Instalación y configuración de *Apache*, un servidor Web gratis

José Márquez Díaz[*], Leonardo Sampedro[**], Félix Vargas[***]

Resumen

El servidor web Apache se ha convertido en el servidor web más utilizado en el mundo debido a sus altas prestaciones y desempeño, además de ser gratuito, lo cual contribuye a su rápida expansión y posicionamiento. La configuración de este servidor web para aquellas personas que posean un conocimiento medio del sistema operativo Linux no debe ser un problema, pero resulta en ocasiones complicado e intimidante enfrentarse a los archivos de configuración del servidor sin una guía o con la base de la información fragmentada y de lenguaje oscuro que se puede obtener en la web.

El propósito de este artículo es ayudar a aquellas personas que deseen configurar un servidor HTTP a cumplir su objetivo de una manera sencilla y rápida para lograr una excelente configuración y puesta en marcha de alto rendimiento.

Palabras clave: http, web, host virtuales, autenticación, página de inicio.

Abstract

The Apache Web Server has became in the most used web server in the world, because of its features and performance; besides it is free, which contribute to its fast expansion and positioning. The web server configuration is not a problem for the people who has a media level knowledge about Linux Operating System; however, to confront the server configuration without a guide or based on the information from the web about a dark language, this task would be complicated and sometimes intimidatory. The purpose of this article is to help to the people, who want to configure a HTTP server, to reach their objective, in a fast and simple way, getting a excellent configuration with a high performance.

Key words: http, web, virtual hosts, authentication, home page.

Fecha de recepción: 28 de febrero del 2002

* Ingeniero de Sistemas de la Universidad del Norte; candidato a Maestría en Ciencias Computacionales del convenio ITESM-CUTB. Docente tiempo completo del Departamento de Sistemas de la Universidad del Norte *(jmarquez@uninorte.edu.co)*

** Estudiante de último semestre del programa de Ingeniería de Sistemas de la Universidad del Norte.

*** Ingeniero de Sistemas de la Universidad del Norte.

1. INTRODUCCIÓN

El servidor web Apache es un servidor Web gratuito desarrollado por el Apache Server Project (Proyecto Servidor Apache) cuyo objetivo es la creación de un servidor web fiable, eficiente y fácilmente extensible con código fuente abierto gratuito. Este proyecto es conjuntamente manejado por un grupo de voluntarios localizados alrededor del mundo que a través de Internet planean y desarrollan el servidor y la documentación relacionada con éste. Estos voluntarios son conocidos como el grupo Apache.

En febrero de 1995, el software de servidor más popular en la Web fue el http de dominio público desarrollado por Rob McCool en el National Center for Supercomputing Aplications (Centro Nacional para aplicaciones con grandes computadores) de la Universidad de Illinois, Urbana-Champaing. Con el tiempo, un grupo de webmasters desarrollaron sus propias extensiones y reparaciones para los «bugs» (errores) inherentes a la distribución, y se estableció un grupo para la coordinación de estos cambios (en la forma de parches). Usando el httpd 1.3 de la NCSA como base se adicionaron estos parches y todas las mejoras que se encontraron y se lanzó la primera versión oficial (0.6.2) del servidor Apache en abril de 1995.

Esta primera versión de Apache fue un éxito, sin embargo, todo su código base fue rediseñado y se le adicionaron nuevas características, para así obtener la versión 0.8.8 en agosto; una versión mejorada y con la adición de más características, esta vez en la forma de módulos estándares, se convirtió en la versión 1.0 de Apache, lanzada en diciembre de 1995.

Menos de un año después de que el grupo Apache se constituyera, el servidor Apache pasó al http de la NCSA como el servidor #1 de Internet.

Según los datos publicados por Netcraft [5], Apache es hoy en día más usado que todos los demás servidores web juntos.

