pasos con los servidores Apache y las plataformas para PC, pero la verdad es que lo que se va a comentar en estas páginas es válido para todas las plataformas.

El ordenador Apache

Apache se ejecuta en varios tipos de ordenadores. Aunque la arquitectura puede variar de uno a otro, los componentes de hardware que pueden llegar a reducir la efectividad del sistema, son los mismos.

Nota: supongo que el ordenador en el que está ejecutando Apache es moderno. Si utiliza un PC clónico como base sobre la cual ha montado su servidor Web, supongo que se trata de un procesador Pentium.

Cuando analice los requisitos relacionados con la efectividad, la primera pregunta que tiene que hacerse es: "¿Necesito una CPU rápida para que funcione con mi ordenador Apache?" La respuesta depende de cómo vaya a utilizar su servidor Apache.

Si la mayoría de los servidores tienen páginas HTML estáticas, la verdad es que no va a necesitar ninguna CPU muy potente para que su servidor trabaje con el nivel de eficacia requerido. Por otro lado, si genera mucho contenido dinámico usando SSI, script CGI, etc., la verdad es que el servidor Apache agradecerá una CPU rápida. En cualquier caso, siempre es preferible una CPU rápida a otra lenta.

¿Cómo de rápida? Pues la verdad es que cualquier velocidad no es lo suficientemente rápida. Cuanto mayor sea la velocidad, mejor. De todas formas, conviene que sepa que por muy rápido que sea el procesador, no conseguirá mejorar la efectividad del sistema trabajando con un gran volumen de contenidos dinámicos. Lo mejor para reducir el efecto del cuello de botella es ampliar la cantidad de memoria RAM.

Nunca se puede tener la suficiente cantidad de memoria RAM. Además, no es que sea precisamente barata. Entonces, ¿cuánta hay que tener para mejorar la eficacia? El único que puede contestar a esta pregunta es el encargado de visualizar el sistema durante la carga. Por ejemplo, en la mayoría de los sistemas UNIX puede ejecutar utilidades que le ayuden a medir la eficacia del sistema. La figura 17.1 le muestra la salida que ha generado una de estas utilidades, llamada top.

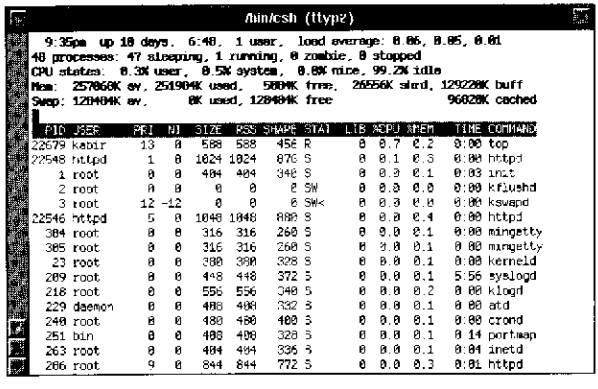


Figura 17.1. Salida del programa top

Un programa como top ofrece una gran cantidad de información sobre el sistema en el que se ejecuta. En este ejemplo en particular, el ordenador está utilizando casi toda la memoria física (256MB), pero aún no ha empezado a utilizar la memoria virtual.

Nota: si ejecuta Apache en una plataforma Windows, utilice las propiedades del sistema para ver la cantidad de memoria real, virtual y CPU que se está utilizando.

Otro programa que se puede emplear con la mayoría de los sistemas UNIX para visualizar el uso de la memoria es vmstat. En la figura 17.2 tiene la salida que genera vmstat al utilizarlo con el ordenador anterior. Cuando vea el uso que hace su ordenador de la memoria virtual (es decir, del espacio swap), se hará una idea de lo escaso que está de recursos. Ha llegado el momento de invertir en RAM.

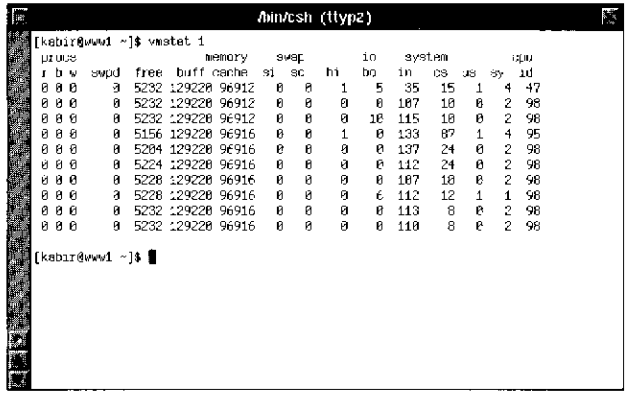


Figura 17.2. Salida generada por vmstat

Si no puede comprar más memoria o si cree que tiene toda la que necesita, puede optar por reducir la cantidad de RAM usada. Para ello puede emplear algunos de los trucos que aparecen en la sección dedicada al software.

El siguiente factor relacionado con el hardware que tiene que tener en cuenta a la hora de valorar los elementos que pueden influir en la eficacia de su ordenador es el disco duro. Un servidor Web pierde gran parte de su tiempo accediendo al disco duro y si éste es lento, se estará reduciendo la velocidad del sistema.

Asegúrese de que su servidor Web use controladores SCSI con discos SCSI. La mejor elección serían los discos ultra-wide SCSI. Siempre es conveniente

tener más de un disco duro instalado en el sistema (por ejemplo, no debería tener guardados los datos relacionados con el sistema operativo del servidor en la misma unidad de disco que los datos Web). Utilice por lo menos dos discos: uno para el sistema operativo y otro para guardar cualquier otro dato. Esta solución suele ser una buena medida de seguridad.

Los discos son otro de los factores que hay que tener en cuenta. Si tiene cierto temor a que se produzca un fallo en la unidad de disco, conviene que haga copias de seguridad con regularidad. Puede aprovecharse de las propiedades de los sistemas RAID (Redundant Array of Inexpensive Disks). Esencialmente, es un grupo de pequeñas unidades de disco que simulan la actuación de una mayor. Si falla una de las unidades, otra se hace cargo del proceso hasta que se restaure la original. El uso de subsistemas RAID puede suponer un gran rendimiento de los discos I/O y cierta seguridad con los datos de su servidor Web.

Mucha gente compra servidores PC a distribuidores que los venden como paquetes e incluyen discos IDE o EIDE bastante grandes. La verdad es que si va a trabajar con esta clase de servidores Web, le recomiendo que compre este tipo de discos, sobre todo si va a utilizar su ordenador para algo más que para jugar.

La última parte del hardware de la máquina que puede influir en la efectividad del servidor es la tarjeta de red. Supongo que el servidor va a estar conectado a una red Ethernet. La tarjeta que utilice tendrá que ser bastante rápida y ofrecer un servicio de calidad. Por ejemplo, si la red Ethernet a la que se ha conectado puede trabajar con nodos de 10Mps o 100Mps, conviene que se compre una tarjeta de 100Mps.

El resto del hardware que pueda llegar a influir en la eficacia del servidor se discutirá en la sección dedicada a redes.

Después de asegurarse que su servidor Apache cumple los requisitos aquí impuestos, puede pasar al software.

El software

El tipo de sistema operativo que utiliza Apache puede afectar considerablemente al rendimiento del servidor. Por ejemplo, si prueba la primera versión de Apache para Windows verá que es demasiado lento y lo único que podrá hacer será esperar a que el servidor madure para esta plataforma.

Hoy en día, la mayoría de servidores Apache están instalados en plataformas UNIX. De todas formas, una creencia muy extendida entre los fans de Apache es que la mayoría también funciona en clones de UNIX, como FreeBSD

o Linux. Si utiliza uno de estos sistemas, está de suerte. Pero si usa otra plataforma clónica de UNIX debería empezar por asegurarse de que cuenta con las mejoras necesarias para cumplir los requisitos impuestos por las versiones actuales de TCP/IP. Compruebe la documentación que se encuentra en www.apache.com/ para acceder a información más específica sobre su sistema operativo.

Siempre es conveniente hacerse con el último parche o actualización del sistema operativo con el que se está trabajando. Muchos de estos parches eliminan los agujeros relacionados con la seguridad y el funcionamiento en red. Por eso conviene estar atento para enterarse de cuándo se publican nuevas versiones de estos parches.

Cuando esté completamente seguro de que el hardware que tiene instalado su ordenador funciona correctamente y de que cuenta con los últimos parches o actualizaciones de su sistema operativo, podrá ponerse a buscar otro tipo de software que le ayude a superar el problema de los cuellos de botella.

Como ya hemos comentado con anterioridad, hay dos formas de hacerse con una copia de Apache. Puede bajarse la distribución binaria e instalarla en su sistema o puede compilar su propio servidor Apache e instalarlo. El último método es el más recomendable porque le permite ajustar el servidor desde el inicio. Mire el archivo binario que tiene en su sistema y compruebe que dispone de todo lo que necesita. Ejecute el comando:

```
nttpd -l
```

Así verá todos los módulos que tiene instalados el ejecutable de Apache. Si ve que hay algo que no va a necesitar, elimínelo editando el archivo de configuración, volviendo a compilar el ejecutable y repitiendo la instalación. Si su sistema operativo le permite eliminar toda la información innecesaria que se encuentra en los ejecutables (generalmente símbolos que no se utilizan para nada), suprímalos. Por ejemplo, el siguiente comando:

```
strip httpd
```

elimina todos los símbolos que se encuentran dentro del ejecutable httpd de los sistemas Linux. De esta forma se reduce el tamaño final de los archivos ejecutables, con lo que disminuye la cantidad de memoria RAM que se utilizará en la ejecución de cada httpd.

Nota: como es muy posible que la mayoría de los lectores use sistemas Linux, puede estar completamente seguro de que, independientemente de la cantidad de memoria RAM instalada en su sistema, programas

como top le informarán de toda la memoria disponible en el sistema. Muchas (si no todas) las distribuciones de Linux vienen con unos kernel (núcleos del sistema) que no utilizan más de 64MB de RAM por cuestiones relacionadas con la configuración de la BIOS. De todas formas, para solucionar este problemilla bastará con acceder a la línea que se encuentra en el archivo etc/lilo.conf:

```
append="mem=256M"
```

Aquí, el límite pasa a ser de 256 MB. No se olvide de ejecutar lilo después de modificar el archivo de configuración.

Si piensa que httpd está bien construido, pero aún cree que los problemas de embotellamiento se pueden deber al servidor Apache, conviene que le eche un vistazo a los archivos de configuración. Algunas de las directrices de Apache cuestan bastante (desde el punto de vista de la efectividad) y suelen ser las relacionadas con la resolución de nombres de dominio, llamadas al sistema, discos I/O y manipulación de procesos. A la hora de crear los archivos de configuración de Apache conviene que siga estas reglas:

- Utilice menos DNS.
- Reduzca las interrupciones de disco.
- Limite los thread de los procesos.

Utilice menos DNS

Si utiliza directrices que pueden trabajar indistintamente con nombres de host o con direcciones IP, use éstas últimas para evitar los problemas relacionados con la resolución de nombres de dominio. Por ejemplo, si utiliza directrices como:

```
allow from .nitec.com
```

o

```
deny from .nitec.com
```

cada una de ellas puede llegar a ser bastante cara desde el punto de vista de la efectividad. El servidor puede realizar varias llamadas para solucionar nombres y determinar la validez de cada petición que caiga fuera de su dominio.

En un servidor muy ocupado, este proceso puede llegar a ser algo molesto, por eso conviene utilizar las direcciones IP en vez de los nombres de dominio. De esta forma, el sistema no tendrá que efectuar ninguna resolución de nombres. La desventaja de este sistema es que se complica la administración de la configuración según se van añadiendo más direcciones IP. ¿Quién dijo que la eficacia era barata?

Existen muchos administradores con poca experiencia activan la directiva HostnameLookup para hacerse con los nombres de los host que se han guardado en los archivos de registro. No es muy buena idea porque así hay que resolver los nombres del host, con lo que se ralentizan los procesos.

Reducir las interrupciones de disco

Si no trabaja con los archivos de configuración para directorios, conviene que le indique a Apache que no los busque. Si no lo hace, Apache buscará en todos los directorios que se encuentren dentro del dominio de un URL, con lo que se produce una cierta cantidad de interrupciones de disco completamente innecesarias. Para desactivar esta función tendrá que dar el valor none a AllowOverride.

Utilizando nuevos conceptos como FastCGI o mod_perl (el módulo de Apache que se encuentra dentro de Perl), puede escribir aplicaciones Web que se ejecuten en la memoria caché, con lo que se reduce el número de interrupciones de disco. Para los sitios que tengan mucho tráfico, esta solución puede influir positivamente en los servidores Web.

Limitar los thread de los procesos

Bajo la implementación de Apache para UNIX, el proceso del servidor crea un thread que es quien se encarga de controlar las peticiones. Hay una serie de directrices que se pueden usar para manipular la cantidad de thread que se van a utilizar y el tiempo que van a esperar antes de finalizarse. Si el servidor que administra suele verse sometido a grandes cargas, es posible que prefiera utilizar directrices como MinSpareServers, MaxSpareServers o MaxRequestsPerChild, ya que le ayudarán a conseguir mejores resultados. También se puede limitar la cantidad de recursos del sistema que utilizará Apache y sus thread cuando procesa directrices tales como RLimitCPU, RLimitMEM, RLimitNPROC.

Una vez que termine de ajustar la configuración de Apache, conviene que se asegure de que ha eliminado todo el software innecesario. En los sistemas UNIX y NT, hay muchos programas (llamados servidores/demonios en UNIX y servicios en NT) que se quedan funcionando en un segundo plano. Tendrá que determinar cuáles son y cuáles se pueden eliminar. Por ejemplo, verá que se está ejecutando el servidor de impresión (lpd) y, según los casos, es posible que prefiera desactivarlo. Además de finalizar el proceso en ejecución, compruebe que no vuelve a ejecutarse la siguiente vez que se ejecute el servidor.

Siempre es conveniente volver a iniciar el servidor después de efectuar cambios en el software. Esto le permitirá verificar que la máquina sigue funcionando como estaba previsto. Es especialmente útil cuando tiene que eliminar o desactivar más de un demonio del proceso de arranque. Asegúrese de que su ordenador funciona como es debido, para lo cual tendrá que ejecutar los servicios con los que suele trabajar. Por ejemplo, si tiene el servidor de correo funcionando en su sistema, conviene que intente enviar y recibir mensajes de correo electrónico después de reiniciar el sistema, o por lo menos que compruebe que el demonio aún se está ejecutando.

La red

Un servidor Web está conectado a una red, por lo que la configuración tiene un impacto importante en la eficacia de los servidores Web. Dependiendo de cómo se esté utilizando el servidor Web, es posible que tenga que hacer una serie de ajustes en la red. Las instalaciones típicas de los servidores Web en una red, son las siguientes:

- Servidor Web ubicado en una casa.
- Servidor Web perteneciente a un ISP.
- Servidor Web de una intranet.
- Red Web repartida.

Servidor Web ubicado en una casa

Muchas organizaciones se conectan a Internet e instalan sus servidores Web en sus propias redes. Generalmente, estas conexiones no tienen el suficiente ancho de banda. Si su servidor no tiene una gran demanda, la verdad es que esta solución puede ser más que suficiente. Además, todo lo que quiere

hacer es atender a sus posibles clientes y ayudarles a encontrar la información que buscan sobre la empresa.

Si la cantidad de información que tiene que ofrecer es grande, al igual que la demanda de su servidor, obviamente tendrá que trabajar con un ancho de banda mayor. La experiencia me ha enseñado que la mayoría de la gente responsable de adquirir el ancho de banda adecuado no hace los cálculos necesarios que les permitirán determinar lo que necesitan. Para hacerse una idea de cuáles son sus requisitos con respecto al ancho de banda, bastará con que efectúe los cálculos que le describimos a continuación.

Tome el tamaño de su página Web más grande, identifique las horas punta diarias y los éxitos de transmisión diarios. Determine la media de bytes transferidos por día por medio de la fórmula: (número máximo de éxitos diarios x tamaño medio de las páginas) / total de horas punta diarias. Una vez que conozca el requisito medio de transferencias diarias en una hora punta, podrá calcular los requisitos de ancho de banda de la siguiente forma: (media de bytes transferidos en hora punta / 3600 segundos). Así obtiene la media de bytes/seg que tendrá en las horas punta. Si ve que el ancho de banda con el que trabaja puede abarcar este ancho de banda, no tendrá ningún problema.

Nota: la mayoría de la información relacionada con el ancho de banda se mide en kilobits/seg (Kbps) o megabits/seg (Mbps). Puede utilizar este sistema de medida para calcular la media de bytes/seg. Para convertir la unidad obtenida con la fórmula anterior a Kbps o Mbps, tendrá que emplear la siguiente fórmula:

```
Kbps = (bytes / 360) * ( 8 bits / byte) * ( 1 Kilobits / 1024 bits)
Mbps = (bytes / seg) * (8 bits / byte) * ( 1 Kilobits / 1024 bits)
* ( 1 Megabits / 1024 Kilobits)
```

Generalmente es conveniente tener el doble de ancho de banda del requerido. Cuando esté seguro de que su conexión a Internet tiene el ancho de banda suficiente, se centrará en la disposición del servidor Web en la red de trabajo. La figura 17.3 muestra un servidor Web conectado a una red LAN que accede a Internet a través de un router.

¿Qué hay de malo en esta configuración? Como se puede ver en la figura, todo el tráfico de la red LAN (Ethernet) para por el servidor Web. Es decir, que en una red LAN con mucho tráfico el servidor Web se comportará como un PC más porque comparte la red con todos ellos. Además, es una configuración muy arriesgada porque, a través del router, se pueden ver todas sus máquinas. La eficacia que se consigue con este esquema tampoco es buena, y de

la seguridad del sistema mejor no hablar. Para mejorar estos dos aspectos, convendría segmentar la LAN como se puede apreciar en la figura 17.4.

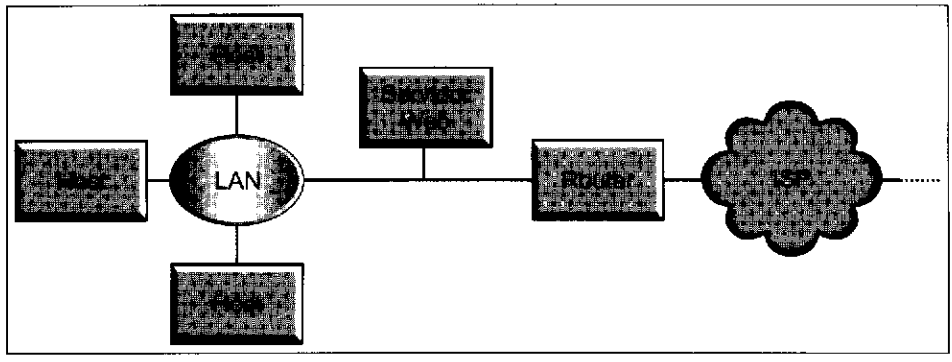


Figura 17.3. Servidor Web en una red LAN conectada a Internet

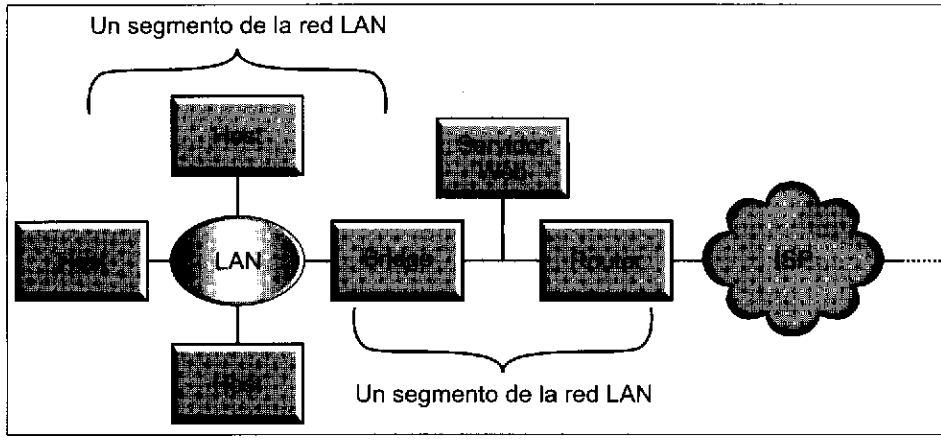


Figura 17.4. Servidor Web en una red LAN segmentada conectada a Internet

Ahora el tráfico de la red LAN se mantiene dentro del segmento y el servidor Web, desde Ethernet, no puede acceder a esta información. De hecho, el único tráfico que pasará por el servidor Web será el que se intercambie con Internet a través del router. De esta forma se libera a la red Ethernet del tráfico propio de la red LAN y se optimiza la eficacia del servidor Web para atender la recepción y envío de paquetes a la Red. El bridge se puede implementar por medio de hardware o simplemente mediante un PC que utilice un sistema operativo que pueda enviar el tráfico IP de una red a otra.

De modo que lo que ha de recordar al conectar un PC a Internet es que debe contar con el suficiente ancho de banda para poder atender los picos de

transacciones que tienen lugar en las horas punta y que el tráfico de su red LAN no se puede mezclar con el que pasa por el servidor Web.

Instalar su propio servidor Web no siempre es posible, principalmente por cuestiones económicas. Si no dispone del personal especializado suficiente que se haga cargo del mantenimiento de los sistemas y de las redes, mi consejo es que instale su servidor Web en un ISP.

Servidor Web perteneciente a un ISP

Instalar un servidor Web en la red de un proveedor de servicios de Internet (ISP) tiene varias ventajas. Si escoge el ISP adecuado, no tendrá que preocuparse por la congestión propia del ancho de banda. El ISP se hará cargo de todos estos problemas. A la hora de escoger un ISP, tenga en cuenta que el mercado hay muchos que le ofrecen la posibilidad de instalar su servidor en sus sistemas, por lo que convendrá decantarse por el mejor. Antes de firmar nada con nadie, visítelos y pregunte cualquier duda que le surja.

Pregunte cómo piensan aislar su servidor del resto de máquinas instaladas en su sistema. Pídales que lo instalen en un segmento de su red. De hecho, éste ha de ser un requisito incondicional, ya que no querrá que nadie tenga acceso a su servidor y pueda llegar a husmear en su contenido o en los paquetes de datos que transmite. Si un ISP le comenta que el segmento de la red Ethernet en el que instalará su servidor está compartido con otros servidores, recuerde que cualquiera de sus usuarios podrá visualizar sus paquetes de datos utilizando aplicaciones como tcpdump o similares. Esto es algo completamente inaceptable por razones de seguridad. Pregúntele qué tipo de conexión tiene con Internet y con qué red trabaja. Hay muchos proveedores que disponen de más de un router para enviar información a Internet.

Pero también tendrá que hacer sus deberes. Puede efectuar una investigación muy simple con algunas herramientas TCP/IP especializadas, fáciles de utilizar. Por ejemplo, puede efectuar un ping al sitio Web del ISP que más le guste. Cuanto menos tiempo tarde, mejor conexión tendrá.

Si tiene acceso a un programa de seguimiento (como tracert, que funciona en plataformas Windows NT), siga la ruta del servidor Web del ISP. Cuente el número de saltos (columnas que aparecen a la izquierda de la salida) que se producen entre su ordenador y el sistema del ISP. Repita la operación dirigiéndose a más de un sistema de Internet. Puede pedirle a unos cuantos amigos suyos que vivan en ciudades distintas a la suya que le ayuden a trazar la ruta. Recolecte tantos datos como le sea posible.

Si efectúa los mismo experimentos con los distintos ISP, podrá hacerse una idea de cómo funcionan. Es verdad que los métodos son algo primitivos,

pero le permiten hacerse con algo de información de valor gracias a la cual podrá determinar cómo están conectados a Internet. Le aconsejo que efectúe estas pruebas con más de un ISP a la vez para que comprenda mejor los resultados obtenidos. Y repita los experimentos en distintas franjas horarias. De esta forma obtendrá unos datos bastante más realistas.

Gracias a estas pruebas aumentará las posibilidades de acertar a la hora de escoger el proveedor de servicios de Internet.

Servidor Web de una intranet

Si usa su servidor Web con una intranet, apenas tendrá que preocuparse del rendimiento de su red, a menos que ésta esté congestionada. Si es el caso, coloque su servidor en un segmento menos congestionado o cree otro nuevo para aislar el servidor y ver si el único factor que afecta a su efectividad es el tráfico.

Si la Red es un medio de expansión muy importante para su negocio, es muy posible que su servidor se vea sometido a una gran carga. Es posible que un único servidor no sea suficiente para hacerse cargo de todo el trabajo que se le confía. Por eso conviene que considere la posibilidad de distribuir el trabajo entre dos o más servidores Web.

Red Web repartida

Cuando sus sitios Web se conviertan en nodos sometidos a mucho tráfico, habrá llegado el momento de distribuir la carga de trabajo. Si ha utilizado los trucos que hemos visto en las secciones anteriores, es posible que el nivel de efectividad que consiga con un único servidor sea más que suficiente para sus requisitos. De todas formas, hay casos en los que un único servidor no es suficiente, sobre todo cuando se trabaja con sistemas Web con mucha carga. En estos casos lo que tendrá que hacer será distribuir la carga entre otros sistemas. Hay dos formas de hacerlo. Vamos a ver cada método con detenimiento y a analizar sus inconvenientes.

Desvío de tareas hacia servidores especializados

Este método se puede utilizar cuando el servidor Web se ve saturado de trabajo que pueden desarrollar otros sistemas. Por ejemplo, si ejecuta cierto número de aplicaciones CGI en el servidor Web y tiene problemas a la hora de hacer frente a las peticiones, es posible que prefiera crear uno o más servidores Web que se encarguen de este tipo de tareas, con lo que se reducirá la carga de trabajo del servidor Web. En la figura 17.5 tenemos un ejemplo una de estas redes.

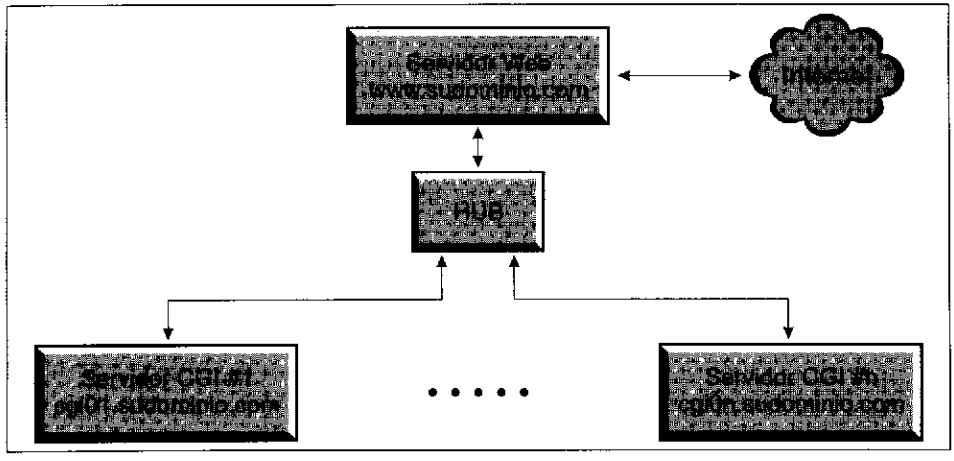


Figura 17.5. Desvío de la carga CGI hacia servidores externos

En esta figura, el servidor www.sudominio.com está conectado a uno o más servidores CGI, como cgi01.sudominio.com. Para utilizarlos tendrá que actualizar los vínculos en sus formularios HTML que llaman a los script CGI. En el listado 17.1 tiene el formulario www.sudominio.com.

```
<HTML>
<HEAD>
<TITLE> Order Form </TITLE>
</HEAD>
<BODY BGCOLOR="white">
<H1> Order Form </H1>
<HR>
<PRE>

<FORM ACTION="/cgi-bin/takeorder.pl" METHOD="POST">
  Last  Name <INPUT TYPE=TEXT NAME="Apellido" SIZE="30">
  First Name <INPUT TYPE=TEXT NAME="Nombre" SIZE="30">
  Email Addr <INPUT TYPE=TEXT NAME="E-Mail" SIZE="30">
  Product    <INPUT TYPE=TEXT NAME="Producto" SIZE="30">
  Quantity   <INPUT TYPE=TEXT NAME="Qty" SIZE="3">

  <INPUT TYPE=SUBMIT VALUE="Pedir">

  <INPUT TYPE=RESET VALUE="Limpiar todo">

</BLOCKQUTE>
</PRE>
</FORM>
</BODY>
</HTML>
```

Listado 17.1. Order.html

Los datos que introduce el visitante se envían a través de una petición HTTP POST al mismo servidor.