2. CARACTERÍSTICAS DEL *SERVIDOR APACHE*

2.1. **Ventajas**

- *Su licencia.* Esta es de código abierto del tipo BSD que permite el uso comercial y no comercial de Apache.
- Una talentosa comunidad de desarrolladores siguiendo un proceso abierto de desarrollo.
- *Arquitectura modular.* Los usuarios de Apache pueden adicionar fácilmente funcio-

nalidad a sus ambientes específicos.
- *Portabilidad.* Apache trabaja sobre todas las versiones recientes de UNIX y Linux, Windows, BeOs, mainframes.
- Es robusto y seguro.

2.2. **Funcionamiento**

El servidor web Apache es simplemente una máquina que ejecuta el programa llamado daemon http, httpd. Al igual que otros demonios de red, httpd recibe peticiones de un cliente web, Netscape por ejemplo, y envía el recurso solicitado.

Para realizar las configuraciones de red, el servidor http Apache debe iniciarse con permisos de root. Específicamente, necesita enlazarse al puerto 80 para escuchar peticiones y aceptar conexiones. Una vez hecho esto, Apache abandona todos sus derechos y se ejecuta como un usuario distinto de root, como se especifique en sus archivos de configuración. El usuario predeterminado es **Apache**, que pertenece al grupo **Apache.**

3. DESCARGA Y COMPILACIÓN DE *APACHE*

3.1. **Descarga**

Si bien muchas de las distribuciones de Linux vienen con el servidor http, Apache puede descargar siempre la última versión desde [4].

Luego de descargar el archivo, por ejemplo **apache_1.3.tar.gz**, se descomprime con el comando:

```
[root@mihost usr] # tar –xvzf apache_1.3.tar.gz
```

Esto creará el directorio **apache_1.3**, el cual contiene todos los archivos necesarios para compilar el programa.

3.2. **Compilación**

Debido a que Apache soporta módulos con funcionalidades adicionales cargados dinámicamente, pueden existir configuraciones diferentes dependiendo de los resultados que se quieran lograr. Aquí se mostrará la configuración predeterminada, que consta de los módulos más comunes.

Para compilar, el servidor debe estar en el directorio creado luego de la instalación, para este caso es **apache_1.3**, después se ejecuta el script ./Configure, que viene con Apache.

```
[root@mihost apache_1.3] #./Configure
```

Luego de la compilación se ejecuta el comando make para compilar el programa:

```
[root@mihost apache_1.3] # make
```

Ahora se instala Apache en el directorio apropiado:

```
[root@mihost apache_1.3] # make install
```

4. CONFIGURACIÓN DEL *SERVIDOR WEB APACHE*

Apache soporta un gran conjunto de opciones de configuración que son fáciles y sencillas de seguir. La configuración por defecto de Apache es bastante eficiente y con ella podemos empezar a crear nuestros documentos HTML de manera inmediata y publicarlos.

Los archivos que van a ser accedidos a través de Apache deben estar ubicados en el directorio **/usr/local/apache/htdocs**, si se ha instalado Apache en el directorio **/usr/local/apache** o en **/var/www/html** si se utiliza la configuración predeterminada de la distribución. La página de inicio predeterminada de Apache es **index.html.**

Por otro lado, los archivos de configuración de Apache se encuentran en el directorio **/usr/local/apache/conf**, como en el caso anterior, si se ha instalado Apache en el directorio **/usr/local/apache.** Para el caso en que se utilice la configuración predeterminada de la distribución, se encontrarán en /**etc/httpd/conf.** Estos archivos de configuración son el **srm.conf, acces.conf** y **httpd.conf.** Los dos primeros son archivos que se mantienen por compatibilidad, pero realmente no se usan, contienen sólo una nota diciendo que todos los cambios de configuración deben ir en el archivo **httpd.conf.**

En este artículo mostraremos cómo trabajar con algunas de las opciones adicionales a la configuración estándar, lo que permitirá añadir características útiles de funcionalidad al servidor.

4.1. **Cambio de Página de Inicio**

Como hemos mencionado anteriormente, la página de inicio por defecto del servidor http Apache es **index.html**. Podemos cambiar esta página de inicio cambiando el valor de la entrada DirectoryIndex, la cual especifica cuál o cuáles serán las páginas utilizadas en forma predeterminada. Por ejemplo:

```
DirectoryIndex index.html index.htm inicio.html inicio.htm
```

En este ejemplo se especificó, además de las páginas index.html e index.htm, las páginas inicio.html e inicio.htm. En el caso de no encontrar las dos primeras, utilizará la siguiente en la lista, y así sucesivamente.

Cuando no se encuentra ninguna de las páginas de inicio especificadas, Apache crea de manera automática la página de examen de directorio, si se encuentra activa la opción *Indexes*. El examen de directorios consiste en el despliegue en el cliente que hace un servidor de Web de las carpetas y archivos que conforman un sitio, cuando no se tiene página de inicio predeterminada.

4.2. **Cambio de Directorio Raíz**

Cambiar el directorio raíz en el cual se va a buscar la página especificada como página de inicio también es bastante sencillo; para hacerlo sólo tenemos que modificar la entrada DocumentRoot como en el siguiente ejemplo:

```
DocumentRoot «/var/www/html/inicio»
```

Aquí hemos definido el directorio inicio dentro de /var/www/html/ en el cual ubicaremos nuestras páginas.

Es de tener en cuenta que al modificar la entrada DocumentRoot debemos modificar la directriz <Directory «/var/www/html/inicio»>, la cual especifica características para este directorio, que a continuación se describen.

4.3. **Condiciones de Seguridad**

Uno de los aspectos que resultan más útiles a la hora de establecer un servidor Web es el relacionado con la seguridad, que nos permite determinar, por ejemplo, qué usuarios pueden acceder a una página o desde qué equipos no podemos acceder a un directorio específico. Veamos cómo podemos lograr esto.

4.3.1. *Denegar la conexión a ciertas estaciones de trabajo*

Para denegar la conexión a una determinada estación utilizamos la directriz Directory junto con las directrices de control de acceso *allow* y *deny*, además de la directriz *order*, que especifica en qué orden se aplicarán las anteriores.

Esta es la forma de denegar el contenido del directorio «/var/www/html» al *host* galaxia.pruebas.com.

```
<Directory «/var/www/html»>
    order allow,deny
    allow from all
    deny from galaxia.pruebas.com
</Directory>
```

En vez de utilizar un nombre para el *host* al cual queremos denegar el acceso al directorio podemos utilizar su dirección IP. Al intentar acceder al contenido del directorio desde dicha estación aparecerá un mensaje indicando que no tiene los permisos suficientes para ver las páginas.

Obviamente podemos utilizar las mismas directrices de acceso para permitir el acceso sólo a servidores específicos, por ejemplo:

```
<Directory «/var/www/html»>
    order deny,allow
    deny from all
    allow from galaxia.pruebas.com
</Directory>
```

Estas líneas permitirán el acceso a las páginas del sitio sólo desde el *host* galaxia.pruebas.com.

4.3.2. *Autenticación*

Para el caso en el que se quiera controlar el acceso a nuestro servidor de forma específica para cada usuario, debemos completar un par de pasos:

1. La creación de un archivo de *passwords* y
2. Configurar la entrada de los directorios para usar estos archivos.

- *Paso 1*

El archivo de *passwords* es aquel en el que se establece quiénes serán los usuarios que pueden acceder a un directorio particular según se especifique. En este directorio aparecerán consignados los nombres de los usuarios con sus respectivos *passwords* encriptados.