Para usar un servidor CGI externo (cgi01.sudominio.com) tendrá que modificar el formulario. La acción ACTION tendrá que cambiar de /cgi-bin/takeorder.pl a http://cgi01.sudominio.com/cgi-bin/takeorder.pl para que el servidor cgi01.sudominio.com sea quien reciba la petición. De esta forma se libera al servidor Web de todas las cargas de trabajo relacionadas con los script CGI, por lo que, mientras tanto, puede centrarse en atender otras páginas.

De todas formas, con esta configuración tiene que atender otros puntos importantes. La mayoría de los script CGI efectúan una tarea y generan unas páginas de salida. Estas páginas se tienen que construir de tal forma que todos sus vínculos apunten al servidor www.sudominio.com. Los vínculos de las páginas de salida que se generan han de utilizar referencias absolutas. Por ejemplo, el script takeorder.pl imprime una página HTML de salida, cuyo código fuente es el que aparece en el listado 17.2. También hay que modificarlo.

```
<HTML>
<HEAD>
<TITLE> Successful Order</TITLE>
</HEAD>
<BODY BGCOLOR="white">
<IMG SRC="/images/header.gif" HEIGHT=50 WIDTH=500 BORDER=0>
<P> Gracias por su solicitud</P>
<HR>
<BLOCKQUTE>
Le enviaremos a su dirección de e-mail el número de
pedido(kabir@mitec.com) para que lo utilice en futuras referencias.
<P>
Haga clic <A HREF="/index.html">aquí</A> para volver a la página
principal.
</P>
</BLOCKQUTE>
<CENTER>
<IMG SRC="/images/copyright.gif" HEIGHT=20 WIDTH=400 BORDER=0>
</CENTER>
</FORM>
</BODY>
</HTML>
```

Listado 17.2. Salida de takeorder.pl

El enlace URL y la fuente de la imagen se tendrán que modificar de:

```
<A HREF="/index.html">aquí</A>
```

a

```
<A HREF="http://www.sudominio.com/index.html">aquí</A>
```

Lo mismo ocurre con los atributos de la fuente de la imagen, ya que tendrán que ser www.yourcompany.com/images/ en vez de simplemente /images/. Es posible que se pregunte que pasaría si tuviera una copia de todos los archivos en el servidor principal y otra en el servidor CGI. Bueno, en este caso los usuarios permanecerían en el servidor CGI después de que se completase la ejecución del script con la consecuente confusión provocada al encontrarse cgi01.sudominio.com en el URL. Además, algunos visitantes habituales incluirían ciertas partes del servidor CGI dentro de favoritos para acceder directamente a ellas en futuras conexiones. Si únicamente tiene un servidor CGI, se encontrará con que tiene dos servidores (uno especializado en un tipo de trabajo) que desarrollan indistintamente un tipo de trabajo o el otro. De esta forma no se aprovecha el hecho de tener dos servidores.

Además, la verdadera razón por la que se tiene un explorador CGI externo es que no tendrá que utilizar demasiada CPU ni abusará de las interrupciones I/O. Por eso, asegúrese de que las páginas CGI que se generan tengan bien configurados los enlaces para que vuelvan al servidor principal en el momento de completar el proceso del script. El ejemplo siguiente, goback.pl le muestra este tipo de acciones con un script CGI escrito en Perl:

```
#!/usr/local/bin/perl
#
print "Location: www.sudominio.com/thanks_for_order.html\n\n\n";
exit 0;
```

Siempre que se ejecute un script utilizando el URL http://cgi01.sudominio.com/cgi-bin/goback.pl, se devolverá un visitante a la página thanks_for_order.html que se encuentra en el servidor www.sudominio.com. Esta técnica se puede utilizar para procesar script que redirijan automáticamente a los visitantes al servidor principal. De esta forma no habrá muchas posibilidades de que los usuarios lleguen a saber (a menos que miren en código fuente del formulario HTML) que el script CGI se ha procesado en otra máquina.

Como se puede observar, este tipo de red puede resultar de gran ayuda a los sistemas Web que tengan mucha carga de trabajo. Tenga en cuenta que aplicar esta distribución a una red con mucho tráfico, además de ayudar al servidor Web, creará un entorno más seguro. Por ejemplo, supongamos que tenemos una base de datos que le gustaría conectar con su servidor Web para que los visitantes puedan intectuar con su contenido. Si le preocupa la seguridad de su sistema, puede utilizar un esquema como el que aparece en la figura 17.6.

En este caso, el servidor en el que se encuentra la base de datos está en una parte de la red IP (192.168.1.0) a la que no tiene acceso el servidor router del

sistema y que únicamente está conectada con dos servidores CGI. Estos están conectados a su vez con Internet a través de una red que cuenta con un router (206.171.50.0). Todo lo que tiene que hacer es instalar el software del cliente de la base de datos en los servidores CGI para que puedan acceder al servidor en el que se encuentra almacenada la información, situado en el segmento de la red al que no puede acceder el router. De esta forma puede distribuir la carga entre tres sistemas externos y disfrutar de una gran eficacia y alto nivel de seguridad.

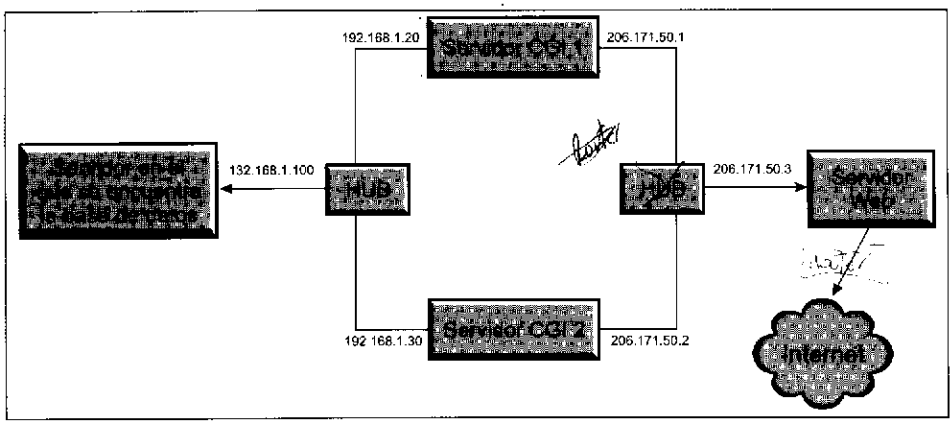


Figura 17.6. Red de trabajo de gran eficacia y seguridad

El grado de seguridad depende de lo bien protegidos que estén los servidores CGI, porque cualquiera que irrumpa en uno de los dos sistemas puede llegar a interactuar, o incluso entrar, en el servidor de la base de datos. Pero como dicho servidor se encuentra completamente aislado, es muy difícil que alguien llegue a conocer su existencia a través de la configuración de la red.

Duplicar servidores Web para equilibrar la carga

Si su sitio Web contiene un gran número de documentos estáticos y un número moderado de documentos generados desde un script CGI (o cualquier otro tipo de documento dinámico) y el sistema está sometido a una gran carga, es posible que tenga que duplicar su red Web para equilibrar las cargas entre varios servidores Web.

Por ejemplo, tomamos tres servidores Web, cada uno con el mismo nombre de host (www.suempresa.com) pero con direcciones IP diferentes. ¿Cómo es posible? La respuesta la encontramos en la configuración DNS.

Usando un truco DNS es posible asignar el mismo nombre a varias direcciones IP. La configuración DNS para la red de este ejemplo sería la siguiente:

```
www     IN A  206.171.50.1
www     IN A  206.171.50.2
www     IN A  206.171.50.3
```

La dirección (A) le dice al servidor DNS que, cada vez que se reciba una petición URL dirigida a www.suempresa.com, suministre las direcciones IP en el mismo orden en que aparecen en la lista. El servidor DNS entrega la primera dirección IP (206.171.50.1), luego la segunda y, por último, la tercera. Recicla la lista constantemente.

Mientras mantenga exactamente iguales los servidores Web, el usuario nunca podrá notar ninguna otra diferencia y se repartirá la carga de trabajo entre varios sistemas. Cuando un visitante se dirige hacia www.suempresa.com/, el servidor DNS le entregará la dirección siguiente en la lista a la que entregó a la última petición. El explorador utilizará ese mismo servidor Web para que le atienda durante toda la consulta. De este modo, el reparto de trabajo se efectúa enviando a los distintos clientes a diferentes servidores.

Pero con esta configuración rápidamente aparecen dos problemas. El primero es la sincronización entre los servidores. Si trata de mantenerla a mano, tendrá que actualizar los archivos a través de FTP (en los tres servidores). Demasiado trabajo. La solución es actualizar todo el material en un único servidor y apoyarse luego en software de actualización, como rdist, para que se encargue de copiar las nuevas versiones de los archivos en el resto de servidores. Así se puede automatizar el proceso de actualización para que se efectúe con cierta periodicidad.

Luego tenemos otro problema. En su sistema tiene algunos script CGI que generan archivos de datos y otros que se encargan de buscar estos ficheros. Por ejemplo, supongamos va a promocionar su sitio Web a través de una aplicación encargada de generar lotería. Sus visitantes se registrarán en el sistema y recibirán una cookie HTTP que hará las funciones de identificador. Indicará a sus visitantes que para obtener puntos tendrán que visitar ciertas páginas Web. Estos puntos se graban en el disco y se irán acumulando. Se basará en ellos para ver qué visitantes han invertido más tiempo revisando sus páginas Web. Gracias a un script CGI que lee los archivos en los que se guarda esta información, podrán saber en todo momento la cantidad de puntos que cada uno tiene almacenados. Como se está trabajando con una réplica de un servidor Web, los puntos de los usuarios se guardan en el cada servidor y se crea y lee un archivo de registro para cada servidor.

Por lo tanto, cuando un visitante acude a un sitio Web y se registra en la promoción de la lotería, todo va bien. Pero cuando vuelve al día siguiente y se le envía a un servidor en el que no ha estado con anterioridad, se sorprenderá cuando vea que en realidad no tiene almacenado ningún punto. ¿Cómo se

puede solucionar este problema? Pues introduciendo en todo este esquema el sistema para archivos de red o NFS. Tiene que preparar una zona del disco de un servidor y efectuar los cambios necesarios en los script del resto de los servidores para que escriban sus registros en este espacio reservado. De esta forma, todos los servidores compartirán datos.

Esta configuración DNS le permitirá crear una réplica de la red de un servidor Web, con lo que podrá equilibrar la carga de trabajo, además de mejorar considerablemente la eficacia que obtendría si utilizase un único servidor. Otra de las grandes ventajas de este tipo de comparación es que es bastante económica. De todas formas, también hay que reconocer que tiene unas cuantas desventajas. Siempre que quiera introducir algún cambio en su configuración DNS, tendrá que tomarse su tiempo para dispersarla por Internet. Si uno de los servidores se viene abajo y quiere redirigir sus peticiones a otra de las máquinas que están en funcionamiento, se encontrará en otra situación complicada. También debe tener en cuenta que hay un límite relacionado con la cantidad de peticiones que se pueden atener por cada registro DNS. En la actualidad el límite es de 32, es decir, que puede tener como mucho 32 servidores duplicados a la vez. Si está buscando una solución mejor, sepa que le va a costar muchísimo más caro. Quizá una alternativa podría ser el uso de paquetes de hardware.

Problemas relacionados con el reparto de carga

Uno de los problemas relacionados con el reparto de cargas en sistemas que cuenten con varios servidores es que el equilibrio nunca llega a ser pleno. En cualquiera de las configuraciones anteriores, no podrá determinar qué servidor está ocupado y cuál está libre. Para alcanzar un equilibrio preciso hay que apoyarse en ciertos componentes de hardware.

Ahora es cuando por fin se empiezan a ver en el mercado componentes de hardware especializados en el reparto de cargas de trabajo. Algunos reciben el nombre de multiplexores y otros simplemente son interruptores Etehernet inteligentes equipados con algoritmos de balance de carga que detectan la cantidad de trabajo de los servidores desde el exterior. Estos componentes de hardware disponen de una serie de características que encarecen su precio. De todas formas, estas características, entre las que se incluyen la detección de fallos y redirección de las peticiones HTTP a un servidor Web funcional, representan una serie de ventajas que conviene tener en cuenta a la hora de determinar los componentes de su sistema. Si decide que esta opción es factible, puede consultar la información publicada por los distribuidores de este tipo de componentes, tales como CISCO (www.cisco.com/) e HydraWeb (www.hydraweb.com/).

El contenido

Cuando esté completamente seguro de que su ordenador Apache, el software y la red están perfectamente ajustados, podrá concentrarse en el contenido de su ordenador.

El contenido estático, como las páginas HTML y las imágenes, no representa tanta carga como el contenido dinámico necesario para que funcionen aplicaciones como los script CGI, Java y bases de datos. Para reducir la carga del servidor Web, conviene que trate de acelerar su producción. Por ejemplo, si utiliza Perl para crear los script CGI, convendría que trabajase con mod_Perl o FastCGI.

También se puede reducir la carga del contenido dinámico del servidor Web creando aplicaciones que utilicen script o applet que se encuentren en el lado del cliente. De esta forma se efectúa gran parte del trabajo antes de establecer la comunicación con el servidor Web. Por ejemplo, al utilizar script o applet en Java, se puede obligar a que sea el explorador quien se encargue de efectuar las tareas de validación. Además, con los applet en Java se puede desviar la mayor parte del trabajo del servidor Web para que se efectúe en el host del cliente.

Siempre que sea posible conviene que reduzca al máximo el uso de SSI en los script. Tenga cuidado a la hora de escoger la extensión del nombre de archivo SSI, porque el servidor lo tratará como si fuese algo que tiene que analizar por si aparece alguna llamada SSI. Por ejemplo, si a un archivo SSI le da la extensión .html y el servidor tiene gran cantidad de páginas HTML en las que no hay ningún comando SSI, reducirá considerablemente su eficacia.

Aunque el contenido dinámico es el responsable de la mayor parte de la carga del servidor, el contenido estático se puede utilizar de muchas formas para optimizar la efectividad de su servidor. Le aconsejo que tenga en cuenta los siguientes puntos a la hora de desarrollar su contenido estático:

- Utilice metaetiquetas HTML, anime a sus clientes y también a los servidores proxy a que carguen en la memoria caché las páginas que no van a cambiar durante algún tiempo.
- Utilice los atributos HEIGHT y WIDTH con las etiquetas IMG para acelerar la velocidad de carga de las páginas HTML.
- Utilice imágenes pequeñas.
- Efectúe las conversiones de imágenes en el lado del servidor para reducir la carga correspondiente en el lado del servidor.

El personal

La eficacia de un sitio Web también depende del personal que la administra y desarrolla. Por lo tanto, conviene que tenga en cuenta que no basta con el ajuste del hardware, software y red para convertir su servidor Web en una máquina de gran velocidad. También hay que considerar el factor humano. Aquí tiene algunos consejos.

- Evite que los usuarios se registren en los servidores y utilicen sus cuentas para desarrollar sus trabajos.
- Suministre a su equipo de desarrollo Web las últimas herramientas y tecnologías. Ofrezca cursos de formación.
- Comente a todo el mundo que siempre es conveniente invertir en optimizar la eficacia, pero sin sacrificar nada más. Por ejemplo, no es aconsejable que el desarrollador HTML inserte direcciones IP en los vínculos para reducir las peticiones DNS que se efectúan en el lado del explorador. Un visitante pensará que la velocidad de descarga de la página es mayor, pero a la hora de cambiar dichas direcciones, tendrá que actualizar todos los vínculos.

El objetivo que se tiene que perseguir con los servidores Web debe ser el mismo que se utiliza con los grupos Apache: en primer lugar, correcciones; luego, velocidad.

18 Ejecución de sitios Web perfectos

Es muy posible que a los sitios Web que ya tenía haya añadido su servidor Apache. Todo el mundo en su empresa admira su trabajo. Piensa en la Red como en el "Paraíso", ¿verdad? Pues muy mal hecho. En muy poco tiempo llegarán sus compañeros pidiéndole que actualice sus páginas. Por ejemplo, el departamento de marketing le llamará para que actualice la lista de precios y el departamento jurídico le pedirá que incluya una serie de avisos legales en ciertas páginas.

Los administradores de sistemas Web medios-grandes están acostumbrados a este tipo de trabajos. En breve se encontrará con una lista interminable de peticiones de actualizaciones. ¿Cómo puede administrar su red si tiene tanto que hacer? En este capítulo aprenderá a administrar un sistema Web profesional permitiendo que todos los implicados en el proceso de desarrollo y conservación de un sitio Web estén en perfecta sincronía con la Red.

Crear un ciclo Web

Desgraciadamente, los proyectos de un sitio Web no suelen comenzar con el diseño de una red que sea fácilmente administrable. En la mayoría de los casos, se consume mucho tiempo consiguiendo que los servidores funcionen

y desarrollando su contenido. Se dedica muy poco tiempo al desarrollo de un esquema que permita su administración. Irónicamente, tan pronto como todo parece funcionar a la perfección, comienzan a fallar las cosas, porque no se ha tenido en cuenta un factor muy importante: la administración. Se entra en un círculo vicioso. En esta sección aprenderá qué es eso del "ciclo Web", que le permitirá crear un sistema Web de sencilla administración.

Un ciclo Web consta de tres fases: desarrollo, construcción y producción. Implementando cada una de estas fases, puede crear un sitio Web cuya administración sea muy sencilla. En la figura 18.1 se puede ver el esquema del ciclo Web.

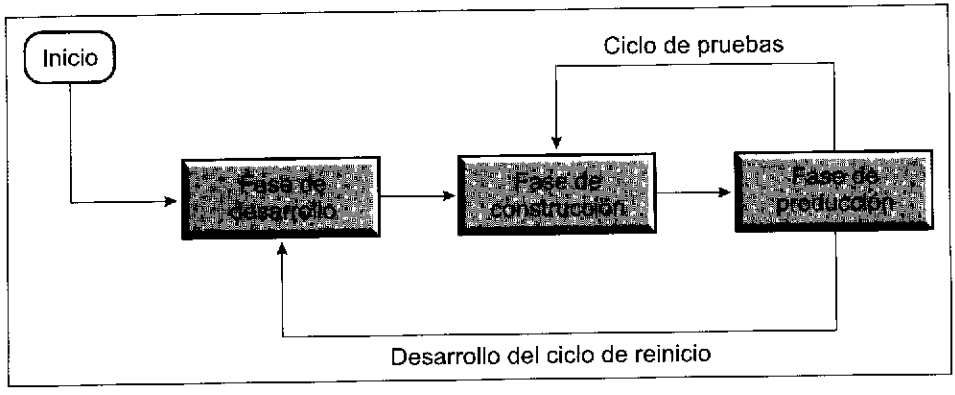


Figura 18.1. Diagrama de un ciclo Web de alto nivel

Como se puede ver en esta figura, el ciclo Web comienza con la fase de desarrollo, continúa con la de construcción y termina con la de producción. Cuando se reinicia, comienza en la fase de producción y se repite todo el ciclo. Vamos a ver detenidamente cada una de estas fases:

- Desarrollo. En esta fase se comienza por desarrollar el contenido Web. Este puede consistir en documentos HTML o script CGI, que se construirán y probarán en esta fase. Una vez que los desarrolladores están completamente seguros de que su trabajo está listo para incluirlo en el sitio Web, se pasa a la siguiente fase.
- Construcción. La finalidad de esta fase es permitir que el nuevo contenido se integre perfectamente con el ya existente, además de efectuar una serie de pruebas. Estas pruebas no las efectúan los desarrolladores, sino personal especializado en este tipo de tareas. La razón es muy sencilla. No se deja que los desarrolladores sean quienes prueben su propio trabajo debido a un exceso de confianza. Los problemas terminarán en el

momento en que se pasen satisfactoriamente todas las pruebas. En el último caso, se pasa a la fase de producción. Si hubiese algún problema, habría que volver al principio del ciclo, para que los desarrolladores corrigiesen lo que estuviese mal. Pero estas correcciones se efectuarían en la fase de desarrollo. Nunca permita que un desarrollador corrija fallos en la fase de construcción.

- Producción. En esta fase se copian todos los contenidos y se desarrollan las tareas para las que se han construido. En primer lugar, se hace una copia de seguridad del contenido (funcional) y se mueve desde la zona reservada para la construcción hasta la reservada para la producción de los sitios Web. Este intercambio tendrá que efectuarse en el menor tiempo posible para reducir el periodo de desconexión y evitar que los visitantes pierdan los datos que se bajan de la Red.

Cuando esté listo para comenzar otro ciclo de desarrollo (es decir, para reiniciar todo el proceso), copiará el contenido que se encuentra en la fase de producción y lo pondrá a disposición de la fase de desarrollo, para que los desarrolladores puedan trabajar con él. El ciclo se repetirá tantas veces como sea necesario.

¿Qué beneficio le aporta todo esto? Le aporta una serie de opciones relacionadas con la administración y efectividad. Por ejemplo, si en la actualidad se limita a desarrollar el contenido Web y colocarlo en el sistema de producción antes de haberlo sometido a todas las pruebas pertinentes, está asumiendo demasiados riesgos. En la mayoría de los casos, los desarrolladores afirman haber probado todos sus trabajos en sus sistemas locales, por lo que parece que están ansiosos por integrarlos al sistema final y completar su trabajo. Como en estos sistemas locales no tienen todo el contenido ya existente en el sistema Web, no pueden ver cómo será su integración, luego el resultado de las pruebas efectuadas no son todo lo reales que deberían. Sólo al integrar el nuevo material con el antiguo se pueden detectar las posibles incompatibilidades. Por ejemplo, sin la fase de construcción, un fallo del sistema de producción procedente de la fase de desarrollo podría producir los siguientes fallos:

- Los archivos que hay en el sistema de producción se pueden sobreescribir al implementar el nuevo contenido. Esto suele pasar con las imágenes, ya que la carencia de un convenio relacionado con la nomenclatura puede hacer que un archivo o directorio nuevo sobreescriba al ya existente.
- Los archivos que se encuentran en el sistema de producción se pueden llegar a anular, porque los desarrolladores CGI utilizan datos antiguos cuando crean el nuevo contenido.

- Cuando se trabaja con muchos desarrolladores a la vez, es muy posible que vuelvan a aparecer archivos antiguos en el servidor de producción, porque cada uno puede haber creado su copia de los archivos en momentos distintos. De esta forma cada desarrollador tiene que volver a la fase de creación para depurar su copia, con lo que se puede llegar a crear una gran confusión.

Si se trabaja con varios desarrolladores o se conectan entre sí los distintos proyectos, es muy probable que aparezcan nuevos problemas. No se puede arriesgar a que aparezcan todos estos problemas en su servidor de producción. Por eso se implementa la fase de construcción. Vamos a ver cómo se puede aprovechar Apache para implementar estas fases.

Poner en acción el ciclo Web

Ya estamos listos para poner en acción el ciclo Web.

En primer lugar, tendrá que configurar su servidor para que trabaje con el ciclo Web. Hay muchas formas de hacerlo, pero aquí veremos tres. De los tres métodos, dos son para organizaciones que tan sólo tienen un servidor Web y el tercero para las que tienen varias máquinas Apache.

Lo ideal es que no se tenga que efectuar ningún trabajo de desarrollo en el servidor de producción. Pero si la configuración de su sistema no le permite disponer de varias máquinas dedicadas al ciclo Web, tendrá que utilizar un único servidor para completar todas las fases del ciclo.

Una vez que ha configurado su servidor Web, estará listo para implementar el ciclo Web.

Configuración para el ciclo Web

Tiene dos formas de configurar el sistema para implementar el ciclo Web: utilizar dos host virtuales para implementar las fases de desarrollo y construcción en el servidor de producción, o bien crear tres servidores Apache independientes, uno para cada fase.

Si en la fase de desarrollo tiene que ajustar los archivos de configuración de Apache o probar las nuevas versiones de este servidor, conviene que utilice ficheros de configuración independientes para el servidor de producción. De todas formas, si en la fase de desarrollo no suele implementar los cambios relacionados con el servidor Apache, puede entonces utilizar la solución del host virtual.

Antes de entrar en detalles relacionados con la configuración del ciclo Web, me gustaría destacar que el ciclo necesita una estructura de directorios bien planificada. En la figura 18.2 puede verla.

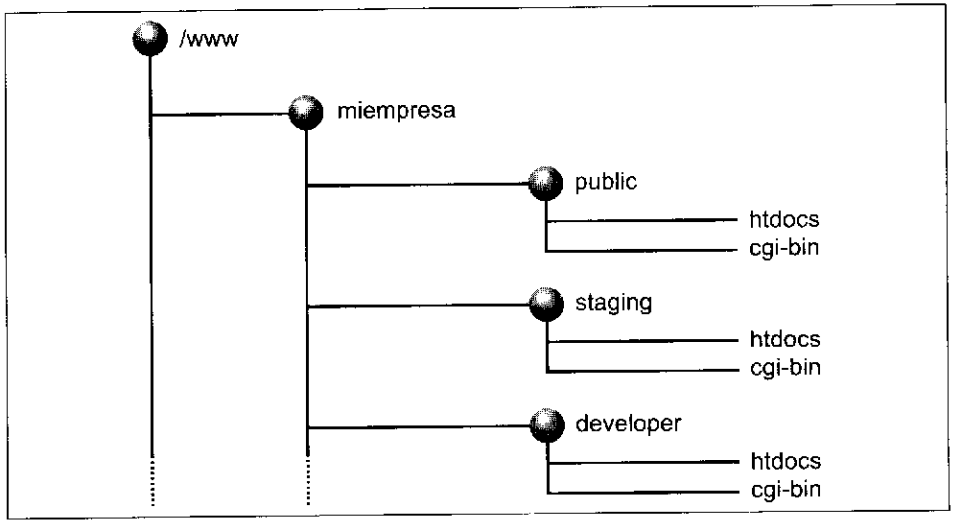


Figura 18.2. Estructura de directorios utilizada para los sitios públicos (public), de construcción (staging) y de desarrollo (developer) de miempresa

Este es un buen ejemplo en el que se puede ver la estructura de los directorios porque le permite conservar los sitios público, de construcción y de desarrollo de cada sistema Web dentro de un mismo directorio superior (en este caso miempresa).

Para añadir otro sitio Web, todo lo que habría que hacer sería copiar una estructura de directorios similar a la que aparece aquí.

En los ejemplos de configuración que veremos en las siguientes secciones, parto de la base de que dispone de la estructura de directorios que acabamos de ver. También supongo que el servidor de su organización se llama www.miempresa.com y que su dirección IP es 206.171.50.50. Asegúrese de sustituir estos valores por los suyos cuando implemente estos archivos de configuración en su sistema.

Crear un host virtual para cada fase

Si decide que no necesita modificar la versión de Apache, puede pasar a crear un host virtual para cada fase. Para ello tendrá que crear dos que tengan el mismo ServerName y que se ejecuten en dos direcciones distintas. En la tabla 18.1 se puede ver un ejemplo de la asignación de puertos a utilizar.

Tabla 18.1. Asignación de puertos para los servidores Apache del un ciclo Web

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

Puede utilizar la asignación de puertos que más le guste, siempre y cuando no use ninguno de los reservados para el sistema ni emplee un número superior a 65535. Conviene que el puerto del servidor de producción sea el 80 porque, por defecto, las peticiones HTTP se envían al puerto 80.

Para crear los host virtuales que se adapten a la asignación de puertos determinada en la tabla 18.1, tendrá que editar el archivo de configuración del servidor Apache (httpd.conf) como se indica a continuación.