Para crear el archivo de *passwords* se utiliza la utilidad **htpasswd**, la cual viene con Apache. Para crear el archivo, desde una terminal escribimos:

```
htpasswd -c /etc/httpd/conf/passwd username
```

Donde /etc/httpd/conf/passwd es la ruta del archivo de *passwords*. htpasswd preguntará por un *password* y luego solicitará que lo reescriba para confirmarlo. Por ejemplo:

```
htpasswd -c /etc/httpd/conf/passwd felix
New password: mypassword
Re-type new password: mypassword
Adding password for user felix
```

La opción -c es usada únicamente cuando se está creando un nuevo archivo. Después de la primera vez se deberá omitir para nuevos usuarios que se adicionen a un archivo de *passwords* ya existente.

```
htpasswd /etc/httpd/conf/passwd leo
```

En el ejemplo anterior se ha adicionado un usuario denominado leo a un archivo de *passwords* que ya ha sido creado previamente. Como antes, se preguntará por el *password* en la línea de comandos y luego por su confirmación.

El *password* es almacenado en el archivo de *passwords* en forma encriptada, sin embargo se debe guardar el archivo en una localización segura tanto como sea posible con un mínimo de permisos para que el servidor pueda leer el archivo. Por ejemplo, si su servidor está configurado para correr con el usuario Apache y grupo Apache, se deben configurar los permisos en el archivo para que sólo ese usuario pueda leer el archivo:

```
chown Apache:Apache /etc/httpd/conf/passwd
chmod 640 /etc/httpd/conf/passwd
```

- *Paso 2*

Una vez se ha creado el archivo de *passwords*, se necesita configurar Apache para que sepa que existe y lo utilice al momento de la autenticación de los usuarios para su admisión. Esta configuración se realiza con las siguientes directivas:

AuthType	:	Tipo de autenticación usado. El más comúnmente utilizado es Basic.
AuthName	:	El nombre que le queramos dar a la autenticación, proporciona el ámbito de autenticación al usuario, el nombre utilizado para especificar el conjunto particular de recursos a los que se accede con el proceso actual de autenticación.
AuthUserFile	:	La ubicación del archivo de passwords.
AuthGroupFile	:	La ubicación del archivo de grupos, si existe.
Require	:	El o los requerimientos que deben ser satisfechos para garantizar la admisión.

El siguiente ejemplo define un nombre de autenticación establecido como «Protegido». El archivo de *passwords* localizado en /etc/httpd/conf/passwd será utilizado para verificar la identidad de los usuarios. Unicamente los usuarios llamados felix o leo tendrán acceso garantizado y sólo si el password utilizado concuerda con el almacenado en el archivo de passwords.

```
<Directory «/var/www/html/Quienes/»>
    AuthType Basic
    AuthName «Protegido»
    AuthUserFile /etc/httpd/conf/passwd
    Require valid-user
</Directory>
```

La palabra «Protegido» aparecerá en la caja de diálogo del navegador utilizado (p.e. Internet Explorer), donde el usuario tendrá que autenticarse para acceder a los archivos del directorio /var/www/html/Quienes.

Para que esta configuración tenga efecto se deberá reiniciar el servidor Apache.

Si usted desea que la totalidad de los usuarios especificados en el archivo de *passwords* tengan acceso a un directorio, puede usar la palabra clave valid-user de la siguiente manera:

```
Require valid-user.
```

OPCIÓN: *Utilizar un archivo de grupos*

La mayoría del tiempo se necesitará agregar más de uno, dos o una docena de usuarios para que tengan acceso a un recurso. Es posible definir un grupo de personas que tengan acceso a este recurso y manejarlo adicionando y removiendo miembros sin tener que editar el archivo de configuración del servidor y reiniciar Apache cada vez.