Para que el servidor escuche a través de estos puertos, utilizará la directriz Listen:

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

A continuación creará los dos host virtuales:

```
# No olvide cambiar las direcciones IP ni los valores de las
# directrices ServerName, DocumetRoot, ScriptAlias, TransferLog, y
# ErrorLog para que utilicen las que se correspondan con su
# configuración.
#
<VirtualHost 206.171.50.50:1080>
ServerName
DocumentRoot /www/miempresa/staging/htdocs
ScriptAlias   /cgi-bin/       /www/miempresa/staging/cgi-bin/
TransferLog   logs/staging server.access.log
ErrorLog      logs/staging-server.error.log
</VirtualHost>

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

Observe que, en el ejemplo anterior, se utiliza la misma dirección IP en los dos servidores virtuales, pero en los contenedores <VirtualHost ...> aparecen

puertos distintos. Las direcciones IP coinciden con la del servidor www.miempresa.com. El valor de la directriz ServerName es el servidor principal. El resto de la configuración del servidor principal coincidirá con la habitual.

Para acceder a la zona de construcción se puede utilizar el siguiente URL:

```
www.miempresa.com:1080/
```

Para acceder a la zona de desarrollo se puede utilizar el siguiente URL:

```
www.miempresa.com:8080/
```

Si tiene pensado incluir la modificación de los archivos de configuración de Apache en el proceso de desarrollo, no utilice la configuración anterior. Tan sólo necesita la configuración de inicio de Apache, ya que si se modificase ésta, se podría llegar a afectar al rendimiento del servidor. En este caso podrá utilizar una única máquina, pero tendrá que trabajar con varios servidores Apache. Lo veremos a continuación.

Uso de varios procesos de servidores Apache

Si tiene previsto hacer pruebas con el propio servidor Apache dentro de la fase de desarrollo, convendrá que utilice varios servidores. Para implementar esta configuración el ciclo Web tendrá que seguir estas instrucciones.

Cree dos subdirectorios dentro del directorio de configuración del servidor Apache, llamados staging y developer:

```
mkdir /ruta/al/directorio/de/configuración/del/servidor/Apache/
staging
mkdir /ruta/al/directorio/de/configuración/del/servidor/Apache/
developer
```

No olvide que tiene que sustituir /ruta/al/directorio/de/configuración/del/servidor/Apache/ por la ruta del directorio de configuración que tenga el servidor de su sistema. Copie todos los archivos *.conf en ambos subdirectorios:

```
cp /ruta/al/directorio/de/configuración/del/servidor/Apache/*.conf /
ruta/al/directorio/de/configuración/del/servidor/Apache/staging/*
cp /ruta/al/directorio/de/configuración/del/servidor/Apache/*.conf /
ruta/al/directorio/de/configuración/del/servidor/Apache/developer/*
```

Ahora, modifique el archivo httpd.conf que se encuentra en el directorio /staging/ para que escuche a través del puerto 1080 en vez de hacerlo a través del predeterminado, el 80. Lo puede hacer con las directrices Port o Listen. Repita el proceso con el archivo httpd.conf, que se encuentra dentro del directorio developer, para que escuche a través del puerto 8080.

También debería modificar los archivos srm.conf (o httpd.conf) que están dentro del directorio developer y staging para que las directrices DocumentRoot y ScriptAlias apunten a la ruta adecuada. Por ejemplo, los cambios que habría que hacer para usar la estructura de directorios vista en la figura 18.2 serían:

```
DocumentRoot        /www/miempresa/staging/htdocs
ScriptAlias         /cgi-bin/   /www/miempresa/staging /cgi-bin/
```

para la configuración del sitio de pruebas. Para el de desarrollo:

```
DocumentRoot        /www/miempresa/developer/htdocs
ScriptAlias         /cgi-bin/   /www/miempresa/developer/cgi-bin/
```

Si utiliza una configuración especial para el servidor de producción, tendrá que usar la ruta absoluta, por lo que es posible que más tarde tenga que volver a modificar la configuración de staging y developer.

La idea es crear tres archivos de configuración diferentes que apunten a distintos DocumentRoot y ScriptAlias. Una vez hecho esto, puede iniciar los tres servidores Apache:

```
httpd -f /ruta/al/directorio/de/configuración/del/servidor/Apache/
httpd.conf httpd -f
/ruta/al/directorio/de/configuración/del/servidor/Apache/staging/
httpd.conf httpd -f
/ruta/al/directorio/de/configuración/del/servidor/Apache/developer/
httpd.conf
```

Cuando decida compilar una nueva versión de Apache y ejecutarla dentro del servidor de desarrollo, todo lo que tiene que hacer es utilizar el archivo de configuración del servidor de desarrollo. Por ejemplo, si quiere añadir un módulo nuevo y desea ver el efecto que tendrá sobre su contenido, bastará con que inicie los servidores de desarrollo y construcción con sus respectivos ejecutables, en vez de utilizar el del servidor de producción. Cuando termine de compilar los nuevos ejecutables, puede que sea conveniente cambiarles el nombre por algo como httpd-xx80, para asegurarse de que no se sobreescribirá accidentalmente el ejecutable del servidor de producción.

Ahora, vamos a ver cómo hay que configurar los distintos host implicados en el ciclo Web.

Uso de varios ordenadores como servidores Apache en el ciclo Web

Si puede comprarse varios ordenadores en los que ejecutará los distintos servidores Apache (es decir, uno para el servidor de desarrollo, otro para el

de construcción y un tercero para el de producción) del ciclo Web, no tendrá que utilizar ninguna configuración especial. Bastará con que instale Apache en todos los host y que trate a cada uno de acuerdo con la función que van a desarrollar. Como cada servidor se encuentra en un ordenador distinto, puede utilizar el puerto 80 en cada uno de ellos. Eso es todo lo que tiene que hacer en un ciclo Web que trabaje con varios host.

Implementar el ciclo Web

Llegados a este punto, deberá tener uno de estos tres entornos:

- Tiene un ordenador con dos host virtuales para desarrollo y construcción. El servidor de producción el servidor principal de Apache. Tenga cuidado a la hora de modificar la configuración de Apache porque los cambios pueden afectar al comportamiento del servidor de producción.
- Tiene un ordenador con tres servidores Apache principales, uno para el desarrollo, otro para la construcción y el tercero para la producción. Crea una configuración independiente para cada uno de ellos. Gracias a ella podrá efectuar cualquier prueba en un servidor sin molestar a los otros dos.
- Tiene tres ordenadores diferentes y en cada uno de ellos ejecuta un servidor de los tres que intervienen en el ciclo Web. Los tres ejecutan el servidor Apache a través del puerto 80.

Para iniciar el ciclo Web tendrá que copiar el contenido del árbol de documentos del servidor de producción en el sitio de desarrollo. Por ejemplo, si la configuración con la que trabaja es cualquiera de las dos primeras de la lista anterior, puede utilizar estos comandos de UNIX para proceder a la copia:

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

Copia todos los directorios del servidor de producción (y sus respectivos archivos) en el árbol de documentos del sitio de desarrollo. Asegúrese de modificar las rutas para que se usen las del nuevo sistema.

Para los entornos en los que se utilizan varios ordenadores, puede crear un archivo tar en su servidor de producción y copiarlo en el de desarrollo por FTP.

Compruebe los permisos para que Apache pueda leer todos los archivos y ejecutar los script CGI. Si tiene directorios en los que se restringe la escritura

a Apache (en los que los script CGI no pueden escribir datos), debería modificarlo. Inicie su servidor Apache.

Asegúrese de que el aspecto del sitio de desarrollo es exactamente igual que el que tenía en el sitio de producción (tiene que utilizar un explorador Web). Efectúe las comprobaciones oportunas y compruebe que los script funcionan como es debido.

Nota: si en el servidor de producción alguno de sus script produce URL codificadas, deberá hacer conservar este comportamiento en el sitio de desarrollo. Puede ignorar estos URL o arreglarlos para que los puedan usar las variables de entorno SERVER_NAME y SERVER_PORT.

Cuando todo funcione correctamente sabrá que ha creado un ciclo Web. Ahora pida a sus desarrolladores que coloquen nuevo material y script CGI en el sitio de desarrollo y pruébelos. Siempre que se complete algún trabajo, tendrá que probarlo en la zona destinada a ello. En la prueba tendrá que poner especial atención a lo siguiente:

- ¿Cumple con su objetivo? En otras palabras, el nuevo contenido, ¿cumple con sus requisitos?
- ¿Tiene algún efecto secundario? Por ejemplo, si el nuevo contenido es un CGI, tendrá que utilizar las propiedades Apache para depurar los fallos de los script (visto en los capítulos anteriores) y visualizar su funcionamiento.

Una vez que está satisfecho con los resultados de las pruebas, deje de desarrollar nuevos elementos en el contenido que acaba de crear para evitar tener que repetir las pruebas. Cuando llega el momento de actualizar el sitio de producción, tendrá que hacer una copia de su contenido y colocarla en el sitio de construcción. Asegúrese de que el contenido de ambos sitios es exactamente igual. Una vez que ha efectuado una serie de comprobaciones manuales para ver que todo funciona como es debido, podrá mover el nuevo contenido y los script al directorio staging e integrarlos.

También es conveniente mover todos los componentes de un proyecto a la vez para que pueda localizar los problemas propios de la fase de construcción. Por ejemplo, puede añadir tres nuevos script CGI a su sistema de desarrollo, moverlos a la vez al sitio de construcción y efectuar al mismo tiempo las pruebas de funcionamiento e integración. Si los script pasan todas las pruebas, puede empezar a probar el siguiente componente. Revise todos los

archivos de registro. Compruebe si aparece algo raro. Si todo va bien, estará listo para actualizar su sitio de producción. Pero tiene que tener mucho cuidado. Por ejemplo, si tiene una serie de script CGI en el servidor de producción que crean datos en el área de producción, no querrá sobreescribir estos datos con el contenido de la zona de construcción.

Por lo tanto, la mejor forma de actualizar su sitio de producción será cuando el servidor apenas esté ocupado. Tendrá que coger los archivos de datos que se encuentran en el sitio de producción y colocarlos en los directorios apropiados de la versión de construcción del sistema. De esta forma conservará la sincronización entre los dos sistemas, el de producción y el de construcción. Tenga mucho cuidado, porque el sitio de producción está "vivo" y no tendrá ninguna forma de saber en qué momento aparecerá un visitante y accederá a una página utilizando el script CGI con el que tiene que trabajar.

Para reducir la cantidad de tiempo puede crear un shell script que haga lo siguiente:

1. Copie en primer lugar todos los datos en las zonas adecuadas del sitio de construcción.
2. Renombre los directorios superiores de producción (por ejemplo, de public a public.old).
3. Renombre los directorios superiores de construcción (como public.old) para que tengan los mismos nombres que en el directorio de producción (por ejemplo, public).
4. Renombre todos los archivos del antiguo directorio de producción para que coincidan con los que tiene en el directorio de construcción.

De esta forma, tan sólo hay que hacer un par de cosas para que el sitio de construcción se convierta en el de producción. Además, evitará tener que trabajar con una gran cantidad de copias. El script que se correspondería con la figura 18.2 podría ser el que aparece en el listado 18.1.

```
#!/bin/sh
# Propósito: un shell script que copia los archivos de datos
# en la zona de construcción y renombra ésta para que se convierta
# en el sitio de producción. También modifica el nombre del
# antiguo  sitio de producción para que se convierta en el de
# construcción.
#
# $Author$
# $Revision$
# $Id$
# $Status
```

```
# Tendrá que cambiar estas variables por las que vaya a utilizar
# su script.
DATA_FILES ="/www/miempresa/public/htdocs/cgi data/*.dat";
TEMP_DIR="/www/miempresa/public.old";
PRODUCTION_DIR="/www/miempresa/public";
STAGE_DIR="/www/miempresa/staging";

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

# Cambie temporalmente el nombre del directorio de producción por
# TEMP_DIR
/bin/mv PRODUCTION_DIR        TEMP__DIR

# Modifique el nombre del sitio de construcción por el directorio
# de producción
/bin/mv STAGE_DIR             PRODUCTION_DIR

# Cambie ahora el nombre del antiguo directorio de producción
# por el de construcción
/bin/mv TEMP_DIR              STAGE_SITE

# Para  estar a salvo tendrá que cambiar los permisos de los
# directorios de producción para que el usuario Apache (httpd) y
# el grupo Apache (httpd) puedan leer todos los ficheros. Si
# utiliza un usuario y un grupo que no sean los predeterminados,
# tendrá que modificar la configuración de este comando.

/bin/chown  -R httpd.httpd    $PRODUCTION_DIR

# Vamos a cambiar los permisos de los archivos para que su
# propietario (en este caso httpd) pueda leer, escribir y
# ejecutarlo, mientras que el grupo (en este caso httpd) pueda
# leer y ejecutarlo.

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

Listado 18.1. Script stage2poduction.sh

Después de ejecutar el script tendrá que efectuar una comprobación rápida para asegurarse de que todo está en orden. Si aparece algún problema, puede cambiar el directorio de producción por otra cosa y copiar el directorio de construcción en el de producción. De esta forma se restaura el estado anterior a las modificaciones.

Cuando se acostumbre a trabajar con este ciclo, encontrará que es muy sencillo desarrollar e integrar la solución a los problemas y asegurarse de que la producción del sitio Web siempre será la correcta. Aparte de mantener un ciclo Web estricto, aún tendrá que hacer unas cuantas cosas más para que su sistema Web funcione a la perfección. Estas tareas entran dentro de la categoría de mantenimiento.

Mantenimiento de su Web

Una vez que se ha implementado el ciclo Web, es muy importante efectuar una serie de tareas de mantenimiento, entre las que tenemos la visualización, registro y copia de seguridad de datos. Como los dos primeros ya los hemos visto en capítulos anteriores, nos centraremos en el tercero. Siempre que sea posible, convendrá que haga dos tipos de copias de seguridad, también conocidas como backup.

Backup online

Este tipo de copia de seguridad es muy útil cuando se presenta una emergencia. Podrá acceder a los datos guardados y restaurar los archivos dañados en cuestión de minutos. Para poder disfrutar de este sistema, tendrá que buscar alguna solución comercial o bien ponerse en contacto con su ISP. Pero si el servidor Web se encuentra instalado en la propia red de la empresa, podrá utilizar el resto de los host para guardar las copias de seguridad. En la mayoría de sistemas UNIX se puede ejecutar un programa llamado rdist para crear un mirror de los directorios de los sitios Web en el resto de los host UNIX del sistema.

También puede ser una buena idea guardar una versión comprimida de los datos Web en el propio servidor. En los sistemas UNIX puede configurar la utilidad cron para que cree un archivo comprimido tar que se encargue de hacer una copia de los datos Web periódicamente. Por ejemplo:

```
# para los sistemas Unix -V, el periodo semanal está comprendido
# entre el 0-6, donde 0=Sunday (domingo).
# Para los sistemas BSD el rango utilizado va del 1-7, donde 1=Monday
# (lunes).
# Este ejemplo se ha pensado para un sistema Linux (rango 0-6)
30 2 * * 0,1, 3, 5,      root /bin/tar czf /backup/M-W-F-Sun.tgz
/www/*
30 2 * * 2, 4, 6   root /bin/tar czf /backup/T-TH-Sat.tgz  /www/*
```

Estas dos entradas cron se guardan en /etc/crontab. Todos los Lunes, Miércoles, Viernes y Domingos se ejecutará el primer cron a las 2.30 a.m. para crear una copia de seguridad de todo lo que se encuentra dentro de /www y se guardará en un archivo comprimido, backup/M-W-F-Sun.tgz. Del mismo modo, los Martes, Jueves y Sábados, a la misma hora, se ejecutará el segundo cron que crea el archivo backup/T-TH-Sat.tgz en el mismo directorio. Con este sistema se asegura de que tiene por lo menos dos copias de seguridad (pertenecientes a días alternos) de los datos de su sitio Web.

Backup offline

También tendrá que efectuar copias de seguridad en medios extraíbles y guardarlos en un lugar seguro. Restaurar este tipo de copias de seguridad lleva su tiempo. Puede utilizar unidades de cinta, discos duros extraíbles (como discos Jaz). Yo prefiero utilizar cintas de 8mm porque me permiten guardar hasta 8GB de información. Además, llevan en el mercado bastante más tiempo que los nuevos sistemas de almacenamiento.

Según crezca el contenido de su sitio Web, notará que el espacio se llena con sorprendente rapidez. A menudo se debe a que hay archivos que no se utilizan nunca y que no se eliminan por miedo a estropear algún enlace. Si es lo que pasa con su sitio Web y trabaja con plataformas UNIX, es posible que le convenga emplear una utilidad capaz de localizar los archivos a los que hace bastante tiempo que no se accede. Por ejemplo:

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

mostrará todos los archivos que se encuentran dentro de los directorios /www y que terminan con la extensión .bak a los cuales no se ha accedido en los últimos diez días. Si quiere eliminarlos, utilice el siguiente comando:

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

Si le sirve de utilidad, puede crearse una entrada cron que ejecute periódicamente esta entrada.

Estándares

Al trabajar con un ciclo Web tendrá que colaborar con varias personas. Pero la mera creación de un ciclo Web no le garantiza una producción de alta calidad en su sitio Web. Para ello, el contenido tendrá que ser de gran calidad y hay una serie de guías que debería seguir, independientemente del contenido de su sitio Web. La finalidad es constituir unas normas estándar.

Cada sitio Web ofrece un contenido único que resulta atractivo a sus posibles visitantes. Cualquier contenido Web puede catalogarse como dinámico o estático. El contenido estático se suele crear con los archivos HTML, el dinámico con las salidas de los script CGI o de otras aplicaciones. La mayoría de los sitios Web utilizan una mezcla de ambos tipos de contenido para mostrar su información al público. Por lo tanto, conviene implantar una serie de estándares para el desarrollo de su contenido dinámico y estático.

Política de desarrollo de documentos HTML

Aunque tenga varias formas de suministrar el contenido estático de su sitio Web a sus clientes (por ejemplo, a través de archivos PDF o en texto plano), la mayoría de los sitios Web utilizan documentos HTML como formato a través del cual mostrar la información. Para ayudar a sus desarrolladores HTML, conviene que cree una política de desarrollo HTML. A continuación, le muestro una serie de puntos que puede adaptar a las necesidades de su organización para implementar los cambios necesarios.

Utilice siempre etiquetas HTML estándar

Los desarrolladores Web deberían utilizar HTML estándar. Las etiquetas HTML dependientes de los exploradores pueden hacer que el aspecto de una página Web en un explorador determinado sea importante y terrible en otro.

Por ejemplo, ahora le mostramos el esquema de un documento HTML que cumple los requisitos mínimos de estandarización de los documentos HTML.

```
<HTML>
<HEAD>
<TITLE> Título del documento </TITLE>
</HEAD>
<BODY>
Aquí va el cuerpo del documento
</BODY>
</HTML>
```

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

Guarde las imágenes con sus documentos

Las imágenes que aparecen en los documentos Web deberían guardarse en un subdirectorio del directorio de documentos. De todas formas, hay una excepción a esta regla: si va a tener que volver a utilizar estas imágenes, conviene que considere guardarlas en un directorio central para imágenes. Por ejemplo, en este directorio se podrían colocar los gráficos que representarán los botones de la barra de navegación de unos documentos Web. La barra de navegación se puede utilizar con un gran número de documentos, por lo que puede ser conveniente guardarlos en un directorio central en vez de mantener una copia en el directorio de cada documento. Además de ahorrar espacio de disco se obtiene un mayor control sobre los elementos de su página Web. Quitando la excepción que acabamos de nombrar, el resto de imágenes se guardarán junto con los documentos en los cuales aparecen. La referencia a

estas fuentes tiene que ser relativa, de modo que cuando se mueva el directorio de un documento (con todos sus subdirectorios) de una ubicación a otra, la representación en pantalla de la imagen no sufra ninguna modificación. En el ejemplo que veremos a continuación, le mostraré como crear un documento HTML que tiene varios vínculos gráficos:

Supongamos que queremos publicar dos documentos HTML (mydoc1.html y mydoc2.html) que contienen tres imágenes (image1.gif, image2.gif y image3.gif). Puede comenzar por crear un subdirectorio dentro del directorio del documento o en cualquier otra ubicación lógica. Supongamos que lo hemos creado dentro del directorio raíz de documentos (/www/mycompany/htdocs) y que lo hemos bautizado mydir.

Ahora creamos un subdirectorio llamado imagen que se encuentra dentro de mydir. En él guardamos todas las imágenes. Edite los documentos HTML para que todos los vínculos utilicen el siguiente atributo SRC:

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

Un ejemplo de uno de los vínculos gráficos en los que se utiliza image3 podría ser éste:

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

En los atributos anteriores no aparece ninguna información relacionada con la ruta. Si los documentos se colocasen en otro directorio, pero conservasen mydir, las imágenes no sufrirían ningún cambio. Pero si la línea anterior fuese como ésta:

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

o

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

entonces habría que corregir los vínculos al modificar la posición de los documentos. Muchos sitios Web conservan las imágenes en un directorio central (algo parecido a imagen), que se encuentra dentro del directorio de documentos, y establecen los vínculos con las imágenes por medio de etiquetas IMG:

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

Está bien, pero tiene un par de problemas:

- Cuando quiera eliminar un documento HTML, tendrá que asegurarse de que borra la imagen adecuada de las que se encuentran en el directorio central imagen.
- Si no lo hace correctamente, estará desperdiciando una gran cantidad de disco duro.

Por lo tanto, no es muy buena idea guardar las imágenes en un directorio central. Guárdelas en un subdirectorio junto con los vínculos.

Muestre claramente los avisos del Copyright en todos los documentos

Cada documento ha de contener un mensaje de Copyright en el que se comenten el nombre del propietario del documento y de todas las imágenes. Debería aparecer uno de estos avisos en todas sus páginas Web. Para facilitar la actualización de estos mensajes, se puede utilizar la directriz SSI:

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

Ahora todo lo que tiene que hacer es crear una página HTML llamada copyright.html que colocará en el directorio raíz de documentos. Como el contenido de este documento se encuentra en una página HTML que se mostrará por medio de una llamada SSI, no se puede utilizar ninguna etiqueta <HEAD>, <HTML>, <TITE> o <BODY>. Gracias a este sistema se simplificará considerablemente la actualización de los mensajes del Copyright (por ejemplo, si es que hay que actualizar la fecha).

Política de desarrollo para aplicaciones dinámicas

Otro tipo de contenido es el dinámico. Lo suelen producir los script CGI u otras aplicaciones que implementan tanto CGI como interfaces del lado del servidor. Como los script CGI y las aplicaciones suelen tener una vida muy reducida, muchos desarrolladores CGI no dedican el suficiente tiempo para producir aplicaciones de gran calidad.

Si tiene pensado trabajar con script y aplicaciones basadas en FastCGI o con mod_Perl, conviene que se desarrollen de la forma correcta. Es posible que quiera considerar la siguiente política a la hora de implementar aplicaciones o script que trabajen con el contenido dinámico de su red.

Utilice siempre el control de versión

Los desarrolladores CGI han de utilizar un control de versión que le permita volver a una versión antigua de la aplicación si se detecta un fallo en una de las modernas. En la mayoría de los sistemas UNIX se puede utilizar el software CVS (Concurrent Versions System) para implementar un entono de control de versiones. Encontrará una versión de este software en:

```
ftp://prep.ai.mit.edu
```

No utilice nombres de rutas absolutas en aplicaciones ni en script CGI

Si no se utilizan nombres de rutas absolutas en aplicaciones ni en script CGI, se asegurará de que los script se puedan utilizar en cualquier sitio Web sin tener que efectuar ninguna modificación. Si se da el caso de que tenga que usarlos, al script habrá que suministrarle un archivo de configuración. De este modo, para actualizar las rutas bastará con actuar sobre los archivos de configuración.

Suministre documentación tanto a nivel de usuario como de código

Conviene que todo lo que distribuya cuente con una buena documentación para ayudar a que los futuros desarrolladores puedan actualizar los script sin tener que invertir una gran cantidad de tiempo en averiguar cuál es el funcionamiento.

Evite utilizar etiquetas HTML en los script o en las aplicaciones

La salida de los script se ha de basar en una plantilla. En otras palabras, un script CGI lee la información de una plantilla de salida y sustituye los campos allí existentes por datos dinámicos (se pueden utilizar etiquetas personalizadas). De esta forma, facilitará enormemente a los desarrolladores HTML la actualización de la página de salida, porque el HTML no se encuentra dentro del script. De hecho, los script CGI suelen contener la menor cantidad de HTML posible.

No se fíe de los datos que introduce el usuario

Para reducir riesgos, asegúrese de que se revisan los datos introducidos por los usuarios antes de procesarlos.

Evite variables globales en los script CGI basados en Perl

Cuando desarrolle CGI en Perl, debería evitar el uso de variables globales. Limitar el alcance de una variable es una forma de eliminar los comportamientos no predecibles. Los programadores de Perl suelen trabajar con declaraciones de variables como la siguiente:

```
my $variable;
```

en vez de:

```
local $variable;
```

porque la última crea una variable global, mientras que la primera crea una variable local.

Utilice una interfaz agradable

Utilizar HTML estándar en vez de aplicaciones/script CGI puede hacer que su sitio Web sea mejor que muchos de los ya publicados en la Red. De todas formas, hay otro aspecto relacionado con el diseño de las páginas Web que conviene tener en cuenta: la interfaz del usuario.

Piense en un sitio Web como en una aplicación interactiva con una GUI, que es lo que ve el explorador Web. El aspecto de la interfaz GUI tiene que resultar agradable a todos los usuarios que visitan sus paginas Web. Con una interfaz agradable al usuario, los visitantes se sentirán cómodos navegando por sus página Web.

En esta sección veremos las claves en el desarrollo de GUI agradable para el usuario. Además de esmerarse en conseguir una buena apariencia, tendrá que tener cuidado con los vínculos que se dirijan a archivos que se han eliminado. Use los archivos de registro de errores de su servidor Web para detectar este tipo de problemas. Proporcione una vía para que los usuarios puedan ponerse en contacto con usted. La mayoría de los sitios Web utilizan un formulario basado en un script CGI. Puede desarrollar uno que se ajuste a sus necesidades. Recopilar sugerencias puede ser una forma de saber qué opinan los usuarios de su sitio Web.

Facilite la navegación por su sitio Web

Los usuarios tienen que poder desplazarse de una página a otra sin tirarse de los pelos por pura desesperación. Tienen que encontrar botones o barras

de menú que les permitan saltar hacia delante o hacia atrás para alcanzar la información deseada.

Los diseñadores de muchas páginas Web aseguran que los exploradores Web más famosos incluyen un botón Adelante y otro Atrás, por lo que incluirlos en la propia página Web puede resultar redundante. No es cierto. Imagínese que un usuario llega a una de sus páginas Web (que no sea la principal) procedente de un motor de búsqueda. Está buscando unas cuantas palabras. Tiene interés en conocer el contenido de su sitio Web, por lo que quiere saltar al principio del documento, pero se encuentra con que no hay ninguna forma de hacerlo, porque si pulsa el botón Atrás que se encuentra en el explorador Web saltará al motor de búsqueda.

Pero si la página incluyese un vínculo (o botón) que lo enlazase con la página anterior, el usuario podría desplazarse hasta ella sin ninguna dificultad. Los diseñadores de páginas Web a los que no les gusta incluir botones adicionales aseguran que basta con que el usuario manipule el URL para acceder a la página principal. Pero para admitir este argumento hay que aceptar que el URL que conduce a la página principal es muy sencillo de averiguar, cosa que no suele ocurrir.

Lo mejor es incluir una barra de menú gracias a la cual el usuario pueda saltar hacia delante y atrás. Incluso le permitirá saltar a una ubicación determinada o a la página principal. En la figura 18.3 tiene un ejemplo de una de estas páginas.

Crear un diseño aparente

Piense en los sitios Web como en una presentación interactiva llena de colorido, activa las 24 horas del día. Si el aspecto de esta presentación no es el correcto, sus visitantes se irán a otra parte. Considere los siguientes puntos a la hora de desarrollar su sitio Web.

Colores apropiados

Asegúrese de no saturar el color de su sitio Web. El uso de colores extremos da un aspecto poco profesional. Sea consciente y utilice una paleta adecuada. Por ejemplo, si el sitio Web trata de juguetes, sí que podrá ser muy colorida, pero si trata de los resultados obtenidos al medir la velocidad de los procesadores del mercado, no necesitará ni demasiados colores ni un fondo chillón.

Tamaño de texto apropiado

Trate de utilizar una fuente de aspecto normal. El uso de fuentes especiales a través de las etiquetas <FONTFACE="myFuente"> puede hacer que el

aspecto de las páginas sea espléndido cuando se utiliza un explorador Web determinado (que posea esa fuente en particular), pero alguien que utilice otro servidor se encontrará con algo muy distinto, costándole incluso leer el contenido de la frase. Tenga cuidado con el tamaño del texto; no lo haga ni demasiado grande ni demasiado pequeño. Recuerde que si sus visitantes no pueden leer el contenido de su página Web no podrán determinar qué trata de comunicárseles.

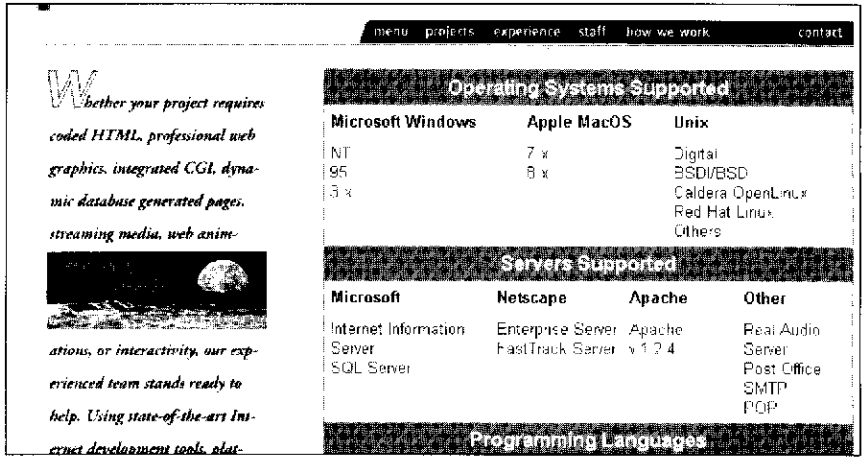


Figura 18.3. Ejemplo de una página en la que se utiliza un menú de navegación

Reduzca el uso de imágenes

Es muy sencillo caer en la tentación de utilizar un número innecesario de imágenes en las páginas Web. Las imágenes hacen la carga de las páginas Web sea lenta. Recuerde que no todo el mundo cuenta con una conexión RDSI. La mayoría tiene un módem a 28.8K. Si la carga de una página le lleva demasiado tiempo, es muy posible que detengan el proceso y se vayan a otro sitio Web.

Tenga cuidado con las animaciones. Incluso las mejores animaciones terminan aburriendo después de un par de visitas, por eso no conviene saturar las páginas con ellas.

Elimine los mensajes de errores criptográficos

Configure la directriz ErrorDocument de Apache para que los usuarios no reciban mensajes de error que no puedan comprender. Por ejemplo, cuando no se encuentra un URL en un servidor Web, se mostrará el mensaje de error que aparece en la figura 18.4.

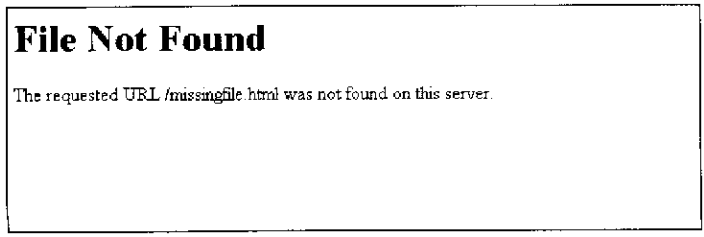


Figura 18.4. No se encuentra el documento solicitado

Para hacer que este mensaje de error sea menos frío, se puede añadir una directriz ErrorDocment como ésta:

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

En el archivo sorry.html se puede escribir un mensaje parecido al que se muestra en la figura 18.5.

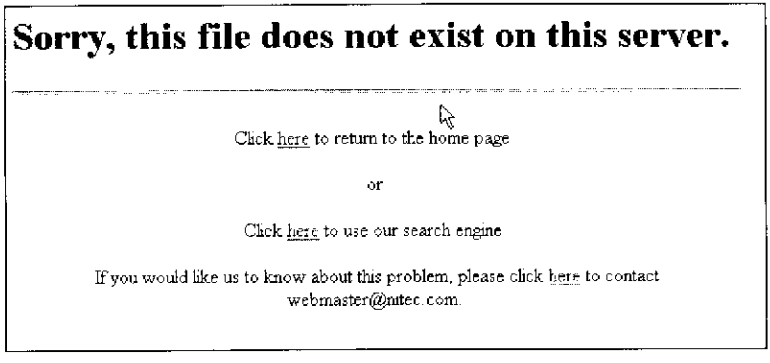


Figura 18.5. Un mensaje de error menos frío que el que genera el sistema

Pruebe la interfaz GUI de su Web

Una de las mejores formas de probar su interfaz Web es utilizar un sistema que se comporte como el sistema de un usuario de Internet o como el ordenador de un posible cliente. Si piensa que todos sus clientes tienen ordenadores muy potentes con conexiones rápidas, es posible que no tenga que preocuparse del uso de gráficos, de aplicaciones como applet en Java o de animaciones.

En la mayoría de los casos no conocerá la potencia de los ordenadores de sus clientes ni la configuración de sus redes, por eso debería probar el sistema con una configuración media. Use un Pentium que no sea muy rápido, con 16MB de RAM y un módem de 28.8K. Trabaje con resoluciones de 640x480

ó 800x600 píxeles. Si sus visitantes utilizan sistemas Web basados en la televisión, utilice una resolución de 550x400.

Si le gusta el resultado obtenido, es muy posible que a otros usuarios también le guste. Pero tenga en cuenta que, si no le gusta, es muy posible que a ellos tampoco.

Si lo prefiere, puede pedir a terceras partes que prueben su sitio Web. Por ejemplo, una compañía llamada AtWeb Inc. suministra un servicio de ajuste gratuito a través de la dirección:

```
www.websitegarage.com/
```

Su aplicación examina el tiempo de carga de cualquier página, la calidad del HTML, los enlaces inservibles, errores, la calidad de diseño HTML y la popularidad de los vínculos. Para utilizarla, todo lo que tiene que hacer es acercarse al sitio Web anterior e introducir la dirección de su sitio Web.

Por ejemplo, si quiere probar la configuración de www.nitec.com, tendrá que introducir esta dirección en la casilla correspondiente de la ventana que aparece en la figura 18.6.

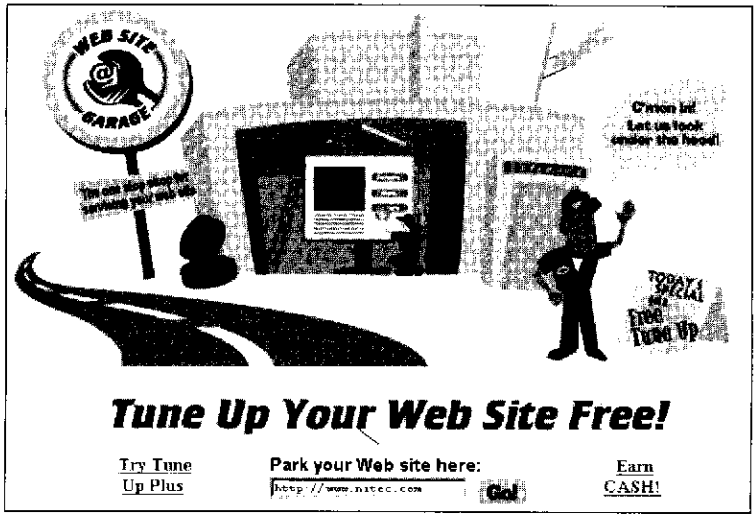


Figura 18.6. Petición de un informe sobre los ajustes

Después de hacer clic en el icono Go, hay que esperar unos segundos y la aplicación genera un informe como el que aparece en la figura 18.7.

El informe generado mide si el sitio Web suministrado cumple con los estándares de AtWeb. Esta versión gratuita está limitada, pero la empresa ofrece paquetes comerciales para ajustar todos los aspectos de su sitio Web.

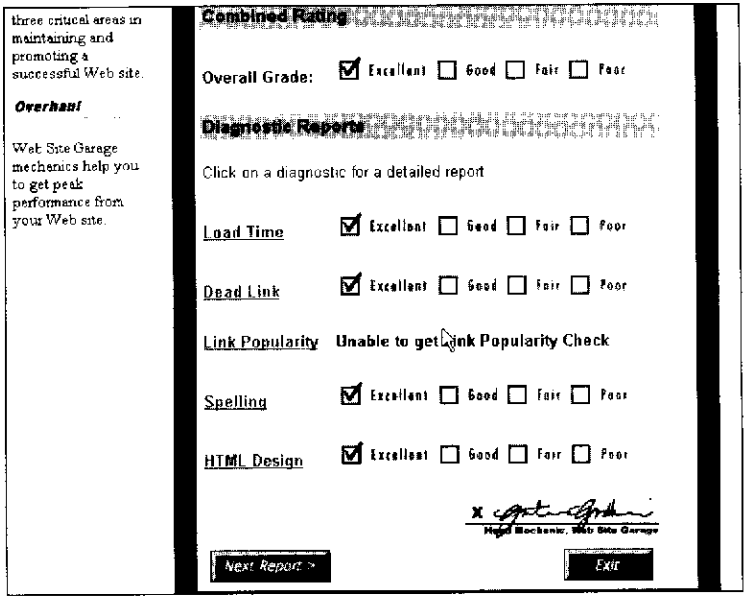


Figura 18.7. Informe sobre www.nitec.com

Punteros para promocionar su sitio Web

¿Qué más da que un sitio Web sea perfecto si no lo conoce nadie? Conviene que se plantee promocionar su sitio Web en la Red. También puede ponerse en contacto con agencias de publicidad para que le ayuden en su propósito. De todas formas, conviene que sepa que los anuncios en la Red son caros. Si su presupuesto es algo limitado, es posible que prefiera efectuar ciertas tareas de promoción por su cuenta.

Antes de hacer nada por promocionar su sitio Web, conviene que se formule la siguiente pregunta: "¿Cómo puedo encontrar información en la Red?" La respuesta es muy sencilla: a través de los motores de búsqueda. Su compañía, ¿aparece en estos motores? Si no es así, éste es el primer paso que tiene que dar para promocionar su sitio Web.

La mayoría de los motores de búsqueda le permiten enviar su URL a la base de datos de sus robots para que visiten su sitio Web en un futuro. Haga una lista con los motores de búsqueda que le parezcan más importantes y envíeles el URL de su sitio Web. Aunque este proceso le puede llevar varios días (o incluso semanas), puede añadir META información para que el contenido de su URL aparezca en una buena posición en la lista de sitios que se le presentan al cliente que efectúa una búsqueda.

Por ejemplo, añadiendo:

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

Para aumentar el tráfico, puede participar en intercambios en vínculos de intercambio, como en www.linkexchange.com. Con este tipo de sitios Web tendrá que utilizar una serie de etiquetas HTML especiales en sus páginas Web. Gracias a ellas aparecerán anuncios en sus páginas Web. Para compensar, los anuncios de su organización aparecerán en las páginas Web de otras empresas. Este tipo de promoción es muy popular entre las pequeñas empresas o incluso a nivel personal.

Tanto si compra un espacio de anuncio en uno de los sitios Web más importantes de la Red (como Yahoo!, AltaVista o Netscape), como si utiliza el sistema de intercambio de enlaces, conviene que revise periódicamente el estado en que se encuentra su empresa en las listas que muestran dichos buscadores a los clientes que hacen alguna consulta.

19 Apache para las redes Web

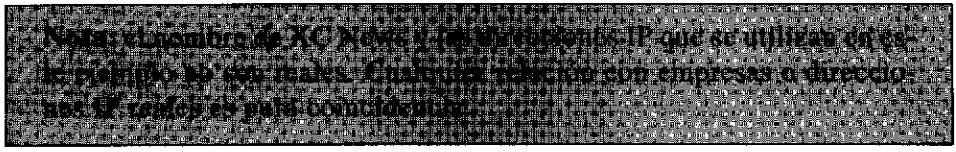
Hasta ahora, en este libro hemos visto las propiedades que puede aprovechar de Apache. Si tiene una máquina en la que haya instalado Apache, seguro entonces que ya se ha puesto a juguetear con todas las cosas nuevas que ha ido aprendiendo.

Si sus necesidades se encuentran en una escala menor, puede instalar el servidor Apache en un ordenador y configurar unos cuantos sitios Web. Pero, ¿qué pasa si sus necesidades se encuentran en una escala mayor, por ejemplo, con una red Web? Si su organización necesita una de estas redes de trabajo, sepa que hay muchos sistemas que le pueden ayudar a crear una red de este tipo. En este capítulo veremos cómo puede hacerla.

El ejemplo que veremos en estas páginas se ha sacado del mundo real, utilizando una red Web basada en Apache.

¿Qué es una red Web?

Antes de nada, tenemos que comprender qué es una red Web. Supongamos que tenemos una empresa llamada XC News que quiere crear una red Web escalable en la que se utilice un sistema de reparto de cargas de trabajo para distribuir las peticiones entre dos servidores Web.



Vamos a ver las necesidades del proyecto:

- **Propósito.** XC News tiene cientos que clientes que quieren que aloje sus sitios Web en su sistema y que les permita acceder a sus servicios (desarrollo de aplicaciones Web, etc.) Después de utilizar un servidor Windows NT independiente, XC News necesita una solución mejor porque su proyección de futuro muestra que el crecimiento de servicios que ofrecen se va a disparar en breve. Por esta razón la compañía quiere desarrollar una red Web que se base en varios servidores Web gracias a los cuales se pueda suministrar a sus clientes un entorno eficaz y escalable. En un principio, la red constaba de dos servidores Web y un monitor en el servidor del sistema, pero el diseño y la implementación de la red tiene que ser lo suficientemente flexible como para añadir más servidores cuando sea necesario.
- **Plataforma del servidor.** XC News ha visto la importancia que tiene una red distribuida y cómo se puede utilizar su arquitectura para obtener una solución escalable. Para disponer de la capacidad que le permita agregar nuevos sistemas rápidamente y satisfacer así las necesidades de la red, la organización quiere utilizar PC basados en procesadores Intel y sistemas operativos que les permitan efectuar funciones administrativas a distancia. Su experiencia con Windows NT les ha convencido que esta plataforma es lo suficientemente estable como para efectuar tareas de administración a través de Internet y prácticamente no hace falta instalar, actualizar o eliminar ningún software adicional para servidores a través del acceso remoto a Internet.
- **Localización de la red Web.** Para reducir el incremento de costes, XC News quiere desarrollar su red Web en el sistema de un ISP. De esta forma podrá aprovecharse de las ventajas de las facilidades de administración que cumplen los estándares industriales y darle acceso instantáneo a una red que cuenta con un gran ancho de banda.
- **Sistema del servidor DNS.** Como XC News quiere tener un DNS para todos los dominios de los clientes y para los suyos propios, necesitará un servidor DNS. También quiere visualizar los servidores Web a través de un sistema en el que se pueda programar una serie de aplicaciones

que comprueben la disponibilidad de estos servidores. Cuando falla uno de ellos, este sistema mandará mensajes a través del correo electrónico o bien de los buscapersonas, a los responsables del mantenimiento de dicho ordenador.

- **Sistema de seguridad del servidor.** XC News querría proporcionar unos servicios de transacciones seguras (basados en SSL). Ni XC News ni sus clientes tienen el más mínimo interés en comprar un certificado para cada sitio. Además, la compañía quiere restringir el acceso a la zona de seguridad de tal forma que los clientes no puedan crear sus propias páginas de seguridad.

- **Diseño de red.** La red se diseñará de tal forma que la carga se repartirá entre varios servidores. El diseño y la implementación tienen que contar con los fallos del servidor Web. Tiene que ser muy sencillo redirigir, manual o automáticamente, todas las peticiones al resto de servidores Web del sistema. Gran parte de las aplicaciones CGI de XC News escriben datos en archivos. Estos ficheros se tienen que sincronizar entre todos los servidores Web de tal forma que cada sitio pueda ofrecer la última versión de los datos a sus clientes.

- **Requisitos de backup de la red.** El proveedor de servicios de Internet (ISP) que escoja XC News utilizará una arquitectura Legato NetWroker, solución que permite hacer copias de seguridad de una red. Este ISP necesita que XC News ejecute un cliente Legato en su sistema para que le envíe los archivos de los que hay que hacer copia de seguridad a su servidor Networker a través del protocolo TCP/IP. Tenga en cuenta que esta solución es específica de un fabricante y por eso se puede encontrar con bastantes ISP que no trabajen con ella. Por lo tanto, no vamos a entrar en detalles respecto a este sistema de copias de seguridad. De todas formas, sí que me gustaría recordar que las copias de seguridad son uno de los aspectos más importantes de una red.

- **Requisitos del sitio Web del cliente.** Cada cliente de XC News tiene un dominio único para el cual XC News proporciona un DNS. El dominio del cliente comenzará por www.cliente-dominio.com, que será el URL del host del sitio Web. Cada cliente tendrá acceso a su directorio de documentos a través de FTP. Un cliente puede solicitar tener más de una cuanta POP3 de correo o solicitar que el correo enviado a su dominio se redirija a un servidor de correo externo.
  Cada cliente del sitio Web podrá trabajar con aplicaciones CGI de XC News. Los sitios Web de esos clientes no tendrán un directorio CGI-BIN.

De todas formas, XC News prevé que esta política cambiará en un futuro, por lo que le gustaría que el diseño de la red permita implementar el acceso local al directorio CGI, aunque de momento no se vaya a usar.

Estas son las especificaciones de alto nivel. Siga la lectura del presente capítulo para asegurarse de que comprende las especificaciones de todos los requisitos.

Los requisitos

Antes de diseñar un proyecto, se han de comprender los requisitos. Vamos a ver los requisitos desde el punto de vista de un diseñador de redes. Parece que XC News quiere crear una red con los servidores, donde cada servidor Web albergará todos los host de los sitios Web de los clientes y los de XC News. Esta red funcionará como un URL que redirigirá automáticamente las peticiones a una máquina diferente que ejecute el mismo sitio Web. En la figura 19.1 se puede ver cómo funciona.

Como se puede observar, una petición dirigida a www.xcnews.com la puede atender tanto el servidor #1 como el #2. Si falla uno de los servidores, se enviará todo el tráfico al otro.

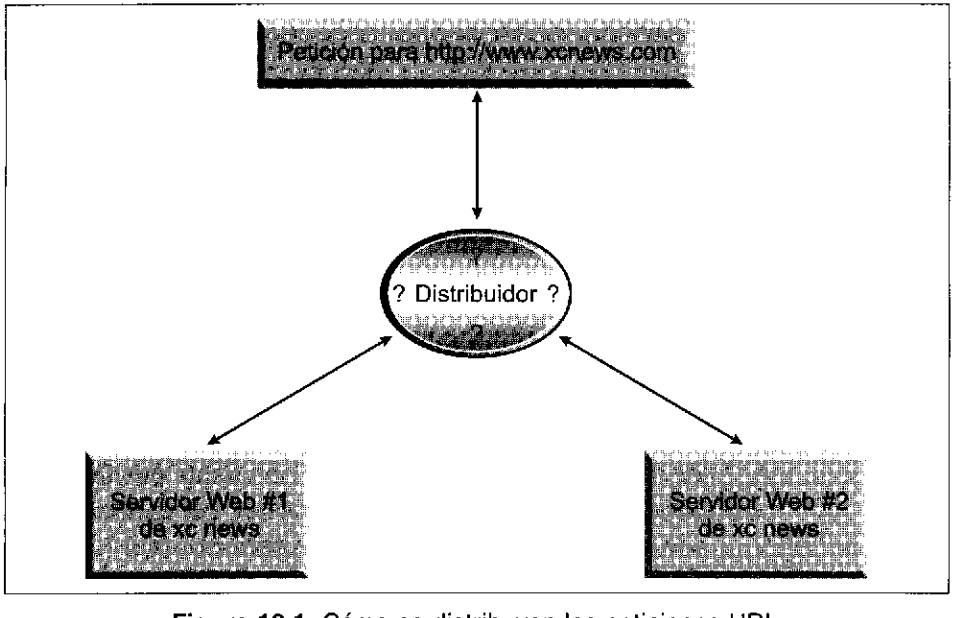


Figura 19.1. Cómo se distribuyen las peticiones URL

Entre los requisitos nos encontramos con que tiene que haber una única zona de construcción para los archivos FTP de los clientes. Esta zona tendrá que estar repartida entre los distintos servidores Web, de modo que cada uno de ellos tenga todos los archivos necesarios para atender las peticiones URL. ¿Dónde se encontrará esta zona de construcción? Antes de responder a esta pregunta, hay que hacer una aclaración sobre los sistemas de los servidores Web. Si lee atentamente las especificaciones, verá que la compañía XC News quiere considerar a los servidores Web como sistemas de producción. Como generalmente es conveniente reducir al máximo las interacciones de estos sistemas de producción con los clientes, la zona de construcción no podrá encontrarse en ninguno de ellos. Tiene que estar en un tercer sistema: servidor DNS y de visualización. Ahora, todo lo que tiene que hacer es crear una zona de construcción en el servidor DNS y distribuir dichos archivos entre los servidores de producción Web.

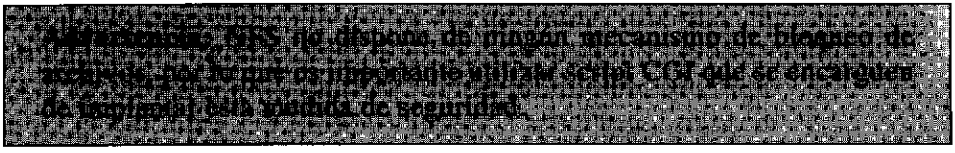
¿Cómo se pueden repartir los archivos entre los servidores? Aquí tiene un par de soluciones:

- Utilice el sistema NFS para compartir el área de construcción con el resto de los servidores Web.
- Utilice un software de distribución remota llamada rdist para distribuir los archivos hasta los servidores Web.

Si utiliza NFS con todos los sitios Web y el número de ficheros es muy elevado, se puede encontrar con un cuello de botella, porque NFS no es tan bueno como un sistema de archivos locales. Por eso rdist parece ser una buena opción. Utilizando cron, una de las utilidades con las que cuentan los sistemas UNIX, en combinación con rdist, podrá programar la distribución de los ficheros.

De todas formas, aún queda un punto relacionado con la sincronización de la escritura de datos por parte de los script CGI. Si se utiliza rdist para sincronizar los archivos de datos CGI, habrá entonces periodos en los que no haya sincronización entre los servidores. Además, existirán varias copias del mismo archivo, lo cual puede llegar a provocar todo tipo de problemas. Así que parece que habrá que utilizar otro método para sincronizar los archivos de datos CGI.

Las buenas noticias son que los archivos de datos CGI se escriben en el mismo directorio, por lo que todo lo que tiene que hacer es montar el directorio como punto de exportación NFS para los servidores Web. El resultado será entonces que todos los servidores Web podrán leer y escribir los mismos archivos.



De los requisitos se ve que el servidor seguro tiene que ejecutarse en un servidor DNS. En la tabla 19.1 tiene los servicios que se desarrollarán en cada uno de los sistemas de la red.

Tabla 19.1. Servicios de cada sistema

Sistema	Servicios
Servidor DNS	Servicios de Nombres de Dominio. Servicio de visualización. Servidor NFS para los directorios de archivos de datos CGI. Servidor FTP para la zona de construcción. Servidor rdist para distribuir los archivos que se encuentran en la zona de construcción. Servidor de seguridad.
Servidor Web #1	Servidor Apache. Cliente NFS para los directorios de archivos de datos CGI. Cliente rdist.
Servidor Web #2	Servidor Apache. Cliente NFS para los directorios de archivos de datos CGI. Cliente rdist.

Obsérvese que en vez de desarrollar una solución local, se utiliza un paquete de visualización basado en Perl, llamado Spong, gracias al cual se cubren los requisitos solicitados. De todas formas, debido a la simplicidad de su instalación y proceso de configuración, no lo veremos más.

Ahora que se han identificado los servicios de cada máquina, vamos a ver el tipo de tráfico que hay en la red. Tenemos los tres tipos que se describen a continuación:

- **Tráfico de Internet.** Cada servidor se comunicará con los clientes a través de Internet.
- **Tráfico de sincronización de archivos rdist.** Este tráfico solamente aparecerá en las tres máquinas implicadas en la distribución de archivos y no podrá viajar a host externos.

- **Tráfico NFS.** Este tráfico únicamente aparecerá en las tres máquinas implicadas. El servidor DNS actuará como servidor NFS, que exportará ciertos directorios a los otros dos servidores Web. El tráfico NFS no podrá viajar a host externos.

Ahora que se han identificado los servicios y requisitos relacionados con el tráfico, ha llegado el momento de diseñar la red.

Diseño de la red Web

El primer paso que hay que dar para diseñar una red Web es esbozar el aspecto global de la solución. En la figura 19.2 se puede ver este diagrama. En él se aprecia que los servidores Web están conectados a la red del ISP a través de un distribuidor (HUB) Ethernet. Todo el tráfico entrante y saliente de Internet pasa a través del HUB y de aquí va a parar a los sistemas. Esta red se encargará de hacer el trabajo, pero la verdad es que no es la mejor idea. ¿Por qué? Si se fija atentamente, la red tiene un HUB Ethernet por medio del cual se conectan las tres máquinas entre sí a través de Internet. Es decir, que todo el tráfico NFS, rdist y el de Internet tendrá que pasar a través del mismo HUB Ethernet. De esta forma se consigue que Ethernet esté saturado de trabajo, por lo que la solución tampoco parece ser la ideal.

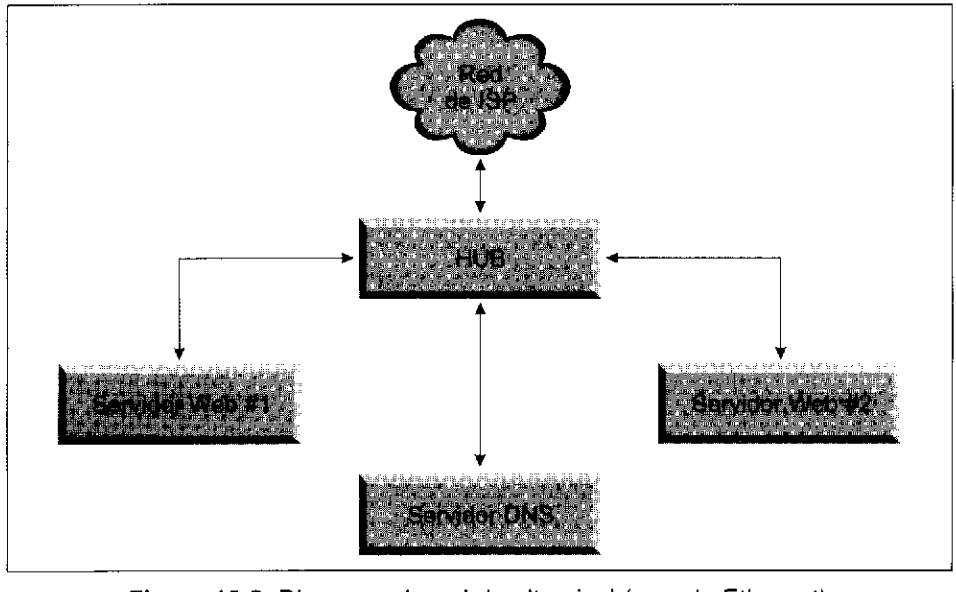


Figura 19.2. Diagrama de red de alto nivel (un solo Ethernet)

Para solucionar el problema, puede modificar el diseño de la red para darle la configuración que aparece en la figura 19.3. En este caso se utilizan dos HUB Ethernet, uno conectado a la red del ISP y el otro que conecta a los servidores entre sí. ¿Qué ventajas aporta esta configuración? Como mantiene el tráfico NFS y rdist en los servidores Ehternet, el HUB conectado a la red del ISP únicamente se puede utilizar con los servicios de Internet. En otras palabras, uno de los Ethernet se podrá utilizar como red de área local (LAN), mientras que el otro se usa para acceder a Internet.

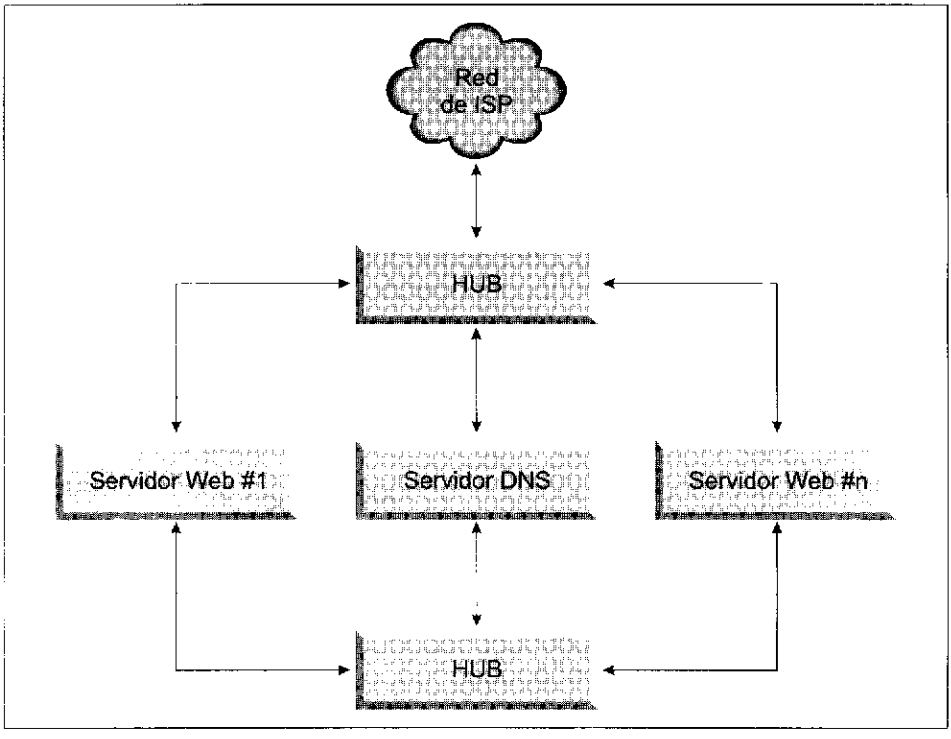


Figura 19.3. Diagrama de red de alto nivel (varios Ethernet)

En la figura 19.4 se puede ver un diagrama que representa el tráfico de una red de alto nivel. Esta es la mejor solución porque el tráfico NFS y rdist únicamente viaja a través de la LAN, dejando el otro HUB libre para la conexión a Internet. Además, se añade una medida más de seguridad a la red. Los administradores de red piensan que NFS no es tan seguro como debería ser. Por eso, si utilizo un único distribuidor Ethernet para exportar los directorios a los servidores Web, hay una posibilidad de que el distribuidor Ethernet se convierta en un objetivo de los ataques de los piratas informáticos.