Esto se hace utilizando grupos de autenticación. Un grupo de autenticación es, como se podría esperar, un nombre de grupo asociado con una lista de miembros. Esta lista es almacenada en el archivo de grupos, el cual debe ser almacenado en la misma ubicación que el archivo de *passwords.*

El formato del archivo de grupos es muy simple. Aparece primero en una línea un nombre de grupo, seguido por dos puntos, y luego una lista de los miembros del grupo separados por espacios. Por ejemplo:

```
Investigacion: leo felix jose
```

Una vez el archivo ha sido creado se puede «requerir» que alguien esté en un grupo particular para acceder a un recurso determinado. Esto se logra con la directiva AuthGroupFile, como se ve en el siguiente ejemplo:

```
AuthType Basic
AuthName «Investigacion Servicios de Red Apache»
AuthUserFile /etc/httpd/conf/passwd
AuthGroupFile /etc/httpd/conf/groups
Require group Investigacion
```

El proceso de autenticación tiene ahora un paso más. Cuando se recibe una petición y el nombre de usuario y *password* son digitados, el archivo de grupos se revisa en primer lugar para ver si el nombre de usuario se encuentra en el grupo requerido. Si así es, el archivo de *passwords* es revisado para ver si el nombre de usuario está allí, y si el *password* proporcionado corresponde con el *password* almacenado en el archivo. Si uno de estos pasos falla, el acceso se deniega.

4.4. **Alojamiento Virtual**

El alojamiento virtual permite a un servidor Web alojar múltiples sitios pertenecientes a un servidor. El servidor Apache soporta Alojamiento Virtual basado en direcciones IP o en nombres.

4.4.1. *Hosts Virtuales basados en direcciones IP*

Mediante este método, el servidor debe asignar una dirección IP distinta para cada *host virtual* que sea especificado sin importar si se trata de diferentes direcciones IP para una misma interfaz de red o si se dispone de varias interfaces.

Para configurar un *host virtual* se debe definir un bloque de directrices **Virtual Host**; dentro de éste es necesario colocar las directrices adecuadas. Es necesario, por ejemplo, emplear las directrices **ServerAdmin**, **ServerName**, **DocumentRoot** y **TransferLog** para especificar los valores concretos para ese *host.* Se debe tener en cuenta que en un bloque **VirtualHost** se puede utilizar cualquier directriz, como si se estuviera trabajando en la configuración del servidor principal, excepto **ServerType**, **StartServers**, **MaxSpareServers**, **MinSpareServers**, **MaxRequestsPerChild**, **BindAddress**, **Listen**, **PidFile**, **TypesConfig**, **ServerRoot** y **NameVirtualHost.** Estas directrices son utilizadas para la configuración del servidor principal y sus valores por defecto deben ser suficientes para la configuración de un servidor sencillo.

Al momento de configurar hosts virtuales basados en direcciones IP debemos recordar mantener una dirección IP para nuestro servidor principal. Si utilizamos todas las direcciones IP disponibles en nuestra máquina para los *hosts virtuales*, no podremos acceder a nuestro servidor principal. Sin embargo, podemos reconfigurar nuestro servidor principal como un *host virtual.*

Veamos un ejemplo:

```
<VirtualHost 192.168.1.3>
    ServerAdmin root@nebulosa.pruebas.com
    ServerName nebulosa.pruebas.com
    DocumentRoot /var/www/Sitio
    DirectoryIndex inicio.htm index.htm index.html
</VirtualHost>
```

En el ejemplo se ha definido un *host virtual* en la dirección 192.168.1.3. En éste, la directiva ServerAdmin nos indica la dirección de correo electrónico incluida en cualquier mensaje de error que se envíe a un cliente. La directiva ServerName establece el nombre de *host* para el servidor, se utiliza para crear las URL de redirección. La directiva DocumentRoot, como ya se mostró, establece el directorio desde el cual httpd sirve archivos, en este caso específicamente para este *host virtual.* Y por último, DirectoryIndex, como vimos, especifica la lista de recursos que hay que buscar cuando el cliente pide un índice del directorio especificado.

Se debe tener en cuenta que para que cada uno de los dominios definidos en los *hosts virtuales* pueda ser accedidos a través de las URL's que éstos especifican, es necesario configurarlos en el servidor DNS que sirva a nuestro sistema junto con su respectiva dirección IP.