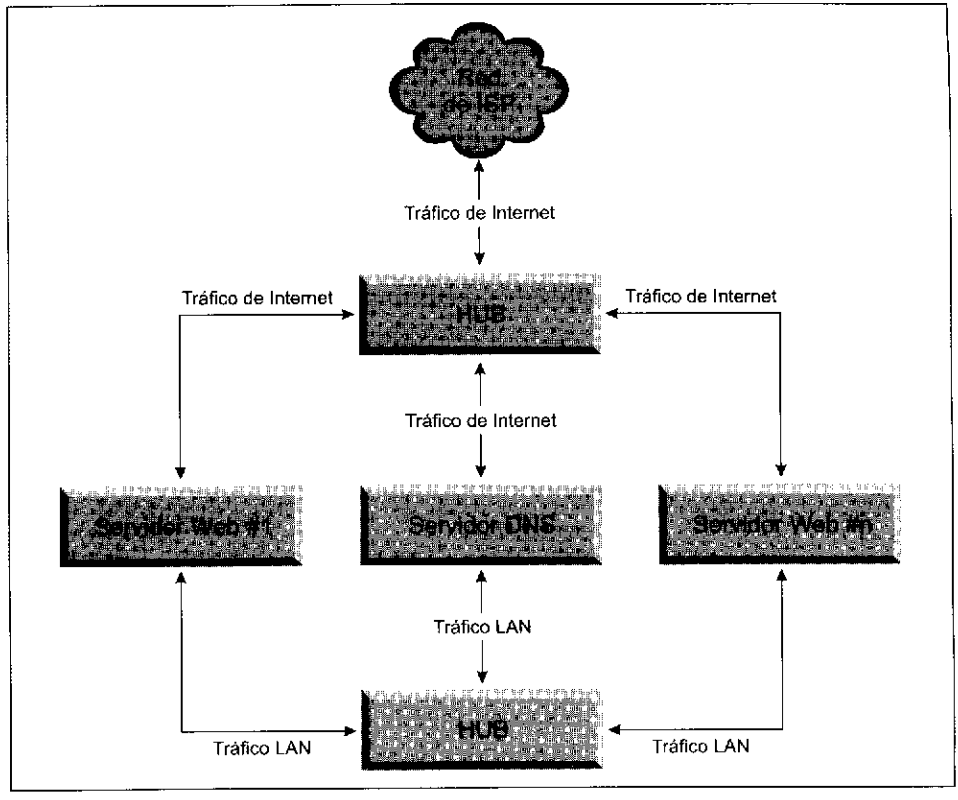


Figura 19.4. Diagrama del tráfico de una red de alto nivel

A muchos sistemas se les ha atacado a través de los agujeros de seguridad NFS, por lo que conviene evitar que se pueda acceder a los directorios NFS por Internet. Para asegurarse de que ninguno de los tres sistemas pueda acceder a la red LAN, compruebe que no hay ninguna dirección directa entre los dos distribuidores. De esta forma evitará que cualquiera que acceda al sistema a través de Internet entre en la red LAN. En la figura 19.5 tiene la asignación de los nodos de ambas redes. Como se puede ver, LAN usa una dirección de red, 192.168.1.60, a la que no se puede acceder a través de Internet.

Nota: También puede utilizar otras direcciones IP a las que no se puede acceder desde una red de Clase A, como 10.0.0.0.

Las direcciones IP de Internet dependen del ISP, ya que sí que se tiene que poder acceder a ellas a través de Internet.

Ahora que ya tiene los conocimientos suficientes sobre la red, ha llegado el momento de meterse con las listas de hardware y software.

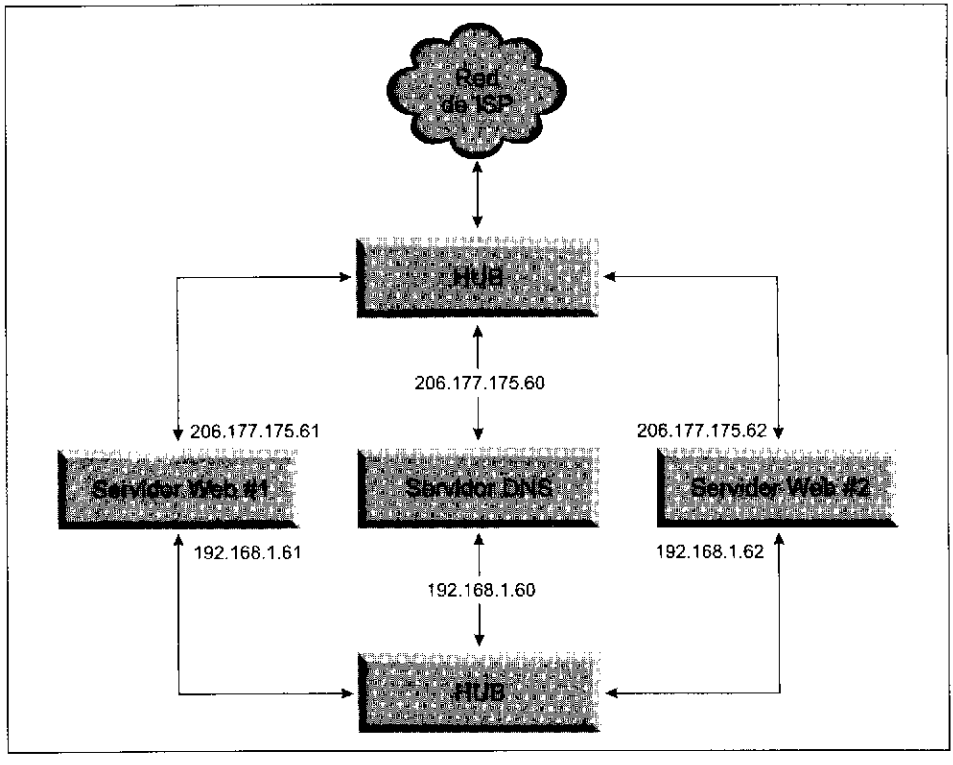


Figura 19.5. Asignación de las direcciones IP de la red

Elección de hardware y software

Como hemos mencionado anteriormente, la compañía XC News quiere utilizar servidores PC en vez de otras plataformas más caras, para que no tengan ninguna dificultad a la hora de añadir nuevos servidores según lo vayan necesitando.

Vamos a ver el hardware que necesitaría el servidor.

Configuración del servidor

Siempre es bueno trabajar con un sistema que se pueda actualizar sin ningún problema. Como Intel ha decidido utilizar un sistema basado en ranuras

de expansión con sus CPU, no parece muy buena idea trabajar con uno que tenga ranuras de expansión del tipo socket 7. Me parece que dentro de muy poco será muy complicado encontrar este tipo de tarjetas. Además, los nuevos procesadores Pentium II son muy rápidos y cualquier PC de hoy en día incorpora uno de estos procesadores. Mi consejo es utilizar un procesador Intel a 233Mhz en cada servidor. La placa madre es otro de los factores importantes, por lo que mi consejo es que se haga con una de Intel.

El siguiente factor a tener en cuenta es la cantidad de memoria RAM que se va a utilizar. La regla a seguir es que cuanta más se tenga, mejor. Pero el inconveniente es su precio, sobre todo el de los módulos SDRAM con un tiempo de acceso de 10 nanosegundos. Yo utilizaría 256MB de SDRAM en cada servidor.

Cada sistema ha de contar con un par de discos SCSI ultra-wide que se conectan a un controlador SCSI UW. Siempre es conveniente tener un par de discos por sistema, porque así se puede tener el sistema operativo en un disco y los datos en el otro. La capacidad total de disco duro la determinarán las necesidades propias de la empresa (tanto las presentes como las futuras). Para XC News, calculo que el espacio necesario está cerca de los 200MB, por lo que será conveniente tener un disco para datos de 4Gb.

También es bueno trabajar con un sistema de discos SCSI, en el que se pueden encadenar fácilmente nuevas unidades siempre que resulte necesario. Casi todos los controladores SCSI admiten entre 7 y 15 dispositivos encadenados.

Cada sistema tendrá dos adaptadores Ehternet PCI de 100Mbps, una tarjeta de vídeo SVGA y una caja con dos fuentes de alimentación. Como la tarjeta de vídeo se utilizará muy de vez en cuando, no hace falta que tenga gran cantidad de memoria RAM.

Del mismo modo, cada sistema ha de contar con una unidad CD-ROM IDE y una disquetera. La cara del servidor tiene que ser fuerte y robusta, suficientemente grande y con una buena fuente de alimentación. Usar dos fuentes de alimentación en vez de una hace que la red sea más segura. Si falla una, la otra se hace cargo del sistema.

Para la red de XC News habrá que utilizar equipo adicional: dos distribuidores Ethernet TX-100 (como se puede observar en la figura 19.5), cables, un monitor SVGA de 9 pulgadas y un teclado de 101 teclas. Todos estos dispositivos se guardarán en las instalaciones del ISP para utilizarlas en caso de emergencia.

Ahora que se ha determinado la configuración básica del hardware que utilizarán los servidores XC News, pasaremos a continuación a describir el software.

Seleccionar un sistema operativo

En el mercado hay disponibles varios sistemas operativos UNIX para PC x86, como por ejemplo, Linux, FreeBSD, BSDI y Solaris. Para XC News utilizaremos Linux. Llevo años utilizándolo en todo tipo de redes y siempre me ha ido muy bien. Especialmente con la distribución de RedHat Linux porque su asistente de instalación es muy sencillo de utilizar y de esta forma actualizará Linux para que disponga de las últimas propiedades. La versión comercial de RedHat Linux se puede encargar en www.redhat.com.

Elección de un servidor Web

La elección del servidor Web es muy sencilla. Vamos a utilizar la versión 1.2.6 de Apache en vez de la última versión beta, por cuestiones de estabilidad. La verdad es que no me gusta nada trabajar con versiones beta en los sistemas de producción.

Como ya hemos visto en capítulos anteriores, Stronghold es un programa que habría que tener en cuenta a la hora de seleccionar los sistemas de seguridad que protegerán el servidor Apache, así que añádalo su lista.

Ahora pasamos a configurar los servidores, de uno en uno.

Configuración de los sistemas

Para configurar los sistemas habrá que instalar RedHat Linux en cada uno de ellos. Pero antes de nada hay que decidir cómo se van a distribuir las particiones del disco.

Particiones de disco

Casi todos los sistemas UNIX tienen que tener un número estándar de particiones de disco (/root, /usr y /home). Los discos se tienen que dividir en particiones para que se pueda estandarizar el sistema operativo y otro software. Incluso un administrador nuevo de UNIX le dirá que el software del sistema operativo se encuentra en la partición / y la mayoría de las aplicaciones se encuentran en /user. Y es que seguir con la tradición impuesta en UNIX con el paso de los años facilita enormemente las tareas de cualquier administrador. En la tabla 19.2 aparecen las particiones que recomiendo para el disco del sistema operativo (OS) de cada ordenador de XC News.

Tabla 19.2. Particiones para el disco del sistema operativo

Partición	Tamaño en MB	Explicación
/	512	Partición root.
/usr	1024	Partición estándar en la que se instalan distintas aplicaciones de software.
/home	384	Directorio principal de los usuarios.
Swap	128	Espacio swap.

Obsérvese que la partición /home, donde se encuentran las directrices locales, no es demasiado grande. Esto es así porque la mayoría de los usuarios registrados no tienen demasiado material en sus directorios principales. De hecho, la mayoría de los clientes XC News harán que sus directorios principales apunten a los directorios Web, como veremos a lo largo del presente capítulo.

En la tabla 19.3 tiene las particiones de disco que aconsejo utilizar con los sistemas de XC News. Aunque se supone que los usuarios no van a registrarse en los sistemas de los servidores Web, mantendremos las mismas particiones que en los discos OS.

Tabla 19.3. Particiones para el disco de datos

Partición	Tamaño en MB	Explicación
/www	3.072	Partición para datos Web.
/log	1.024	Espacio para archivos de registro.

La partición de registro tiene que tener 1GB (1.024MB) porque este tipo de sitios Web generan un registro por cada una de las peticiones recibidas.

Instalación de Linux

Ahora ha llegado el momento de instalar Linux en los sistemas XC News. RedHat Software facilita enormemente el proceso de instalación de Linux. Todo lo que tiene que hacer es iniciar el sistema con el disco de inicio y las particiones deseadas. Hay varios tipos de instalaciones que se pueden utilizar con RedHat, pero aquí veremos la instalación desde CD-ROM porque éste es el método que lleva menos tiempo.

A continuación, instale el sistema operativo utilizando un programa de instalación. Al instalar el sistema operativo no incluirá ningún programa que no vaya a necesitar. Restringir la instalación de programas, además de reducir la cantidad de espacio de disco utilizado, mejora la seguridad del sistema. Por ejemplo, los servidores para CV News no necesitarán las X-Windows, por lo que tanto X-Windows como todo el software relacionado podrán excluirse de la instalación. XC News necesita un sistema operativo que se base en redes, por lo que cada máquina tendrá que contar con FTP, NFS, SMTP y el propio servidor Apache. Los servidores y lectores de news, editores extra, software de soporte, servicios de administración de redes, demonios de impresión, etc. no se tendrán que instalar.

En el sistema del servidor DNS he incluido el software BIND (Berkeley Internet Domain).

Configurar las redes

Ha llegado el momento de configurar las redes en cada sistema.

Comenzaremos con la red de Internet, que es la que se conecta a la red del ISP. El sistema para la red de Internet debería configurarse para que utilice las direcciones IP que aparecen en la tabla 19.4.

Tabla 19.4. Nombres de host y dirección IP de la red de Internet

Nombre del host	Dirección IP	Máquina
ns.xcnews.com	206.177.175.60	Servidor DNS
www1.xcnews.com	206.177.175.61	Servidor Web #1
www2.xcnews.com	206.177.175.62	Servidor Web #2

Estas direcciones IP no se han escogido al azar. Las asigna el ISP que suministra un servicio de conversión de nombres para los números IP, mientras que ns.xcnews.com se encarga de suministrar el servicio principal de nombres para él mismo y todos los dominios de sus clientes. El ISP también utiliza un servicio DNS secundario para todos los dominios de XC News.

El ISP suministra la máscara de red (255.255.255.0) y la puerta de enlace (gateway) IP predeterminada para esta red, de tal modo que todo el tráfico se dirigirá al gateway del sistema a través del distribuidor.

Seleccionamos un nombre ficticio para la red LAN, como xcnews-lan.com y asignamos todas las direcciones IP no routable, tal y como se puede apreciar en la tabla 19.5.

Tabla 19.5. Nombres de host y dirección IP de xcnews-lan.com

Nombre del host	Dirección IP	Máquina
ns.xcnews-lan.com	192.168.1.60	Servidor DNS
www1.xcnews-lan.com	192.168.1.61	Servidor Web #1
www2.xcnews-lan.com	192.168.1.62	Servidor Web #2

He hecho que la parte de la dirección IP del servidor sea la misma (60-62) que la de la otra red, para que al administrador le resulte más sencilla su memorización. La máscara de red también es 255.255.255.0 porque he utilizado direcciones de Clase C (192.168.1.0) y la puerta de enlace de esta red es el nombre del sistema del servidor, ns.xcnews-lan.com.

Una vez que se ha instalado el sistema operativo y se han configurado todas las redes, utilizaremos el programa ping para probarlas.

Para comprobar la red de Internet, enviamos peticiones ping de un servidor a otro, utilizando las direcciones IP apropiadas. Por ejemplo, para mandar un ping de ns.xcnews.com a www1.xcnews.com desde ns.xcnews.com con el comando:

```
ping 206.177.175.61
```

No se pueden utilizar los nombres de los host con el comando ping porque aún no se ha configurado el nombre del servidor. Cuando se hace un ping de un servidor a otro, observe que la pantalla LED de uno de los HUD parpadea rápidamente y la del otro permanece apagada. Es una forma de comprobar visualmente que los paquetes se envían a través del distribuidor correcto. Haga ping con las direcciones IP de la red xcnews-lan.com para que parpadee el LED del otro distribuidor. Obviamente, puede determinar la ruta que siguen los paquetes por medio de la tabla que genera el siguiente comando:

```
/sbin/route -n
```

Mostrará la tabla que aparece en el listado 19.1.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
206.177.175.0   0.0.0.0         255.255.255.0   U     0      0        179 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0         23 eth1
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0          2 lo
0.0.0.0         206.177.175.1   0.0.0.0         UG    0      0      53116 eth0
```

Listado 19.1. Tabla de redirecciones del servidor

Configuración del servidor DNS

El servicio DNS de la red Web que se utiliza con XC News utilizará un truco DNS conocido como "round-robin". En esta configuración, el servidor DNS se configurará de tal forma que suministrará una serie de direcciones IP a un único host. Como XC News quiere que todos los servicios de los servidores Web tengan la misma distribución de dominios, cada nombre de host tiene que apuntar a dos direcciones IP. En la figura 19.6 se puede ver este efecto.

En el listado 19.2 se muestra el contenido del archivo /etc.named.boot/ del servidor DNS. El servidor DNS se configura como principal para xcnews.com y xcnews-lan.com. Como el ISP no utiliza un sistema DNS invertido, el servidor se configura para que trabaje con el dominio 175.177.206.in-addr.arpa. De todas formas, para el dominio invertido 1.168.192.in-addr.arpa se configura como invertido.

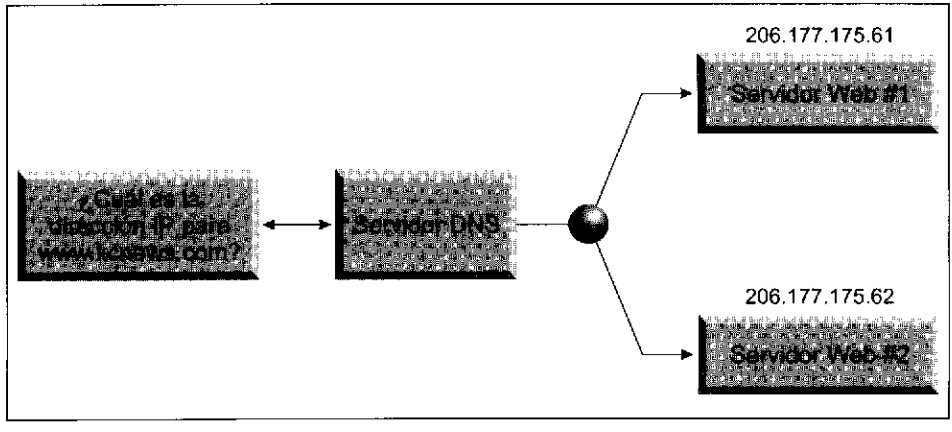


Figura 19.6. Dos direcciones IP por cada host Web

```
Directory                                       /var/named
Cache                                           named.ca
Primary          0.0.127.in-addr.arpa           named.local
;
; XC News domains
;
primary          xcnews.com                     xcnews.db
secondary    175.177.206.in-addr.arpa           xcnews.rev
;
; This domain is used for internal LAN that is not routed on
; the Internet.
;
```

```
primary          xcnews-lan.com            xcnews-lan.db
primary          1.168.192.in-addr.arpa    xcnews-lan.rev
```

Listado 19.2. Archivo /etc/named.boot

Ahora se mira en el archivo de la base de datos DNS llamado xcnews.db, cuyo listado aparece en el listado 19.3. Verá que, junto con los registros SOA, NS, MX, A y CNAME habituales, aparecen otros dos registros CNAME adicionales (CNAME = Canonical Name).

```
@ IN    SOA   xcnews.com.     hostmaster.xcnews.com. (
              19980205000     ; serial YYYYMMDDXXX
              7200            ; refresh
              3600            ; (1 hour) retry
              1728000         ; (20 days) expire
              3600)           ; (1 hour) minimal TTL
; Name server and Mail eXchange records
              IN    NS    ns.xcnews.com.
              IN    MX    5 mail.xcnews.com.

; A records
ns            IN    A     206.177.175.60
www1          IN    A     206.177.175.61
www2          IN    A     206.177.175.62

; CNAME records
mail          IN    CNAME      ns
ftp           IN    CNAME      ns
www           IN    CNAME      www1
www           IN    CNAME      www2
secured       IN    CNAME      ns
```

Listado 19.3. Archivo /var/named/xcnews.db

Los registros CNAME especiales son los siguientes:

```
www           IN    CNAME      www1
www           IN    CNAME      www2
```

Estos registros indican que www.xcnews.com es un CNAME de www1.xcnews.com y www2.xcnews.com. Entonces, ¿qué hace el servidor de nombres cuando recibe una petición de un cliente relacionada con la dirección de www.xcnews.com? Le entrega www1.xcnews.com o www2.xcnews.com, indistintamente. En realidad, esto es lo que ocurre con el sistema round-robin: el servidor DNS devuelve 206.177.175.61 (www1.xcnews.com) a la primera petición dirigida a www.xcnews.com y 206.177.175.62 (www2.xcnews.com) a la petición siguiente. Este truco permite redirigir al cliente solicitante hacia

uno de los servidores Web. El registro de la base de datos inversa de xcnews.com aparece en el listado 19.4.

```
@ IN    SOA   xcnews.com.      hostmaster.xcnews.com. (
              19980205000     ; serial YYYYMMDDXXX
              7200            ; refresh
              3600            ; (1 hour) retry
              1728000         ; (20 days) expire
              3600)           ; (1 hour) minimal TTL

; Name server and Mail eXchange records
              IN    NS    ns.xcnews.com.
              IN    MX    5 mail.xcnews.com.

; PTR Records
60            IN    PTR   ns.xcnews.com.
61            IN    PTR   www1.xcnews.com.
62            IN    PTR   www2.xcnews.com.
```

Listado 19.4. /var/named/xcnews.rev

Obsérvese que no hay ninguna necesidad de crear registros PTR para los registros CNAME, ya que tan sólo son alias para las entradas que tienen que tener un registro PTR. De esta forma, cuando un cliente quiere averiguar un nombre, tomará www1.xcnews.com o www2.xcnews.com, cada uno de los cuales tiene un registro PTR.

Los registros DNS de xcnews-lan.com aparecen en la figura 19.5.

```
@ IN    SOA   xcnews-lan.com.      hostmaster.xcnews-lan.com. (
              19980205000     ; serial YYYYMMDDXXX
              7200            ; refresh
              3600            ; (1 hour) retry
              1728000         ; (20 days) expire
              3600)           ; (1 hour) minimal TTL

; Name server and Mail eXchange records
        IN    NS    ns.xcnews.com.

; A records
ns      IN    A     192.168.1.60
www1    IN    A     192.168.1.61
www2    IN    A     192.168.1.62
```

Listado 19.5. /var/named/xcnews-lan.db

No hay ningún registro de intercambio de correo (MX) porque no hace falta. Los registros inversos DNS de xcnews-lan.com se muestran en el listado 19.6.

```
@ IN    SOA   xcnews-lan.com.      hostmaster.xcnews-lan.com. (
              19980205000      ; serial YYYYMMDDXXX
              7200             ; refresh
              3600             ; (1 hour) retry
              1728000          ; (20 days) expire
              3600)            ; (1 hour) minimal TTL

; Name server and Mail eXchange records
              IN    NS    ns.xcnews-lan.com.

; PTR Records
60            IN    PTR   ns.xcnews-lan.com.
61            IN    PTR   www1.xcnews-lan.com.
62            IN    PTR   www2.xcnews-lan.com.
```

Listado 19.6. /var/named/xcnews-lan.rev

La clave estará en facilitarle a XC News todo lo posible la agregación de nuevos servidores Web, además de usar el servicio de nombres round-robin. Si los administradores de sistema de la compañía deciden añadir un servidor nuevo, llamado www3.xcnews.com (206.177.175.63), todo lo que tendrán que hacer entonces será añadir las siguientes líneas en el archivo /var/named/xcnews.db:

```
www3    IN    A          206.177.175.63
www     IN    CNAME      www3
```

A continuación, tendrá que añadir una entrada PTR nueva en el archivo /var/named/xcnews.rev:

```
63      IN    PTR        www3.xcnews.com
```

Posiblemente haya que actualizar /var/named/xcnews-lan.db:

```
www3    IN    A     192.168.1.63
```

Por último, habrá que actualizar var/named/xcnews-lan.rev:

```
63      IN    PTR        www3.xcnews-lan.com
```

Una vez que se completen estos cambios, habrá que reiniciar el servidor y todas las peticiones dirigidas a www.xcnews.com se enviarán a los otros tres servidores.

Todo esto está muy bien, pero, ¿cómo aprovecharán los sitios Web de los clientes el servicio DNS round-robin? Vamos a verlo con un ejemplo. Supongamos que hay que configurar el cliente XC News, client-.onc.com, para que

trabaje con esta red. En este caso, se añadirá la línea siguiente al archivo /etc/named.boot:

```
primary              client one.com          client-one.db
```

Entonces, en /var/named, hay un archivo llamado client-one.db que habrá que crear de acuerdo con el listado 19.7.

```
@ IN    SOA  client-one.com.      hostmaster. client-one.com. (
             19980205000     ; serial YYYYMMDDXXX
             7200            ; refresh
             3600            ; (1 hour) retry
             1728000         ; (20 days) expire
             3600)           ; (1 hour) minimal TTL

; Name server and Mail eXchange records
             IN     NS  ns.xcnews.com.

; CNAME records for www.client-a.com
www          IN     CNAME     www.xcnews.com.
```

Listado 19.7. client-one.db

Así se configura ns.xcnews.com como servidor DNS para el client-one.com y hace que www.client-a.com sea www.xnews.com. Como www.xnews.com es un alias de www1.xcnews.com y www2.xcnews.com el servidor DNS mostrará las direcciones IP de ambos host. De esta forma, se utiliza el sistema round-robin tal y como se había planeado.

Ahora, si XC News añade un nuevo servidor Web, no habrá que cambiar los registros DNS de client-a.com mientras www.xcnews.com siga siendo el alias del nuevo servidor Web.

¿Qué ocurriría se alguno de los servidores Web dejase de funcionar? Por ejemplo, supongamos que hay que dejar www1.xcnews.com temporalmente fuera de servicio. ¿Habría que cambiar su DNS? En el archivo /var/named/xcnews.db tiene que dirigirse a la siguiente entrada:

```
www        IN    CNAME       www2
```

y dejarla sin comentar:

```
; www      IN    CNAME       www1
```

A continuación, habrá que reiniciar el servidor DNS para que los cambios sean efectivos. De todas formas, debido a la naturaleza del servicio DNS, los cambios aún tardarán cierto tiempo en propagarse.

Aunque la configuración de los clientes de XC News se haya construido para que sea lo más simple posible, es posible que se dé un error humano en el proceso. Por ejemplo, si al final de la línea no aparece un punto, tal y como se puede ver en la línea siguiente, la configuración no funcionará:

```
www        IN    CNAME        www.xcnews.com
```

Para reducir la probabilidad de que aparezca un error humano, habrá que crear un script en Perl que utilice una plantilla (como la que aparece en el listado 19.8) para crear el archivo de la base de datos del nuevo cliente. Además, la plantilla modifica el archivo /etc/named.boot para que ns.xcnews.com sea el DNS primario del dominio del cliente.

```
@       IN    SOA        <DOMAIN>. hostmaster.<DOMAIN>. (
                         19980205000     ; serial YYYYMMDDXXX
                         7200            ; refresh
                         3600            ; (1 hour) retry
                         1728000         ; (20 days) expire
                         3600)           ; (1 hour) minimal TTL

; Name Servers
        IN    NS         ns.xcnews.com.
        IN    MX         10 mail.xcnews.com.

; CNAME records
mail          IN    CNAME          ns.xcnews.com.
ftp           IN    CNAME          ns.xcnews.com.
www           IN    CNAME          www.xcnews.com.
```

Listado 19.8. named.template

Observe que, en esta plantilla, el nombre de dominio se especifica como una etiqueta llamada <DOMAIN>. El script la sustituirá por el nombre del cliente. Además, habrá que añadir cierta configuración adicional al servidor Apache para incluir al nuevo cliente. Como en este proceso también se incluye el error humano, dejaremos que el script en Perl cree la configuración del contenedor <VirtualHost ...> usando otra plantilla, como se puede apreciar en el listado 19.9.

```
# Domain Configuration for  <WWW-SITE>
#
<VirtualHost REPLACE-THIS-WITH-WEB-SERVER-IP-ADDR>
ServerAdmin webmaster@<DOMAIN>
DocumentRoot       <HTDOCS-DIR>
#ScriptAlias       /cgi-bin/       <CGI-BIN-DIR>
ScriptAlias        /cgi-bin/       /www/xcnews/cgi-bin/
Alias              /cgi-data/      /www/cgi-data/
```

```
ServerName          <WWW-SITE>
ErrorLog            logs/<WWW-SITE>.error.log
TransferLog         logs/<WWW-SITE>.access.log
</VirtualHost>
#
# End of Domain Configuration for <WWW-SITE>
```

Listado 19.9. httpd.template

El script en Perl, llamado makesite, sustituye la etiqueta <WWW-SITE> con www.nombre-dominio-cliente. El listado 19.10 nos muestra este script.

```
#!/usr/local/bin/perl
#
# Propósito: makesite crea sitios virtuales. Utiliza una serie de
# plantillas para crear configuraciones DNS, HTTPD y SMTP
# y las agrega a los archivos de configuración adecuados.
#
# $Author$ (kabir@nitec.com)
# $Version$
# $Id$
# $Status
#
##################################################################

# Es posible que tenga que cambiar algunos de estos valores antes
# de utilizar el script en su sistema.
#

# Directorio en el que se encuentra el script
my $MAKESITE_DIR = '/usr/local/build/makesite';

# Nombre del usuario al que pertenece el nuevo cliente
# directory
my $USER = 'httpd';

# Grupo al que pertenece el nuevo cliente
# directory
my $GROUP = 'httpd';

# Archivo predeterminado de permisos para $USER.$GROUP
my $PERMISSION = '2770';

# Directorio de la partición Web
my $BASE_DIR = '/www';

# Directorio raíz de documentos (relativo a
# /$BASE_DIR/<CLIENT-DOMAIN-DIR>)
my $HTDOCS = 'htdocs';

# Directorio de programas CGI (relativo a
# /$BASE_DIR/<CLIENT-DOMAIN-DIR>)
my $CGIBIN = 'cgi-bin';
```

```
# FQPN del directorio de registros DNS de BIND
my $NAMED_PATH = '/var/named';

# Extensión de los archivos de registro de cada dominio
my $NAMED_FILE_EXT = '.db';

# FQPN de la plantilla de registros DNS de BIND
my $NAMED_TEMPLATE_FILE = "$MAKESITE_DIR/named.template";

# FQPN del archivo BIND named.boot
my $NAMED_ETC_FILE = '/etc/named.boot';

# FQPN del archivo de configuración httpd.conf de Apache
my $HTTPD_CONF_FILE = '/www/apache/conf/httpd.conf';

# FQPN de la plantilla del host virtual
my $VIRTUAL_HOST_TEMPLATE = "$MAKESITE_DIR/httpd.template";

# FQPN del archivo de registro
my $LOG_FILE = "$BASE_DIR/makesite.log";
my $date = `/bin/date`;

# Elimina el carácter de nueva línea de la fecha.
chomp($date);

# Toma el primer argumento que se entrega al script.
my $site = $ARGV[0];

# El argumento es el nombre del sitio (como nitec.com)
# Se le asigna a la variable $dir.
my $dir = $site;

# Guarda los tipos de dominio con los que se puede trabajar en una
# array
my @domain_types = (com,net,org,edu);

my $domain_ext;         # se utiliza para guardar la extensión del
                        # dominio
my $thesite_dir;        # se utiliza para guardar el nombre del
                        # directorio del sitio
my $htdocs_dir;         # se utiliza para guardar el nombre del
                        # directorio DocumentRoot
my $cgibin_dir;         # se utiliza para guardar el nombre del
                        # directorio cgi-bin
my $named_file;         # se utiliza para guardar el nombre del
                        # archivo DNS del sitio Web
my $dir_len;            # nombre del directorio length
my $temp_len;           # variable temp

# nombre del sitio suministrado por el usuario.
$site =~ y/[A-Z]/[a-z]/;

# Si al script no se le suministra ningún argumento, se vaciará
# y se mostrará la siguiente sintaxis
&syntax if($site eq '');
```

```
# Toma lenght con EXT
$dir_len   length($dir);

# Elimina la extensión del dominio del nombre suministrado
foreach $domain_ext (@domain_types){
   $dir =~ s/\.$domain_ext//g;
   }

# Toma el nuevo length sin EXT
$temp_len = length($dir);

# Si el usuario no ha introducido una extensión mostrará la sintaxis
&syntax if($temp_len == $dir_len);

# Crea nombres FQPN
$named_file = $NAMED_PATH . '/' . $dir . $NAMED_FILE_EXT ;
$thesite_dir = $BASE_DIR . '/' . $dir;
$htdocs_dir = $BASE_DIR . '/' . $dir . '/' . $HTDOCS;
$cgibin_dir = $BASE_DIR . '/' . $dir . '/' . $CGIBIN;

# Si ya existe el directorio del sitio mostrará un mensaje de error
# y finalizará.
if(-e $thesite_dir){
   print "$thesite_dir already exist! No action taken for
$thesite_dir\n";
   exit 0;
   }

# Ha llegado el momento de crear los directorios necesarios.
system("mkdir $thesite_dir");
system("mkdir $htdocs_dir");
system("mkdir $cgibin_dir");

# Crea el archivo de registro DNS para este sitio Web
&createNamedFile($named_file,$site,"$dir$NAMED_FILE_EXT");

# Crea un índice para el directorio DocumentRoot.
&createIndexFile($htdocs_dir,$site);

# Crea el archivo de configuración httpd.conf para el host virtual
&createVirtualHostConf(domain=>$site,website=>"www.$site",
cgibin=>$cgibin_dir,htdocs=>$htdocs_dir);

# Cambia el propietario y los permisos de acceso de los directorios
# del sitio Web.
system("chown -R $USER.$GROUP $thesite_dir");
system("chmod -R $PERMISSION $thesite_dir");

# Escribe un archivo de registro
open(FP,">$LOG_FILE") || die "Can't write to log file.\n";
print FP "Date: $date created www.$site [$htdocs_dir] \n";
close(FP);

# ¡Hecho!
exit 0;
```

```
sub createNamedFile{
#
# Propósito: crea el archivo de registro DNS en formato BIND
#

# Toma los parámetros que se le han entregado
   my $file = shift;
   my $domain = shift;
   my $database = shift;
   my $line;

# Abre el archivo de registro DNS para escribirlo
   open(OUT,">$file")   | die "Can't write $file\n";

# Abre la plantilla para proceder a su lectura
   open(FP,$NAMED_TEMPLATE_FILE) || die "Can't open
$NAMED_TEMPLATE FILE\n";

# Lee cada línea de la plantilla y sustituye la etiqueta
# <DOMAIN> con el nombre del sitio Web especificado
# que se le entrega a través de la variable $domain
#
   while($line=<FP>){
     $line =~ s/<DOMAIN>/$domain/g;
     print OUT $line;
     }
# Cierra todos los archivos que acaba de abrir
  close(FP);
  close(OUT);

# Abre el archivo named.boot y agrega el sitio Web al archivo de
# configuración del DNS principal.

   open(FP,">$NAMED_ETC_FILE") || die "Can't open
$NAMED_ETC_FILE\n";

# Escribe el registro
   print FP "primary\t\t$domain\t\t$database\n";

# Cierra el archivo
   close(FP);
   }

sub createIndexFile{
#
# Propósito: crea un índice index.html en el directorio
# DocumentRoot del nuevo sitio.

# Toma los argumentos que le entrega esta subrutina
  my $htdocs = shift;
  my $domain = shift;
   # escribe en el archivo index.html.
  open(FP,">$htdocs/index.html") || die "Can't write index.html \n";
  print FP <<INDEX_PAGE;
  <HTML>
```

```
  <HEAD> <TITLE> $domain </TITLE> </HEAD>
  <BODY BGCOLOR="white">
  <CENTER> This is $domain </CENTER>
  </BODY>
  </HTML>
INDEX_PAGE

# Cierra el archivo
  close(FP);
  }

sub createVirtualHostConf{
#
# Propósito: crear una configuración para el host virtual
# utilizando la plantilla y guardando la configuración en el
# archivo httpd.conf.

# Toma todos los parámetros que le entrega esta subrutina
  my %params = @_;

# Variable Temp.
  my $line;

# Abre httpd.conf para agregar datos
  open(OUT,">$HTTPD_CONF_FILE")  | die "Can't open
$HTTPD_CONF_FILE\n";

# Abre la plantilla del host virtual host para proceder a su
# lectura
  open(FP,$VIRTUAL_HOST_TEMPLATE)  | die "Can't open
$VIRTUAL_HOST_TEMPLAT\n";

# Lee cada línea de la plantilla y sustituye las etiquetas
# especiales por sus verdaderos valores
  while($line=<FP>){
    $line =~ s/<DOMAIN>/$params{domain}/g;
    $line =~ s/<CGI-BIN-DIR>/$params{cgibin}/g;
    $line =~ s/<HTDOCS-DIR>/$params{htdocs}/g;
    $line =~ s/<WWW-SITE>/$params{website}/g;
    print OUT $line;

    }
# Cierra todos los archivos
  close(FP);
  close(OUT);
  }
sub syntax{
#
# Propósito: mostrar la sintaxis del script.
#
    print <<SYNTAX;

    makesite <virtual Internet domain>

    Ejemplo: makesite nitec.com
```

```
    No confunda el nombre del dominio con el nombre del host
(www.nitec.com).
    Este script creará el host www.<dominio virtual de Internet>
automáticamente.

SYNTAX

# Finalizar después de mostrar la sintaxis.
   exit 0;
   }
```

Listado 19.10. El script makesite

Este script se puede ejecutar de la siguiente manera:

```
./makesite nombre-dominio-cliente
```

Por ejemplo:

```
./makesite client-a.com
./makesite client-b.org
```

Así se crea la configuración DNS y Apache para client-a.com y client-b.com. Los archivos de configuración DNS se guardarán en /var/named/client-a.db y /var/named/client-b.db. La configuración Apache se guardará en el archivo/www/apache/conf/httpd.conf. Este script creará una estructura de directorios para cada uno de estos sitios Web. Por ejemplo, client-a.com tendrá la estructura de directorios que aparece en la figura 19.7.

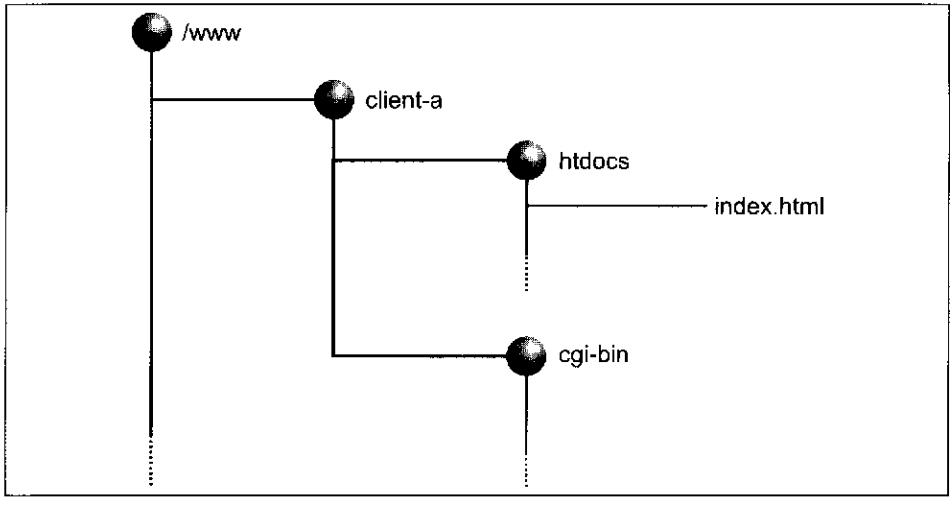


Figura 19.7. Estructura de directorios que tendrá client-a.com

El script makedir también creará una página index.html con la que se facilitará la prueba del sitio que se acaba de crear. Después de ejecutar el script makesite con cada nuevo dominio, tendrá que reiniciar el servidor Apache y agregar los archivos de configuración (las nuevas secciones de los contenedores <VirtualHost> que se encuentran en el archivo /www/apache/conf/httpd.conf) al httpd.conf de cada servidor Web. Por ejemplo, cuando se ejecuta makesite client-a.com, la configuración de Apache (que aparece en el listado 19.11) se anexa al archivo /www/apache/conf/httpd.conf.

```
# Domain Configuration for www.client-a.com
#
<VirtualHost REPLACE-THIS WITH-WEB-SERVER-IP-ADDR>
ServerAdmin webmaster@www.client a.com
DocumentRoot      /www/client-a/htdocs
#ScriptAlias      /cgi bin/       /www/client-a/cgi-bin/
ScriptAlias       /cgi-bin/       /www/xcnews/cgi-bin/
Alias             /cgi-data/      /www/cgi-data/
ServerName        www.client-a.com
ErrorLog          logs/cerror.log
TransferLog       logs/ www.client-a.com.access.log
</VirtualHost>
#
# End of Domain Configuration for www.client-a.com
```

Listado 19.11. Archivo /www/apache/conf/httpd.conf

A continuación se tomará esta sección de /www/apache/conf/httpd.conf y se pegará en los archivos de configuración httpd.conf de www1.xcnews.com y www2.xnews.com. Habrá que sustituir REPLACE-THIS-WITH-WEB-SERVER-IP-ADDR por la dirección IP del servidor. Por ejemplo, cuando se añada esta sección de configuración al archivo httpd.conf del servidor Apache de www1.xcnews.com, se sustituye REPLACE-THIS-WITH-WEB-SERVER-IP-ADDR por 206.177.175.61.

No se tiene que reiniciar el servidor Apache hasta que se repartan los nuevos directorios por los servidores Web. Así pasamos a los aspectos de la distribución de archivos de este proyecto.

Distribución de ficheros con rdist

El programa rdist le permite conservar copias idénticas de los archivos distribuidos por varios host. Utiliza las llamadas rcmd o el shell remoto (rsh) para acceder a cada uno de los ordenadores host.

La forma más sencilla de hacer que rdist funcione es crear una cuenta común a todas las máquinas y crear archivos .rhosts para cada sistema (www1 y

www2). Así, el usuario que se encuentra en el servidor de nombres podrá ejecutar sesiones rsh. Por esto mismo, se crea un usuario llamado httpd en los tres sistemas. En cada uno de los servidores Web se añade un archivo .rhosts en el directorio principal del httpd del user. Este archivo contiene una línea de configuración:

```
ns.xcnews-lan.com
```

Este archivo pertenece al usuario root y el resto de usuarios tan sólo tendrán permiso de lectura. De esta forma, un usuario llamado httpd que se encuentre en ns.xcnews-lan.com podrá ejecutar sesiones shell remotas en cada uno de los servidores Web. Obsérvese que el host ns.xcnews-lan.com se utiliza para esto mismo. Esto se debe a que hace falta una red LAN interna con el tráfico rdist. El siguiente paso será crear un distfile para rdist. Un distfile es un archivo de texto que contiene las instrucciones para rdist a través de las cuales se le muestra cómo efectuar las tareas de distribución. El listado 19.12 muestra uno de estos distfile, rdist_distfile.

```
#
# Distfile para rdist
#
# Se utiliza para distribuir archivos de ns.xcnews-lan.com
# a los sistemas www[12].xcnews-lan.com
#
# $Author$ (kabir@nitec.com)
# $Version$
# $Date$
# $Id$

# Lista todos los host que hay que actualizar.

# La lista se genera utilizando las entradas separadas por un
# espacio en blanco de user@hostname.
#
HOSTS = (httpd@www1.xcnews-lan.com httpd@www2.xcnews-lan.com)

# Lista todos los host que hay que actualizar.
#
FILES = ( /www)

# Lista los directorios que hay que excluir del proceso de
# actualización.
EXCLUDE_DIR    (/www/cgi-data/    /www/apache /www/secured)

# Estos son los comandos:
# Instale todos los directorios que aparecen en la lista en FILES
# para todos los host listados en HOSTS, a excepción de los
# directorios que aparecen en EXCLUDE_DIR
#
```

```
${FILES} -> ${HOSTS}
  install ;
  except ${EXCLUDE_DIR};
```

Listado 19.12. rdist_distfile

Es un archivo distfile muy sencillo. Define una variable llamada HOST que tiene dos valores: httpd@www1.xcnews-lan.com y httpd@www2.xcnews-lan.com. Le indica a rdist que utilice la cuenta del usuario httpd para la conexión con www1.xcnews-lan.com y www2.xcnews-lan.com. Para la siguiente variable, FILES, se definen los archivos y directorios que tendrá que distribuir rdist. Como la zona de construcción es /www en ns.snews-lan.com, lo único que se utiliza como valor es una sola entrada.

La tercera variable es EXCLUDE_DIR. Esta variable se configura para que liste todos los archivos y directorios que se van a excluir de la distribución. El primero, /www/cgi-data/, es el directorio de datos CGI donde todos los script CGI escriben sus datos. Este directorio se exportará al host del servidor Web a través de NFS, por lo que no hace falta utilizar rdist para copiarlo en todos y cada uno de los servidores. El directorio /www/apache es donde makesite escribe el archivo /www/apache/conf/httpd.conf, que hay que copiar porque cada servidor tiene su propio archivo de configuración en el directorio local /www/apache. El valor final es /www/secured, que el servidor de seguridad utiliza como raíz de documentos, y también habrá que copiarlo en el resto de los servidores. El resto del archivo se limita a describir un único comando:

```
${FILES} -> ${HOSTS}
  install ;
  except ${EXCLUDE_DIR};
```

Este comando toma todos los archivos y directorios especificados en la variable FILES y los instala en los host indicados en la variable HOST. También le indica a rdist que excluya los archivos y directorios especificados en la variable EXCLUDE_DIR. Para ejecutar rdist (como httpd), utilice la siguiente sentencia desde la línea de comandos:

```
/usr/bin/rdist -p /usr/sbin/rdistd -oremove,quiet -f
/usr/local/rdist/ rdist.distfile
```

La opción -p especifica la ubicación del programa rdist. El parámetro -o especifica que hay una o más opciones a continuación (en este caso, eliminar y quieto). La opción eliminar le dice a rdist que elimine cualquier archivo extraño que se encuentre en los directorios del sistema final. Así, se facilita el

mantenimiento de zonas de construcción en cada servidor Web. La opción quiet le dice a rdist que se esté quieto durante la operación. La última opción, -f, especifica la ubicación de distfile.

Para reducir la cantidad de errores humanos a la hora de ejecutar este comando, cree un script sh llamado rdisstribute.sh, como el que le mostramos en el listado 19.13.

```
#!/bin/sh
#
# Este script ejecuta runs para actualizar los servidores Web a
# través la red xcnews-lan.com. El encargado de ejecutar el script
# es cron y lo hace cada cierto tiempo.
#
# /etc/rc.d/rc.local inicia el script que limpiará
# las plantillas que han sobrado después del cierre del sistema.
# Este proceso también elimina el archivo de registro.
#
# $Author$ (kabir@nitec.com)
# $Version$
# $Id$
# $Date$
# $Status
##############################################################

# Si la llamada al script incluye un argumento, entonces
case "$1" in
  boot)
  # como el argumento es 'boot', se llamará al script durante el
  # inicio del sistema, así que retire todos los antiguos bloqueos
  # de archivos y registros.
  echo -n "Cleaning up rdistribute.sh tmp files: "
     rm -f /tmp/rdist.lck
        rm -f /tmp/rdist.log
  echo "complete."
  exit 0;
  ;;

  # como el argumento es 'restart', el script
  # se tiene que purgar como si se reiniciase el sistema.
  restart)
  $0 boot
  ;;

esac

# Si hay algún archivo bloqueado, no hace nada.
if [ -f /tmp/rdist.lck ]; then
   exit 0
fi

# Otra forma de crear archivos bloqueados es utilizando /bin/touch
/bin/touch  /tmp/rdist.lck
```

```
# Ejecuta rdist
/usr/bin/rdist -p /usr/sbin/rdistd -
oremove,nochkgroup,nochkmode,nochkowner,quiet -f
/usr/local/rdist/rdist_distfile

# Archivo bloqueado remoto
rm -f /tmp/rdist.lck

# Escribe la fecha y la hora en el archivo de registro
echo `date` >> /tmp/rdist.log

# Finaliza el script
exit 0
```

Listado 19.13. Script rdistribute.sh

El script es lo suficientemente inteligente como para detectar mientras se ejecuta el proceso rdistribute.sh a través de un archivo de bloqueo, que le avisa cuando hay un rdistribute.sh en ejecución. Esto puede pasar cuando se está actualizando una gran cantidad de archivos en varios servidores. El script también acepta un argumento llamado boot que se puede utilizar para limpiar el archivo de bloqueo y el de registro durante el proceso de arranque. Se puede llamar al script desde /etc/rc.d/rc.local de la siguiente manera:

```
/usr/local/rdistribute.sh boot
```

Este script se puede programar para que se ejecute con cron /etc/crontab. Por ejemplo, para ejecutar este script cada 10 minutos, tendrá que agregar la siguiente entrada en /etc/crontab:

```
0,10,20,30,40,50 * * * * httpd  /usr/local/rdistribute.sh >
/dev/null
```

El demonio cron ejecutará el script como httpd.

Uso de NFS en una red interna

De momento hemos configurado la distribución de archivos y DNS, pero aún no podemos reiniciar los servidores. Antes de nada, hay que montar NFS en la zona de datos CGI /www/cgi-data del host Web para que el servicio Web sea completamente funcional. Para que los servidores Web puedan acceder al directorio de datos CGI (/www/cgi-data/), habrá que configurar el servidor de nombres como servidor NFS y los servidores Web como clientes NFS.

Configurar un servidor NFS

Un servidor NFS tiene que ejecutar un programa llamado portmapper (también tiene otros nombres como portmap o rcp.portmap) que se ejecuta con el script rc. Para comprobar si portmapper ya está ejecutándose en su sistema, utilice el siguiente comando:

```
ps auxw | grep portmap
```

Si se ejecuta con RedHat Linux verá que portmapper lo inició automáticamente el script /etc/rc.d/rc3.d/S40portmap (es decir, el script /etc/rc.d/init.d/portmap.init), por lo que no tendremos que reiniciarlo manualmente para utilizarlo con XC News.

El siguiente paso será modificar el archivo /etc/exports para indicarle al sistema qué archivo o directorios hay que exportar a los clientes NFS. Como XC News sólo quiere copiar el directorio /www/cgi-data, el archivo de exportación para de ns.xcnews-lan.com tendrá el siguiente aspecto:

```
/www/cgi-data      www1.xcnews-lan.com(rw) www2.xcnews-lan.com(rw)
```

Esta línea le indica al servidor NFS que permita que los servidores www1.xnews-lan.com y www2.xcnews-lan.com tengan permiso de escritura y lectura en el directorio /www/cgi-data.

Nota: puede que la sintaxis para exportar archivos no sea la misma en todas las versiones de UNIX.

A continuación, el programa tendrá que ejecutar mountd (rpc.mountd) y nfsd (rcp.nfsd). Estos dos programas se inician automáticamente desde los script rc en /etc/rc.d/rc3.d. De todas formas, siempre que se efectúe algún cambio en el fichero /etc/exports, habrá que comunicárselo a estos dos archivos. Se puede utilizar un script que se llama exportfs para reiniciar estos programas:

```
exportfs
```

Si exportfs no se encuentra en un sistema, utilice un script como éste:

```
#!/bin/sh
killall -HUP /usr/sbin/rpc.mountd
killall -HUP /usr/sbin/rpc.nfsd
echo re-exported file systems
```

Utilice el programa killall que se encuentra en la mayoría de los sistemas Linux como RedHat. Si no está disponible, use el comando ps, localice el PID de estos procesos y finalice manualmente todos los procesos. Ahora asegúrese de que los programas mountd y nfsd funcionan correctamente y ejecute un programa llamado rpcinfo:

```
rpcinfo -p
```

La salida tiene este aspecto:

```
100000    2       tcp        111       rpcbind
100000    2       udp        111       rpcbind
100005    1       udp        635       mountd
100005    2       udp        635       mountd
100005    1       tcp        635       mountd
100005    2       tcp        635       mountd
100003    2       udp        2049      nfs
100003    2       tcp        2049      nfs
```

Indica que portmapper, mountd y nfsd funcionan correctamente. Antes de configurar el lado del cliente de NFS de los servidores Web, es importante asegurarse que se cumplen los siguientes puntos relacionados con la seguridad.

Seguridad del servidor

Si se utiliza portmapper, en combinación con nfsd, es posible hacerse con los archivos que se encuentran en el servidor NFS sin necesidad de privilegio. Afortunadamente, el portmapper que utiliza Linux ya es bastante seguro de por sí. Además, si se le añade la siguiente línea a /etc/hosts.deny:

```
portmap: ALL
```

El sistema negará el acceso portmapper a todo el mundo. Ahora, hay que efectuar la siguiente modificación en el archivo /etc/host.allow:

```
portmap: 192.168.1.0/255.255.255.0
```

Permite que todos los host de la red 192.168.1.0 disfruten de un acceso administrado por programas portmapper como nfsd y mountd.

Advertencia: nunca utilice los nombres de los host en la línea de portmap que se incluye en /etc/host.allow porque puede provocar cierta actividad por parte de portmap que disparará un bucle en la búsqueda de dichos nombres.

Otro aspecto relacionado con la seguridad concerniente al servidor lo encontramos al permitir que la cuenta root del cliente se trate como tal en el servidor. Por defecto, Linux lo prohibe. Es decir, el usuario root no puede modificar un archivo que se exporte desde la cuenta root del servidor. Para obligar a que se cumpla esta regla, se puede modificar el archivo /etc/exports de la siguiente manera:

```
/www/cgi-data    www1.xcnews lan.com(rw, root_squash)
www2.xcnews-lan.com(rw, root_squash)
```

Ahora, si un usuario usa un UID 0 (el número de identificación del usuario root) en un intento del cliente de acceder (leer, escribir o climinar) al archivo de sistema, el servidor sustituirá el UID de su cuenta "nobody" (nadie). Es decir, que el usuario root del cliente no puede acceder o cambiar los archivos que únicamente la cuenta root del servidor tiene permiso para modificar.

Nota: para garantizar el acceso al sistema NFS, conviene que utilice la opción no_root_squash.

Ya hemos configurado y protegido el servidor, así que ahora nos meteremos con la configuración de los host del cliente NFS.

Configuración de un cliente NFS

RedHat puede trabajar por omisión con los archivos NFS, por lo que no hay que tocar nada del kernel. Para montar el directorio /www/cgi-data que ha exportado ns.xnews-lan.com, añadirá la siguiente línea en el archivo/etc/fstab de los dos servidores Web:

```
ns.xcnews-lan.com:/www/cgi-data          /www/cgi-data       nfs
```

La línea anterior prepara automáticamente el directorio /www/cgi-data cuando se reinicia cualquiera de los dos servidores. A continuación crea el directorio / www/cgi-bin en ambos sistemas y se tendrá que montar el directorio manualmente por medio del comando mount:

```
ns.xcnews-lan.com:/www/cgi-data          /www/cgi-cata       nfs
```

Truco: uno de los problemas más frecuentes relacionados con el montaje NFS ocurre cuando se olvida ejecutar exportfs (es decir, reiniciar rpc.mountd y rpc.nfsd) una vez que se ha modificado el archivo /etc/exports que se encuentra en el servidor NFS.

Para desmontar un sistema de archivos NFS se hace exactamente igual que el desmontaje de un archivo de sistema local. También se puede mejorar la seguridad del cliente NFS. Bastará con no fiarse demasiado del servidor NFS. Por ejemplo, puede desactivar programas suid para desahogar al sistema de archivos NFS sin tener que utilizar la opción nosuid. Con esto el usuario root del servidor no podrá crear un programa suid-root en el archivo del sistema, registrarse en el cliente como un usuario normal y corriente para utilizar su programa suid-root y obtener así los privilegios de la cuenta root. También se pueden prohibir las exclusiones de todos los ficheros de un sistema de archivos NFS utilizando la opción noexec. Puede introducir estas opciones en la columna correspondiente de la línea que describe el montaje del sistema NFS del archivo /etc/fstab.

Vamos a configurar Apache y a cerciorarnos de que es seguro.

Configuración de Apache

La configuración del servidor Apache será como siempre. Para trabajar con XC News no tiene que cumplir ningún requisito especial.

Las secciones del contenedor <VirtualHost> que produce el script makesite se tienen que añadir al archivo httpd.conf de configuración de cada servidor Web. Como ya hemos comentado, los host virtuales determinan las direcciones IP específicas de cada servidor Web. Al utilizar un script /etc/rc.d/rc3.d/S85httpd (es decir, un script /etc/rc.d/init.d/httpd), asegúrese que el servidor arrancará al final del proceso de reinicio.

Para los servicios Web protegidos, recomiendo utilizar Stronghold en secured.xcnews.com, que es otro CNAME de ns.xcnews.com. Como ya hemos visto al principio del capítulo, XC News no quiere trabajar con una configuración que obligue a cada host virtual a pagar por un certificado de CA, por lo que parece que es conveniente configurar Stronghold en el servidor protegido .xcnews.com.

En esta configuración, el servidor Stronghold apunta a /www/secured como el directorio raíz de los documentos. Así, XC News puede crear subdirectorios bajo /www/secured para que sus clientes guarden los archivos a los que hay que acceder a través de SSL, es decir, de canales seguros de comunicación. Por ejemplo, si el sitio del cliente www.client-a.com tiene que instalar un formulario de pedido protegido llamado order.html, lo primero que tiene que hacer es pedirle a XC News que cree un directorio dentro de /www/secured en el que guardará dichos archivos. XC News creará el subdirectorio /www/secured/client-a/ y guardará order.html en él. Ahora, el cliente se ase-

gurará de que todas las referencias (vínculos HREF) dirigidas a order.html apuntan al URL https://secured.xcnews.com/client-a/order.html.

Como XC News también quiere que se pueda acceder a los script CGI que se encuentran en www.xcnews.com (/www/xcnews/cgi-bin) a través del servidor protegido, utilizará una directriz ScriptAlias con el archivo httpd.conf de configuración de Stronghold:

```
ScriptAlias /cgi-bin/    /www/xcnews/cgi-bin/
```

De esta forma, el servidor protegido podrá ejecutar los mismos script CGI.

Los documentos del cliente han de incluir un vínculo que apunte al sitio correspondiente. Por ejemplo, si el formulario de XC News lo procesó un script CGI, en las páginas que se generen tendrá que aparecer un vínculo a la página original. Así se consigue una configuración más agradable.

Siempre que uno de los clientes de XC News genere una página HTML la guardará dentro del subdirectorio protegido que ha creado con anterioridad, por lo que hay que comprobar que el vínculo apunte al sitio Web del cliente. Si la página carece de este enlace, los visitantes estarán algo confundidos.

Para todos los formularios HTML que se puedan procesar y otros script CGI, XC News debería asegurarse de que los script se escriben de tal forma que añadan un salto al sitio Web original. Por ejemplo, un formulario HTML podría contener una variable oculta como ésta:

```
<INPUT TYPE=HIDDEN NAME="return url" VALUE="http://www.client-a.com/
thanks_for_the_order.html" >
```

Se tendrían que utilizar los script CGI encargados de procesar formularios para que redirigieran a los usuarios al URL que se suministra como valor de la variable return-url o para que creasen una página de salida que tuviese un enlace llamado "Volver a nuestro sitio Web" o algo parecido.

Cuentas de usuario FTP para clientes

XC News quiere que sus clientes puedan actualizar sus archivos Web a través de FTP de tal modo que no haya que crearlos en el servidor ns.xcnews.com. Como se ha configurado el alias ftp.cxnews.com para xcnews.com en el sistema DNS, se puede utilizar como alias IP del servidor FTP oficial.

RedHat tiene un programa llamado /usr/sbin/adduser (o /usr/sbin/useradd) que se puede utilizar para crear nuevos usuarios desde la línea de comandos. El programa utiliza la configuración predeterminada almacenada en el archivo

/etc/default/useradd. Este fichero contiene la siguiente configuración para el directorio principal:

```
HOME=/home
```

Cámbielo a HOME=/www para que cada cuenta de clientes tenga su directorio Web como directorio principal, siempre y cuando coincidan el nombre de la cuenta del cliente y el de su dominio.

Nota: el directorio /www se encuentra en la ruta del directorio principal del usuario, por lo que cualquiera tiene que poder leerlos y ejecutarlos.

Por ejemplo:

```
adduser client-a
```

crea un usuario llamado client-a, cuyo directorio principal está en /www/client-a. Recuerde que makesite se ejecuta como makesite client-a.com y crea el directorio /www/client-a, por lo que el directorio principal del usuario coincide con su directorio Web. Cuando un usuario utiliza FPT para acceder a ftp.xcnews.com, automáticamente se inicia una sesión en su directorio principal (es decir, en el directorio superior del sitio Web).

El programa adduser crea por defecto un nuevo grupo por usuario y lo agrega al archivo /etc/group. Por lo tanto, habrá que cambiar los permisos del directorio principal. Por ejemplo, después de ejecutar el comando adduser client-a, necesitará ejecutar los siguientes comandos para determinar los permisos:

```
chown -R client-a.httpd /www/client-a/
chmod -R 750 /www/client-a/
```

El primer comando se asegura de que el directorio /www/client-a/ y todos sus subdirectorios pertenecen a client-a y a su grupo (que también se llama client-a). El segundo comando determina los permisos de todos los archivos y directorios de /www/client-a de forma que el propietario (client-a) tenga los permisos de lectura, escritura y ejecución. El grupo (httpd) es el único que tiene los permisos de ejecución y lectura. Tiene que contener el usuario de httpd, que utiliza rdist y cualquier cuenta de soporte técnico. Como nadie, a excepción del propietario y el grupo, puede acceder al directorio principal, un cliente no podrá ver el contenido del directorio principal de otro cliente a

través de FTP. Para determinar las contraseñas de cada usuario se emplea, como siempre, la utilidad usr/bin/passwd. Una vez que se cree una cuenta, tendrá que probar a través de FTP los permisos de carga y descarga.

Probar los nuevos sistemas

Llegados a este punto, hemos configurado y ejecutado todos los nuevos sistemas, por lo que comenzaremos con la fase de prueba. Para comprobar su funcionamiento desde un host remoto, tendrá que cambiar el nombre de éste por la dirección IP de ns.xcnews.com. Así podrá probar las peticiones que los host remotos dirigen a ns.xcnews.com., que tiene la configuración DNS correspondiente de los nuevos sitios. Los probadores continuarán con su labor sin afectar al contenido de estos sistemas.

Con la primera serie de pruebas utilizará herramientas TCP/IP como ping, nslookup y tracerouter. Por ejemplo, si utiliza nslookup desde la línea de comandos, los probadores verán si el servidor DNS ns.xcnews.com suministra la información adecuada tanto a .xcnews.com como a cualquiera de los sitios Web de sus clientes.

Hay que probar bastantes dominios, por lo que utilizaremos aquellos métodos que permitan reducir la interactividad humana. Creamos un archivo de texto con los nombres de los dominios:

```
xcnews.com
client-a.com
client b.com
...
...
client-z.org
```

Ahora se utilizará una serie de herramientas de UNIX para crear archivos batch basados en comandos de texto. Por ejemplo, para comprobar la información DNS de cada dominio, emplee comandos como éste:

```
cat ./domain-list.txt | awk '{printf("nslookup -query=ns -timeout=3
%s\n",$1);}' > ./ns_test
chmod 750 ./ns_test
./ns_test
```

La primera línea crea un archivo llamado ns_test, que contiene líneas como las siguientes:

```
nslookup -query=ns -timeout=3 xcnews.com
nslookup -query=ns -timeout=3 client-a.com
```

La segunda convierte el script ns_test en un script ejecutable y la tercera lo ejecuta. Así se prueba la información DNS de cada dominio. La primera línea se puede modificar para que pruebe los intercambios de registros de correo de cada dominio, para lo cual se cambia -query=ns a -query=mx:

```
cat ./domain-list.txt | awk '{printf("nslookup -query=mx -timeout=3
%s\n",$1);}' > ./mx_test
```

Ahora mx_test puede convertirse en un script ejecutable como ns_test y utilizarse para probar registros MX. Del mismo modo, se pueden ejecutar los script para comprobar los sitios Web a través de HTTP. Aquí tiene un ejemplo de uno de estos script:

```
cat ./domain-list.txt | awk '{printf("lynx -dump -head
www.%s\n",$1);}' > ./www_test
chmod 750 ./www_test
./www_test > /tmp/www_problems
```

Aquí, he utilizado un explorador Web en modo texto (lynx) para construir las cabeceras HEAD de las peticiones HTTP de cada host Web de mi dominio. Cuando se ejecuta el script www_text como se ha visto en el ejemplo anterior, se crea un archivo llamado www_problems en el directorio /tmp. Si lynx no puede acceder al sitio Web del host porque la configuración no es correcta, en el archivo www_problems aparecerá entonces un mensaje de error como éste:

```
lynx: Can't access start file http://www.some-client site.com
```

Como todos los sitios se han creado utilizando makesite, no habrá ningún error humano en la configuración.

Además, sería conveniente que efectuase estas pruebas a través de los exploradores Web. Como ninguno de estos sitios tienen ningún contenido real, simplemente mostrarán la página index.html que crea makesite para identificar a cada uno de ellos.

Cuando se termine con las pruebas anteriores habrá llegado el momento de transferir los archivos al nuevo sistema.

Puesta en marcha

Tan pronto como se muevan los archivos a la nueva zona de construcción, es posible que aparezcan problemas diferentes. Por ejemplo, muchas de las

páginas de XC News y de los clientes utilizan directrices SSI para ejecutar los script que mostrarán un cartel con anuncios. Aunque se configure Apache correctamente para trabajar con llamadas SSI, no aparecerán estos carteles, sino un mensaje de error.

El problema se debe a la forma en que Microsoft IIS Server ejecuta los script SSI. Una de estas llamadas que funciona correctamente en un servidor NT pero no en uno Apache es la siguiente:

```
<!-#exec cmd="perl.exe d:/var/www/xcnews/cgi-bin/rotate_banner_ad.pl
d:/var/www/xcnews/banners/giflist.txt" ->
```

Habrá que transformarla en:

```
<! #exec cmd " /www/xcnews/cgi-bin/rotate_banner_ad.pl /www/xcnews/
banners/giflist.txt" ->
```

Es posible que también tenga que actualizar un script en Perl llamado rotate_baner_ad.pl para que utilice #!ruta/a/perl como primera línea, algo que no hay que utilizar con NT. También hay que comprobar las posibles referencias a d:/var/www/ o cualquier otra ruta del tipo DOS. Una vez que se convierta la llamada SSI en una estándar, el servidor Apache podrá ejecutar el script.

Se pueden encontrar problemas parecidos con las llamadas SSI que utilicen la ruta del tipo DOS. Hay que traducir los nombres de las rutas al estilo UNIX. Como hay que corregir cientos de archivos, lo mejor que se puede hacer es escribir un script que automatice proceso, como el que aparece en el listado 19.14.

```
#!/bin/csh -f
#
# Este script se debería eliminar de los sitios AMLNEWS
# que se hayan movido de NT (BVS)
# Kabir

set thisdir = `/bin/pwd`;

echo "fixssi traducirá todas las llamadas SSI de NT dirigidas a
Apache."
echo "Directorio actual: $thisdir";
echo -n "Sustituir perl.exe d:/var/www/htdocs/cgi-bin :"

/usr/bin/find . -type f -exec /usr/local/bin/tcgrep "perl.exe
d:/var/www/htdocs/cgi-bin " "/www/xcnews/cgi-bin" {} \;

echo "done."
```

Listado 19.14. fixssi

Cuando se ejecuta fixssi desde /www, corrige todos los sitios que tengan llamadas SSI incorrectas.

Truco: fgres se trata de una utilidad que puede hacerse con la mayoría de archivos comp.unix.source que se puede localizar por medio de los motores de búsqueda.

Todos los script CGI se modifican para que utilicen #!/usr/bin/perl para invocar al intérprete de Perl. Todos los permisos de los archivos se configuran para que los procesos httpd de cada servidor Web puedan ejecutar los script CGI y escribir datos en el directorio /www/cgi-bin.

Con estos problemas resueltos, habrá llegado el momento de pasar de los sistemas antiguos a los nuevos que se encuentran en la red. Sólo para estar seguro, compruebe que los sitios Web están configurados correctamente y que todos los script funcionan como es debido. Hay que ajustarlos para que usen el demonio de correo sendmail. Una vez que se termine de probar los script CGI y otras líneas de contenido y todo funcione como es debido, el sistema estará en marcha.

Gracias a la actualización de las páginas Web de los archivos de registro del sistema DNS de InterNIC, se envía una petición para cambiar el nombre del servidor primario por el secundario de todos los dominios relacionados. En poco tiempo, se propagan todos los cambios DNS por Internet. Habrá veces en las que los clientes no puedan acceder a sus sitios Web, pero no se puede hacer nada para facilitar la propagación DNS por Internet. Después de un par de días, se completa la distribución y todos estos sitios se actualizarán y ejecutarán en los nuevos sistemas.

Posibilidades futuras

Es mejor utilizar el sistema DNS round-robin que nada. Pero la verdad es que no es la mejor solución de todas. Desgraciadamente, el resto de las soluciones son más caras que la que acabamos de implementar. Sus posibles desventajas son las siguientes:

- El sistema no puede detectar ningún fallo en el servidor, por lo que seguirá enviando peticiones de los clientes a un servidor que haya dejado de funcionar hasta que el administrador modifique manualmente el sistema DNS.

- Cuando se modifican los registros del sistema DNS, no se distribuyen en ese mismo momento porque la memoria caché de los servidores DNS reducen el ancho de banda. Es posible acelerar la propagación utilizando pequeños valores TTL, pero se puede provocar un embotellamiento en el tráfico de la red DNS.

Entonces, ¿qué se puede hacer para mejorar la situación? Desarrollar un interruptor encargado de repartir las cargas, como AceDirector, de Cisco, especializado en sistemas de redes. Recientemente están apareciendo nuevas soluciones de hardware capaces de atajar los problemas que acabamos de mencionar. Generalmente funcionan con redes donde los interruptores de hardware trabajan con estadísticas de carga en la red y otra información similar. Como este tipo de tecnología está empezando a ver la luz, su precio aún es elevado. Por eso es conveniente esperar hasta que bajen y utilizar estos componentes en el futuro.

20 Apache para Microsoft Windows 95/NT

Se ha desarrollado Apache para Microsoft Windows 95/NT para que se convierta en la distribución principal de este servidor. Por lo menos es lo quiere el Apache Group. Pero como es un reto muy grande, el grupo ya ha advertido que no es algo que vaya a ocurrir en poco tiempo.

Un usuario de Apache no puede esperar tanto tiempo. Instala la última versión beta en Windows y aprovecha esta oportunidad que le brinda Apache Group. Este grupo empresarial ha publicado la versión beta en la mayoría de los sitios Web de distribución de Apache, incluyendo el oficial de Apache.

En este capítulo veremos cómo se puede usar esta versión beta (1.3b3) para ver cómo será la versión de Windows. Tengo que advertirle que las versiones beta se consideran muy inestables y que sólo se tienen que utilizar para experimentar con ellas. No la use con los sitios Web de ninguna organización.

Puntos importantes relacionados con la implementación de Apache en Windows

Las tareas de importación suelen ser bastante complicadas. No están pensadas para los programadores de fin de semana. Y se van complicando cada

vez más según las diferencias que haya entre los sistemas operativos, como por ejemplo, las que hay entre UNIX y Windows.

Como Apache es una herencia de UNIX (se creó en esa plataforma), tiene todas las ventajas de los diseños de dicha arquitectura. Utiliza un modelo de servidor que se podría decir que es el estándar en el mundo de UNIX. El proceso del servidor principal apenas hace nada. Son los subprocesos los que realizan todo el trabajo duro. Por eso los subprocesos son programas tan complicados, aunque el código principal sea muy sencillo.

En Windows, los procesos del servidor se llaman thread (en UNIX, forks), por eso se le conoce como un sistema operativo multitarea. Un thread no puede separar procesos. Es una entidad ejecutable que trabaja dentro del espacio de proceso del disco duro y que puede efectuar una tarea independiente en cada thread.

Como en Windows no se puede trabajar con fork (propios de UNIX, como ya hemos comentado, capaces de separar procesos), Apache se tendrá que ejecutar en un único proceso que será quien se encargue de abrir los thread oportunos. Además, la tarea principal efectúa una serie de tareas antes de que todos los thread se hagan cargo de las peticiones. ¿Qué pasaría si le ocurriese algo a un thread? El desastre. Todos los threads restantes se finalizarían. En este caso, Apache cerraría el servicio Web. Se ha trabajado duramente para solventar el problema. La solución es que los procesos Apache adicionales se inician como servidores Apache que permanecen a la espera. Cuando finaliza el proceso de un servidor Web, uno de los servidores que están en estado de espera toma su lugar. De esta forma el servicio Web no desaparece del todo.

Otro de los requisitos relacionados de las aplicaciones multitarea es que los datos han de estar protegidos contra los thread. Algunos módulos Apache usan variables locales que no se pueden usar en la versión multitarea de Apache porque los thread pueden llegar a corromper los datos. Hay que poner un cuidado especial para asegurarse de que cada thread tiene su propia copia de las variables.

Aparte de los problemas relacionados con la multitarea, hay otra diferencia muy importante que sale a la luz. Windows carga sus módulos a través de los archivos DLL (Dynamically Linked Library), mientras que todos los módulos de UNIX se compilan en archivos ejecutables. Se tiene que escribir un módulo llamado mod_dll que se encargue de compilar el resto de módulos como archivos DLL. Tenga en cuenta que puede necesitar hacer más cosas para importar una aplicación de UNIX a Windows. El actual puerto no funciona como una aplicación nativa de Windows porque no tiene una interfaz gráfica para usuario (GUI). Aún es una aplicación que se ejecuta de la línea de comandos. Esto cambiará en versiones futuras.

Primero veremos qué tiene que ofrecernos. No olvide que es una versión experimental y que puede que no funcione exactamente como ha de hacerlo.

Conseguir Apache para Windows

Tiene dos formas para hacerse con una copia de Apache para Windows. Puede bajarse la distribución del código fuente o la binaria. A menos que sepa bastante del desarrollo Visual C++ o de Borland C++ bajo Windows, no conviene que se complique la vida trabajando con la distribución en código fuente. Hágase con ella sólo si tiene curiosidad por ver qué aspecto ofrece.

Le recomiendo que se haga con la distribución binaria de la última versión. Encontrará ambas distribuciones en las páginas Web de Apache:

```
www.apache.org
```

De todas formas, lo mejor que puede hacer es acercarse al mirror que tenga más cerca y bajarse el software. La distribución binaria no incluye ninguna documentación. Tendrá que bajársela por otro lado.

Instalación de Apache para Windows

El proceso de instalación es muy parecido al de cualquier software para Windows. Tiene que hacer lo siguiente:

1. Ejecute el archivo de descompresión automática bajado de la Red. Cuando inicie el ejecutable, aparecerá una ventana como la de la figura 20.1.

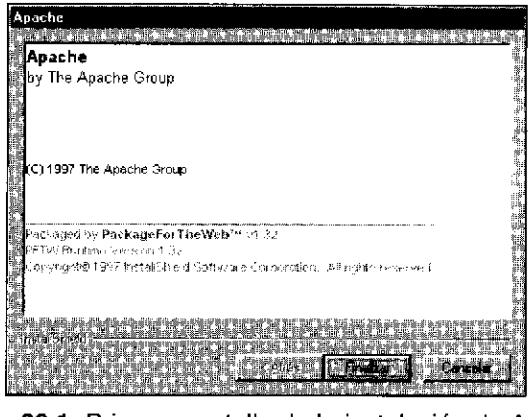


Figura 20.1. Primera pantalla de la instalación de Apache

En ella podrá ver el mensaje del Copyright diciéndole que el paquete se ha creado con el software InstallShield.

2. Haga clic en el botón Finalizar para iniciar la instalación del software. Verá entonces una pantalla de bienvenida como la que aparece en la figura 20.2.

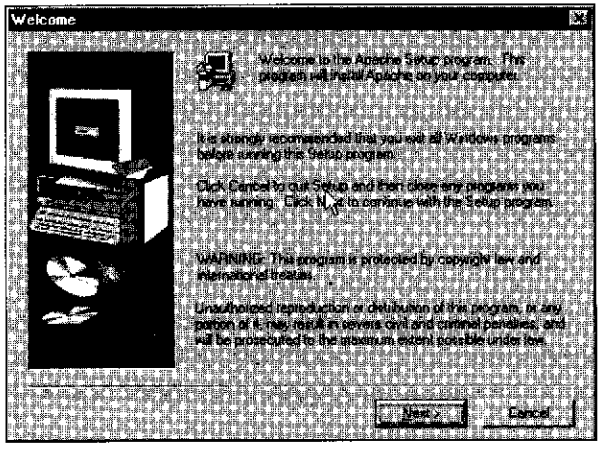


Figura 20.2. Pantalla de bienvenida

3. Haga clic en el botón Next (Siguiente, según versiones) y verá los términos de la licencia de Apache, que aparecen en la figura 20.3. Si está de acuerdo con estos términos, haga clic en el botón Yes (Sí, según versiones). En caso contrario, pulse No.

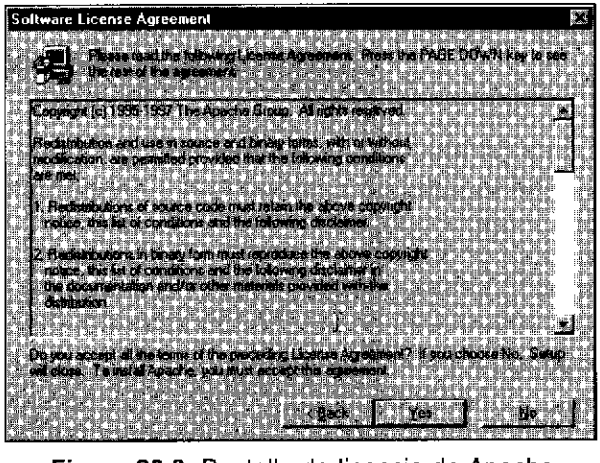


Figura 20.3. Pantalla de licencia de Apache

4. Suponiendo que contestase sí en la pantalla anterior, accederá al contenido de la figura 20.4. Desde esta pantalla podrá escoger si desea utilizar el directorio predeterminado (por ejemplo C:\Archivos de Programa\Apache) o bien seleccione la carpeta en la que desea guardar la instalación. Por defecto, los archivos de configuración de Apache para Windows se guardan en un directorio llamado \apache\conf, por lo que si escoge la instalación /Apache/, no tendrá que modificar nada. Para la instalación ejemplo que estamos viendo en este apartado usamos el directorio predeterminado. Cuando determine el directorio de instalación, pulse Next.
5. A continuación se le pedirá que escoja el tipo de configuración. Tiene la ventana en la figura 20.5.

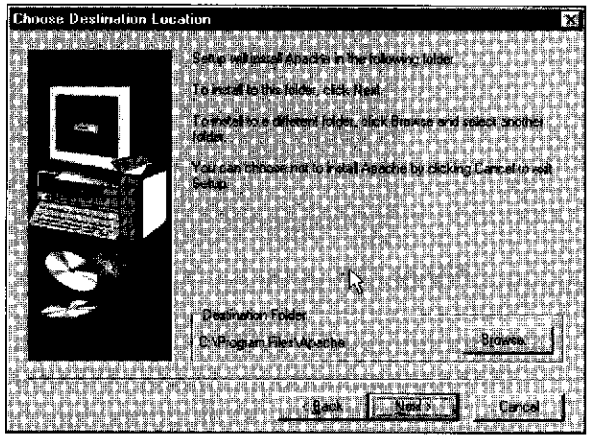


Figura 20.4. Directorio de instalación de Apache

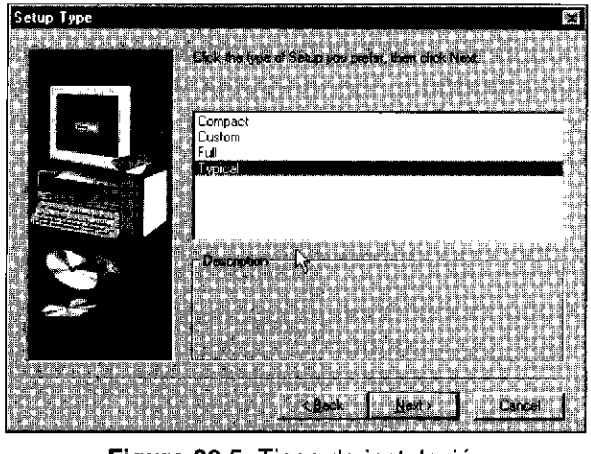


Figura 20.5. Tipos de instalación

Seleccione el tipo de instalación que desea utilizar. He escogido la opción "custom" (personalizada) para ver el resto de opciones de configuración. Todo lo que se me pide es que escoja un directorio. Después de escoger el tipo de configuración, el asistente de instalación copiará los archivos en la ubicación apropiada. Cuando finalice, le mostrará la ventana que aparece en la figura 20.6.

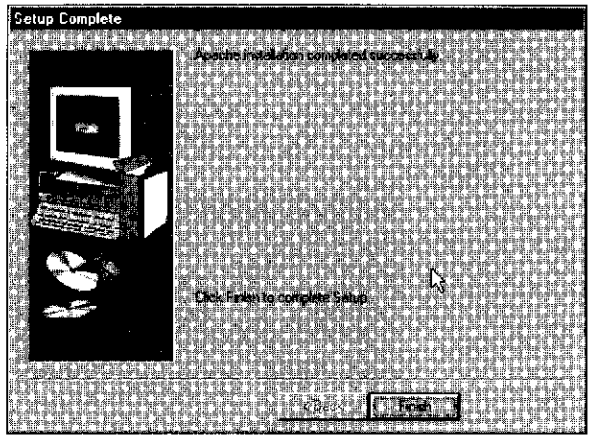


Figura 20.6. Aviso que le indica que ha finalizado la instalación

6. Si decide instalar la documentación, asegúrese de que la instala en el directorio principal de Apache. En otras palabras, si ha instalado Apache en C:\Archivos de programa\Apache, instale la documentación en el mismo directorio. El asistente encargado de la instalación de la documentación le pedirá que introduzca el nombre de la ruta, tal y como se ve en la figura 20.7.
   Si ha completado la instalación del archivo binario, estará listo para ejecutarlo. Pero tenga en cuenta que, dependiendo de la versión que tenga de Windows, tendrá que hacerlo de una forma o de otra.

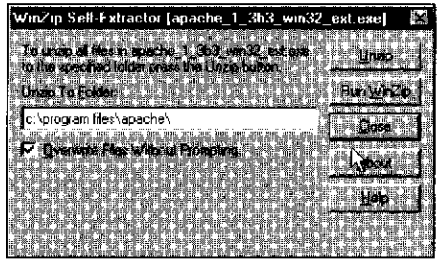


Figura 20.7. Instalación de la documentación de Apache

Ejecutar Apache en Windows 95

En Windows 95, Apache se ejecuta como una aplicación de la consola. En otras palabras, se puede acceder a ella desde el Explorador de Windows (no desde Internet Explorer) o bien ejecutándola a través de una ventana de DOS, como se puede ver en la figura 20.8.

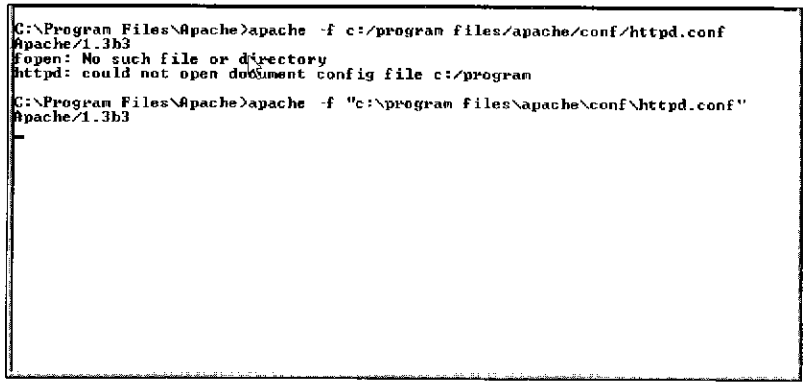


Figura 20.8. Ejecución de Apache en Windows 95

Cuando escribo la siguiente sentencia para abrir el ejecutable de Apache desde C:\Archivos de programa\Apache:

```
Apache -f  c:/Archivos de programa/apache/conf/httpd.conf
```

veremos que no funciona, pero cuando escribo:

```
Apache -f  "c:\Archivos de programa\apache\conf\httpd.conf"
```

sí. La diferencia es que las comillas dobles que he utilizado en esta versión de Apache no se confunde con la ruta en la que se utilizan espacios en blanco. Además he utilizado la barra inclinada propia de UNIX en vez de la barra invertida propia de Windows. De todas formas, con mi prueba veo que también funciona lo siguiente:

```
Apache -f  "c:/Archivos de programa/apache/conf/httpd.conf"
```

Como se puede ver, la barra invertida también funciona desde la línea de comandos. Pero en todos los archivos de configuración (httpd.conf, srm.conf y access.conf) tendrá que utilizar la barra inclinada propia de los sistemas UNIX. Cuando se ejecuta Apache de esta forma, verá que no muestra ninguna información adicional aparte de su número de versión.

Truco: También puede iniciar el servidor desde el menú Inicio por medio del icono que se ha añadido durante el proceso de instalación.

Si quiere que el servidor se inicie durante el arranque del ordenador, tendrá que colocar una copia del icono de Apache que ha creado el asistente de instalación en el grupo Inicio, Programas, Inicio. Se abrirá automáticamente una ventana DOS con el servidor Apache durante el arranque del ordenador. Esta ventana se puede minimizar e ignorar (desde una de las opciones del cuadro de propiedades de la ventana DOS).

Si quiere detener el servidor, tendrá que cerrar la ventana DOS o pulsar Ctrl+C. Aún no se ha desarrollado una forma mejor para detenerlo.

Ejecutar Apache como un servicio en Windows NT

Puede ejecutar Apache desde Windows NT exactamente igual que desde Windows 95. Pero Apache se ejecutará como un servicio de Windows NT 4.0 (superior), utilizando el comando:

```
Apache -i
```

Si ha instalado los archivos Apache en un directorio que no sea /Apache, tendrá que usar la opción -d para especificar la nueva ubicación. Por ejemplo:

```
Apache -i  d  "c:/Archivos de programa/apache"
```

De esta forma se instalará Apache como un servicio de NT y se le indicará que busque los archivos de configuración que se encuentran en el directorio C:\Archivos de programa\Apache. Una vez que tenga instalado el servicio, puede controlar su ejecución desde la ventana de Servicios, cuyo contenido se muestra en la figura 20.9.

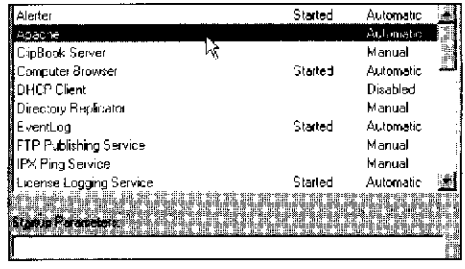


Figura 20.9. Apache se ejecuta como un servicio de Windows NT

Por defecto, la configuración de Apache hace que se inicie automáticamente durante el arranque de NT. De todas formas, si lo quiere ejecutar ahora, seleccione el servicio Apache de la lista y haga clic en el botón Iniciar (Start).

Si quiere eliminar Apache de los servicios de NT, utilice el comando:

```
Apache -u -d "c:/Archivos de programas/apache"
```

Así lo eliminará de la lista de servicios. Este comando asume que el directorio en el que se ha instalado Apache es C:/Archivos de programas/apache. Si personalizó las opciones de la instalación, utilice el de su sistema.

Cuando tenga Apache funcionando bajo Windows 95/NT, puede utilizar el Administrador de Tareas para comprobarlo. En la figura 20.10 se puede ver su contenido.

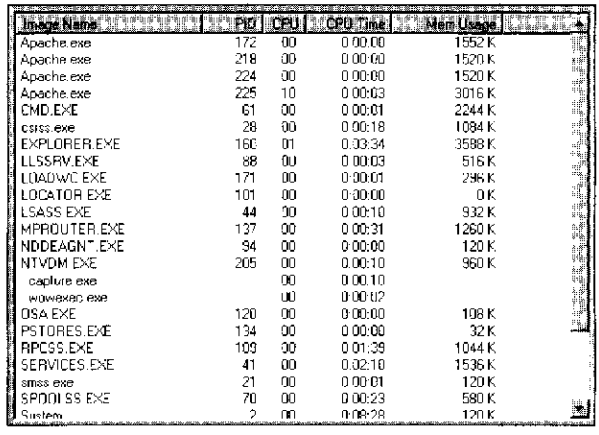


Image Name	PID	CPU	CPU Time	Mem Usage
Apache.exe	172	00	0:00:00	1552 K
Apache.exe	218	00	0:00:00	1520 K
Apache.exe	224	00	0:00:00	1520 K
Apache.exe	225	10	0:00:03	3016 K
CMD.EXE	61	00	0:00:01	2244 K
csrss.exe	28	00	0:00:18	1084 K
EXPLORER.EXE	160	01	0:33:34	3588 K
LLSSRV.EXE	88	00	0:00:03	516 K
LOADWC.EXE	171	00	0:00:01	296 K
LOCATOR.EXE	101	00	0:00:00	0 K
LSASS.EXE	44	00	0:00:10	932 K
MPROUTER.EXE	137	00	0:00:31	1260 K
NDDEAGNT.EXE	94	00	0:00:00	120 K
NTVDM.EXE	205	00	0:00:10	960 K
capture.exe		00	0:00:10	
wowexec.exe		00	0:00:02	
OSA.EXE	120	00	0:00:00	108 K
PSTORES.EXE	134	00	0:00:00	32 K
RPCSS.EXE	109	00	0:01:39	1044 K
SERVICES.EXE	41	00	0:02:10	1536 K
smss.exe	21	00	0:00:01	120 K
SPOOLSS.EXE	70	00	0:00:23	580 K
System	2	01	0:08:28	120 K

Figura 20.10. El servidor Apache aparece en el Administrador de tareas

Como se puede observar, se está ejecutando un proceso de un servidor Apache. El resto está en modo de espera por si se finaliza alguno de los que están en ejecución. Puede controlar la forma en la que se abren los procesos por medio de las opciones de configuración que se ven a continuación.

Configuración de Apache para Windows

La configuración de Apache para Windows es muy parecida a la configuración para sistemas UNIX, a excepción de lo siguiente.

Cualquier directriz que utilice la información de la ruta tendrá que utilizar el formado disco:/ruta/a/algo en vez de /ruta/a/algo que es el que se usa en

UNIX. Recuerde que tiene que utilizar las barras inclinadas de UNIX en los nombres de las rutas. Por ejemplo, se puede utilizar C:/www/misitio, pero no C:\www.misitio. Si no especifica la unidad de disco, se utilizará por defecto aquella en la que se haya instalado Apache.

Aparte de las diferencias sobre la ruta, hay otros cambios. Algunas directrices antiguas tienen otro significado y aparecen directrices nuevas. Las vemos en la sección siguiente.

Directrices de Apache específicas de Windows

La versión de Apache para Windows tiene un módulo especial llamado mod_dll. Se utiliza para cargar los módulos externos que no se han compilado dentro del servidor Apache. Con el siguiente comando accederá a la lista de los módulos compilados con el servidor:

```
Apache -l
```

La versión actual de Apache cuenta con los siguientes módulos:

- http_core.c
- mod_dll.c
- mod_mime.c
- mod_access.c
- mod_auth.c
- mod_negotiation.c
- mod_include.c
- mod_autoindex.c
- mod_dir.c
- mod_cgi.c
- mod_userdir.c
- mod_alias.c
- mod_env.c
- mod_log_config.c
- mod_asis.c

- mod_imap.c
- mod_actions.c
- mod_setenvif.c
- mod_isapi.c

El módulo mod_dll se encuentra dentro del archivo binario. Es la única forma de que funcionen. Pero hay otros módulos que se pueden encontrar en el subdirectorio del servidor Apache. Son los siguientes:

- ApacheModuleAuthAnon.dll
- ApacheModuleCERNMeta.dll
- ApacheModuleDigest.dll
- ApacheModuleExpires.dll
- ApacheModuleHeaders.dll
- ApacheModuleRewrite.dll
- ApacheModuleStatus.dll
- ApacheModuleUserTrack.dll

Si quiere utilizar cualquiera de estos módulos con su servidor Apache, necesitará mod_dll y las directrices que se muestran en las secciones siguientes.

Aparte de las nuevas propiedades de mod_dll, hay algunas directrices antiguas cuyo significado ha cambiado levemente. Exceptuando las directrices que se van a mencionar aquí, el resto deberían funcionar correctamente. De todas formas, no se olvide que es una versión beta, por lo que es posible que aún no se hayan incluido algunas directrices.

Aquellas a las que se les ha modificado su significado son las siguientes:

- StartServers. Esta directriz especifica el número de servidores en estado de espera que iniciará Apache en el caso en que se cierren los ya existentes por una finalización accidentada del servidor. Si abre demasiados, estará desperdiciando memoria. Se recomienda trabajar con el valor predeterminado.
- MinSpareServers. Como sólo se utiliza un servidor (los otros se encuentran en modo de espera), en Windows no hay que utilizarla.
- MaxSpareServers. Como sólo se utiliza un servidor (los otros se encuentran en modo de espera), en Windows no hay que utilizarla.

- MaxRequestsPerChild. Bajo Windows, un único proceso es el encargado de controlar todas las peticiones especificadas en esta directriz. Si se utiliza un número demasiado bajo, el servidor finalizará. De todas formas, si utiliza el valor 0, el servidor nunca finalizará.

Vamos a ver las directrices de mod_dll.

LoadModule

```
Sintaxis: LoadModule nombre_módulo ruta_a_módulo_dll
Contexto: configuración servidor
```

Esta directriz le permite cargar un archivo DLL en un módulo de Apache durante el inicio. Por ejemplo:

```
LoadModule   mod_digest  modules/ApacheModuleDigest.dll
```

Cargará el archivo modules/ApacheModuleDigest.dll como si fuese el módulo mod_digest. Si la ruta no comienza por /, se supondrá que es un subdirectorio del directorio principal del servidor.

LoadFile

```
Sintaxis: LoadFile nombre_archivo ruta_a_archivo_dll
Contexto: configuración servidor
```

Esta directriz le permite cargar un archivo DLL que necesite un módulo DLL. Si la ruta no comienza por /, se supondrá que es un subdirectorio del directorio principal del servidor.

ThreadsPerChild

```
Sintaxis: ThreadsPerChild número_thread
Predeterminado: ThreadsPerChild 50
Contexto: configuración servidor
```

Esta es una de las novedades. Determina el número total de thread que podrá usar la versión de Apache para Windows. Como cada thread sólo puede atender a un proceso, con esta directriz se limita la cantidad de peticiones de las que se hará cargo el servidor.

A Códigos de estado de HTTP/1.1

Por cada petición recibida de un cliente Web, el servidor tiene que devolver un código de estado HTTP, que consiste en un número de tres cifras. Un cliente Web tratará de comprender la respuesta del servidor observando el código de estado que le ha devuelto en la cabecera HTTP Status-Line. Al código le acompaña una pequeña frase con la que se intenta ofrecer una breve explicación al usuario. Por ejemplo, sea el código:

```
HTTP/1.1 404 Not Found
```

"404" es el código de estado y "Not Found" es la explicación. En un cliente Web, el explorador mostraría en la pantalla "Not Found". En la versión 1.1 de HTTP hay cinco tipos de código disponibles, que son los que veremos en las siguientes secciones.

Códigos de estado informativos (100-199)

La finalidad de estos códigos es permitir que el cliente sepa que el servidor está procesando una petición. Estos códigos de estado son meramente informativos. El cliente no tiene que hacer nada cuando los recibe. En la

versión 1.0 de HTTP no se definen estos tipos de código de estado, por lo que no se deberán enviar a clientes HTTP/1.0. En la actualidad, estos códigos son los siguientes:

Código	Descripción
`100 Continue`	El servidor envía este código para que el cliente sepa que está listo para recibir peticiones.
`101 Switching Protocols`	El servidor envía este código cuando quiere cambiar el protocolo de la aplicación al definido en la petición Upgrade, que le envía el cliente. Este cambio solamente tendrá lugar si el nuevo protocolo tiene más ventajas que el que se está utilizando. Por ejemplo, si el cliente le pide al servidor que utiliza un protocolo HTTP más moderno que el que tiene en uso, el servidor tratará de hacer el cambio.

Recibida satisfactoriamente la petición del cliente (200-299)

Si el servidor envía un código comprendido entre 200-299, estará indicando que ha recibido y aceptado la petición sin ningún problema. Estos códigos son los siguientes:

Código	Descripción
`200 OK`	El servidor ha procesado la petición y se anexa el documento solicitado.
`201 Created`	El servidor ha creado el URI que se especifica en la cabecera Location.
`202 Accepted`	Se ha aceptado la petición y se está procesando, pero aún no ha terminado con ella.
`203 Non-Authoritative Information`	La metainformación que se encuentra en la cabecera de la respuesta no la ha generado el servidor. Se copió de otro servidor.
`204 No Content`	Petición completa, pero no hace falta que se envíe más información. El cliente continuará con la visualización del documento.

Código	Descripción
205 Reset Content	El cliente tiene que reiniciar el documento actual. Es útil cuando hay que limpiar todos los campos de un formulario.
206 Partial Content	El servidor ha completado la petición GET efectuada por la fuente. Este código es utilizado para responder a las peticiones de rango. El servidor envía una cabecera Content-Range para indicar qué segmento de datos ha sido anexado.

Redirigir la petición (300-399)

Los códigos de estado comprendidos entre 300 y 399 se le envían al cliente cuando se le quiere indicar que ha de efectuar otra acción más para completar la petición. Estos códigos son los siguientes:

Código	Descripción
300 Multiple Choices	La fuente solicitada se corresponde con un conjunto de documentos. El servidor puede enviar información sobre cada uno de los documentos dentro de su propia ubicación y en información de la negociación para permitir que el cliente escoja una.
301 Moved Permanently	La fuente solicitada no se encuentra en el servidor. Se envía una cabecera Location para redirigir al cliente hacia el nuevo URL. El cliente redirigirá las futuras peticiones a la nueva dirección.
302 Moved Temporarily	La fuente solicitada se ha movido temporalmente. Se envía una cabecera Location para redirigir al cliente hacia el nuevo URL. El cliente redirigirá las futuras peticiones a la nueva dirección.
303 See Other	Se ha encontrado la fuente solicitada en una ubicación diferente a la indicada en la cabecera Location y el cliente tiene que usar el método GET para recuperarla.
304 Not Modified	El servidor utiliza este código para responder a una cabecera If-Modified-Since. De esta manera se indica

Código	Descripción
	que el documento no se ha modificado desde una fecha determinada y que el cliente puede utilizar la copia que guarda en la memoria caché.
305 Use Proxy	El cliente tiene que utilizar el proxy que hay especificado en la cabecera Location para recuperar la fuente solicitada.

Peticiones incompletas (400-499)

Los códigos de estado comprendidos entre 400-499 se utilizan para indicarle al cliente que su petición no se ha podido completar y que tiene que enviar más información. Los códigos son los siguientes:

Código	Descripción
400 Bad Request	El servidor ha detectado un error de sintaxis en la petición del cliente.
401 Unauthorized	La petición necesita la autenticación del cliente. El servidor envía una cabecera WWW-Authenticate para indicar el tipo de autenticación y su alcance para la fuente solicitada.
402 Payment Required	Este código está reservado para utilizarlo en el futuro.
403 Forbidden	Se prohibe el acceso a la fuente solicitada. El cliente no tiene que repetir la petición.
404 Not Found	El documento solicitado no se encuentra en el servidor.
405 Method Not Allowed	El método que utiliza el cliente en su petición es inaceptable. El servidor envía una cabecera Allow indicando qué métodos se pueden usar para acceder a la fuente solicitada.
406 Not Acceptable	La fuente solicitada no se encuentra en un formato que pueda aceptar el cliente (basándose en las cabeceras que puede recibir el servidor). Si la petición no es HEAD, el servidor podrá enviar entonces cabeceras Content-Language, Content-Encoding y

Código	Descripción
	Content-Type para indicar los formatos que hay disponibles.
407 Proxy Authentication Required	Acceso no autorizado para la petición en el servidor proxy. Lo primero que tiene que hacer el cliente es autenticarse al proxy. El servidor envía la autenticación y su alcance para la fuente solicitada.
408 Request Time-Out	El cliente no ha podido completar su petición dentro del periodo de tiempo que le facilita el servidor. Tendrá que volver a repetirla.
409 Conflict	La petición del cliente entra en conflicto con otra petición. El servidor puede añadir información sobre el tipo de conflicto junto al código de estado.
410 Gone	La fuente solicitada se ha eliminado permanentemente del servidor.
411 Length Required	El cliente tiene que suministrar una cabecera Content-Type en la petición.
412 Precondition Failed	Cuando un cliente envía una petición con una o más cabeceras condicionales, el servidor utiliza este código para indicar que una o más de las condiciones especificadas en la cabecera es falsa.
413 Request Entity Too Large	El servidor no quiere procesar la petición porque el cuerpo del mensaje es demasiado largo. El servidor cerrará la conexión para detener la entrada de la petición.
414 Request-URI Too Long	El servidor no quiere procesar la petición porque el URI es demasiado largo.
415 Unsupported Media Type	El servidor no quiere procesar la petición puesto que no puede trabajar con el formato del cuerpo del mensaje.

Errores del servidor (500-599)

Los códigos de estado comprendidos entre 500-599 se útilizan cuando el servidor se encuentra con un error por culpa del cual no puede completar la petición. Estos códigos son los siguientes:

Código	Descripción
`500 Internal Server Error`	La configuración del servidor o de un programa externo ha producido un error.
`501 Not Implemented`	El servidor no puede trabajar con la funcionalidad necesaria para completar la petición.
`502 Bad Gateway`	El servidor ha encontrado una respuesta no válida de un servidor o un proxy.
`503 Service Unavailable`	El servicio está temporalmente fuera de servicio. Puede enviar una cabecera Retry-After para indicar cuándo volverá a estar disponible.
`504 Gateway Time-Out`	La puerta de enlace (gateway) o el proxy ha caducado.
`505 HTTP Version Not Supported`	No se admite la versión de HTTP que utiliza el cliente.

B Expresiones regulares

Una expresión regular se compone de un carácter normal y otro especial, con cuya combinación se crea un modelo. Con éste se trata de marcar una o más subcadenas de una cadena completa. Por ejemplo:

```
([a-z]+)\.([a-z]):\.([a-z]+)
```

Con esta expresión regular coinciden www.idgbooks.com, www.apache.org, etc. Los caracteres especiales que se utilizan con las expresiones regulares reciben el nombre de *metacaracteres*. Los que aparecen a continuación son los más utilizados:

Metacaracter	Descripción
.	Marca cualquier carácter (exceptuando el de nueva línea).
^	Marca el inicio de una cadena.
$	Marca el final de una cadena.
\b	Marca el límite de una palabra.
x?	Marca 0 ó 1 x, donde x es una expresión regular.

Metacaracter	Descripción
x*	Marca 0 o más x.
x+	Marca 1 o más x.
foo|bar	Marca un foo o una barra
[abc]	Marca cualquier carácter del conjunto abc.
[A-Z]	Marca cualquier carácter comprendido entre A-Z.
[^xyz]	Marca cualquier carácter que no esté incluido en el conjunto xyz.
\w	Marca un carácter alfanumérico (como por ejemplo [a-zA-Z0-9_]).
\s	Marca un espacio en blanco.
\t	Carácter de tabulación.
\n	Carácter de nueva línea.
\r	Carácter de retorno.
\v	Tabulación vertical.
\a	Carácter de campana.
\e	Carácter Escape.
\077	Carácter octal.
\x9f	Carácter hexadecimal.
\c[	Carácter de control.
\l	Carácter "siguiente en minúscula".
\L	Minúscula hasta \E.
\U	Mayúscula hasta \E.
\E	Modificación final.
\u	Carácter "siguiente en mayúscula".
\Q	Coloca comillas en los metacaracteres hasta que llega a \E.

Si tiene que utilizar un metacaracter en expresiones regulares puede utilizar el formato /metachar para eliminar su significado especial. Un ejemplo lo tenemos en \$, que indica simplemente $.

Los identificadores estándar que se utilizan con las expresiones regulares son los siguientes:

Identificador	Descripción
*	Marca 0 o más veces.
+	Marca 1 o más veces.
?	Marca 0 o 1 veces.
{n}	Marca exactamente n veces.
{n,}	Marca por lo menos n veces.
{n, m}	Marca por lo menos n veces, pero nunca más de m.

Al carácter se le trata como si fuese un operador OR. Un par de paréntesis ( ) le permite agrupar los caracteres en una expresión regular. Un par de paréntesis cuadrados [ ] crea un rango o clase de caracteres.

Volvamos a nuestro primer ejemplo:

```
([a-z]+)\.([a-z]+)\.([a-z]+)
```

Como ya hemos comentado, esta expresión se puede utilizar para marcar cadenas como www.idgbooks.com. El primer [a-z]+ indica que hay que marcar uno o más caracteres comprendidos entre a-z especificado en el primer paréntesis. Si no se marca ningún carácter, se podrá especificar todo lo que se desee con $1. En esta expresión hay tres pares de paréntesis. El primero (empezando por la izquierda) es $1, el segundo es $2 y el tercero $3. Obsérvese que se utiliza \ para colocar un punto entre los dos grupos de metacaracteres.

Dos ejemplos más:

- `^foo\.htm$`

  Marcará una cadena foo.htm, pero no una afoo.htm porque se ha utilizado el metacaracter ^ para especificar que la cadena tiene que empezar con el carácter f. tampoco marcará foo.htmf porque el metacaracter $ se utiliza para especificar que la cadena tiene que terminar con una m.

- `^www\.([^.]+)\.host\.com(.*)`

  Marcará una cadena, como www.username.host.com STATUS=java y asignará $1 al host y $2 a todo lo que siga a www.username.host.com. $2 será STATUS=java.

C Fuentes de Internet para Apache

En este apéndice encontrará una lista de direcciones de sitios Web, grupos de news de Usenet y listas de mail que le resultarán de utilidad.

Fuentes gratuitas

Las que aparecen a continuación son algunas de las fuentes gratuitas de Apache en Internet.

Sitios Web

Sitio Web Oficial de Apache: `www.apache.org`

Apache Module Registry: `http://modules.apache.org/`

Apache-SSL: `http://www.apache-ssl.org/`

Apache/Perl Integration Project: `http://perl.apache.org/`

Apache GUI Project: `http://butler.disa.mil/ApacheConfig/`

Java-Apache Project: `http://java.apache.org/`

Apache para OS/2: `http://www.slink.com/ApacheOS2/`

Apache para Amiga: `http://www.xs4all.nl/~albertv/apache/`

Grupos de news de Usenet

También encontrará información de temas relacionados con la World Wide Web en 15 grupos de news distintos. Utilice el grupo que mejor encaje con lo que busca. Antes de nada, asegúrese de que publica el mensaje en el grupo correcto y, siempre que sea posible, lea el apartado de preguntas y respuestas más comunes (FAQ).

Grupos de news relacionados con los servidores Web

```
comp.infosystems.www.servers.unix
```

En este grupo de news se tratan temas relacionados con los servidores para las plataformas UNIX. Entre los temas que se tratan encontrará preguntas y respuestas, asuntos relacionados con la seguridad, estructura de directorios e informe de fallos.

```
comp.infosystems.www.servers.ms-windows
```

En este grupo se habla de los servidores Web para MS Windows y plataformas NT. Entre los temas que se tratan tenemos la configuración, preguntas y respuestas, asuntos relacionados con la seguridad, estructura de directorios e informe de fallos.

```
comp.infosystems.www.servers.mac
```

Grupo de news en el que se habla de los servidores Web para Macintosh (MacOS). Entre los temas que se tratan tenemos la configuración, preguntas y respuestas, asuntos relacionados con la seguridad, estructura de directorios e informe de fallos.

```
comp.infosystems.www.servers.misc
```

Se habla de los servidores Web para otras plataformas como Amiga, VMS, etc. Entre los temas que se tratan tenemos la configuración, preguntas y respuestas, asuntos relacionados con la seguridad, estructura de directorios e informe de fallos.

```
japan.www.server.apache
```

En este grupo de news se habla de los servidores Apache en japonés.

Grupos de news relacionados con los autores

```
comp.infosystems.www.authoring.cgi
```

En este grupo se habla de los script CGI y de su creación para que trabajen con las páginas Web. Entre los posibles proyectos nos encontramos: cómo manipular el resultado de los formularios, cómo generar imágenes sobre la marcha y cómo conjuntar otras ofertas interactivas.

```
comp.infosystems.www.authoring.html
```

Este grupo habla del lenguaje HTML y de su relación con la creación de páginas Web. Entre los temas que trata tenemos los editores HTML, trucos para dar formato y los estándares HTML.

```
comp.infosystems.www.authoring.images
```

Se habla de la creación y edición de imágenes desde el punto de vista de la creación de páginas Web. Entre los temas que trata tenemos cómo mejorar las capacidades de la Web para representar imágenes y las preguntas y respuestas más comunes para trabajar con los mapas de imágenes.

```
comp.infosystems.www.authoring.misc
```

Aquí se habla de varios temas sobre la autoría en la Red que no se ven en el grupo anterior. Entre los asuntos que se tratan tenemos el audio y el vídeo.

Grupos de news relacionados con los exploradores Web

```
comp.infosystems.www.browsers.ms-windows
```

Aquí se habla de los exploradores Web para MS Windows y NT. Entre los temas tratados tenemos preguntas y respuestas relativas a la configuración, visualizadores externos (aplicaciones de ayuda) e informe de fallos.

```
comp.infosystems.www.browsers.mac
```

Aquí se habla de los exploradores Web para Macintosh. Entre los temas tratados tenemos preguntas y respuestas relativas a la configuración, visualizadores externos (aplicaciones de ayuda) e informe de fallos.

```
comp.infosystems.www.browsers.x
```

Aquí se habla de los exploradores Web para sistemas X-Windows. Entre los temas que son tratados aquí tenemos preguntas y respuestas relativas a la

configuración, visualizadores externos (aplicaciones de ayuda) e informe de fallos.

```
comp.infosystems.www.browsers.misc
```

Aquí se habla de los exploradores Web para otras plataformas (Amiga, DOS, VMS y UNIX en modo texto). Entre los temas tratados tenemos preguntas y respuestas relativas a la configuración, visualizadores externos (aplicaciones de ayuda) e informe de fallos.

Grupos de news de anuncios

```
comp.infosystems.www.announce
```

En este grupo de news se tratan novedades relacionadas con la Red. Antes de publicar ningún mensaje, LEA EL CONTENIDO YA EXISTENTE.

Otros grupos de news

```
comp.infosystems.www.advocacy
```

Grupo dedicado a comentarios, argumentos y debates sobre los exploradores y servidores Web, programas externos de visualización o de si un programa es mejor o peor que otro.

```
comp.infosystems.www.misc
```

Este grupo proporciona un foro para tratar temas generales relacionados con la Web que no se ven en ninguno de los grupos anteriores. Aquí encontrará los últimos rumores sobre los futuros cambios que afectarán a la estructura y los protocolos de la Red que afectan tanto a servidores como a clientes.

Grupos de news sobre Perl

```
comp.lang.perl.misc
```

Estos grupos están especializados en Perl. Hablan desde todos los informes de fallos que se generan hasta las nuevas propiedades, pasando por algo de historia, humor y trivialidades. Es la mejor fuente de información para cualquier tema relacionado con Perl, sobre todo con las novedades de Perl5.

```
comp.lang.perl.announce
```

Aquí se anuncian las nuevas versiones, preguntas y respuestas más frecuentes y módulos.

Fuentes WWW para los grupos de news de Usenet

DejaNews: `www.dejanews.com/`

The Reference Archive: `www.reference.com/`

American Web Services: `www.awebs.com/news_archive`

Critical Mass Communications: `http://www.criticalmass.com/concord/index.htm`

Listas de mail

Si quiere recibir información sobre Apache en cualquier momento, se puede suscribir a las listas de mail. Para ello deberá enviar un mensaje a apache-announce-request@apache.org con la palabra "suscribe" en el cuerpo del mensaje. Recuerde que no es un foro al que se le pueda formular preguntas.

Para suscribirse a un grupo más general, envíe el comando "subscribe apache" a majordomo@geek.net. Hay una versión resumida de la lista a la que se podrá suscribir con el comando "subscribe apache-digest".

Fuentes comerciales

Cada vez son más los usuarios de Apache que buscan fuentes comerciales que ofrezcan software o servicios relacionado con este servidor. Las que le mostramos a continuación son las más famosas:

Revista online gratuita The Apache Week: `www.apacheweek.com`

US/Canada Stronghold: `www.c2.net`

Para fuera de US/Canada Stronghold: `www.uk.web.com`

Covalent Raven: `http://raven.covalent.net`

Rovis: `www.rovis.com/warpaint/`

Otras fuentes relacionadas

WWW Consortium: `www.w3.org/`

Sitio Web de Netcraft Survey Report: `www.netcraft.co.uk/Survey/`

Todos los documentos RFC: `http://ds.internic.net/ds/dspg-1intdoc.html`

Server Watch: `www.ServerWatch.com`

Search Engine Watch: `www.SearchEngineWatch.com`

Browser Watch: `www.BrowserWatch.com`

Web Compare: `www.WebCompare.com`

Web Developer: `www.WebDeveloper.com`

Web Reference: `www.WebReference.com`

Comercio Electrónico en Internet: `http://e-comm.internet.com`

The List: `www.TheList.com`

Internet News: `www.InternetNews.com`

Especificación CGI: `http://hoohoo.ncsa.uiuc.edu/cgi/interface.html`

FastCGI: `http://www.fastcgi.com`

Sitio del lenguaje Perl: `www.perl.com`