4.4.2. ***Hosts Virtuales basados en nombres***

Como alternativa al alojamiento virtual basado en direcciones IP está el alojamiento virtual basado en nombres, que permite definir *hosts virtuales* sin utilizar direcciones IP adicionales. Esto es posible gracias al protocolo HTTP/1.1, que permite que un servidor identifique el nombre por medio del cual se está accediendo al mismo. Un navegador que utilice este protocolo puede enviar una cabecera «host:» que especifique el *host* particular que se va a usar en una máquina.

Para implementar el alojamiento virtual basado en nombres, se debe utilizar una directriz VirtualHost para cada *host,* y una directriz NameVirtualHost para especificar la dirección IP que quiere usar para los host virtuales. Dentro de la directriz VirtualHost se utiliza la directriz ServerName para especificar el nombre de dominio que quiere usar para este *host.* Esta directriz es importante para evitar una búsqueda por DNS, la que inhabilita el *host virtual.* Cada una de las directrices VirtualHost toma la misma dirección IP especificada por el argumento de la directriz NameVirtualHost. Veamos el siguiente ejemplo:

```
ServerName vialactea.universo.com
NameVirtualHost 192.168.1.1

# Virtual host andromeda (dominio1)

<VirtualHost 192.168.1.1>

    ServerAdmin andromeda@dominio1.com
    ServerName andromeda.dominio1.com
    DocumentRoot /var/www/html/
    DirectoryIndex index.html index.htm Pruebas2.htm

</VirtualHost>

# Virtual host nebulosa (dominio2)

<VirtualHost 192.168.1.1>
```

```
    ServerAdmin nebulosa@dominio2.com
    ServerName nebulosa.dominio2.com
    DocumentRoot /var/www/Sitio2/
    DirectoryIndex index.html index.htm Pruebas2.htm

</VirtualHost>
```

En este ejemplo hemos definido dos *hosts virtuales* para dos dominios diferentes, andromeda.dominio1.com y nebulosa.dominio2.com; hay que tener en cuenta que estos dominios deben corresponder con entradas en el servidor DNS.

Al tener una sola dirección IP, la implementación de *hosts virtuales* impide el acceso al servidor principal por medio de esa dirección, así que sólo se podrá utilizar para administrar los *hosts virtuales;* por esto, si se está utilizando el servidor Web principal y desea implementar *hosts virtuales* a través de nombres, será necesario configurar uno de los *host virtuales* para gestionar las páginas del servidor principal. Si tiene en su máquina más de una dirección IP, es posible utilizar una de estas direcciones para el servidor principal y otra para los *hosts virtuales* basados en nombres, e incluso es posible combinar *host virtuales* basados en direcciones IP y *host virtuales* basados en nombres, dentro del servidor. También es posible usar diferentes direcciones IP para dar soporte a diversos grupos de *host virtuales.*

5. INICIO Y PARADA

Para iniciar Apache se ejecuta el siguiente comando:

```
[root@mihost etc] # service httpd start
```

para detener el servicio se digita la siguiente línea:

```
[root@mihost etc]  # service httpd stop
```

y se reinicia así:

```
[root@mihost etc]  # service httpd restart
```

6. ACCESO AL SERVIDOR

Para acceder a las páginas montadas en nuestro servidor Apache debemos iniciar el servicio y a través de un navegador web acceder a la máquina servidor. Así, si Apache se encuentra sobre la máquina nebulosa.galaxia.com (192.168.1.1), se debe

introducir en el navegador http://galaxia.com o http://nebulosa.galaxia.com, si se definió bajo DNS o http://192.168.1.1, si no existe la definición en DNS. Inmediatamente se debe subir la página de inicio respectiva.

CONCLUSIONES

Por ser gratuito, Apache es uno de los servidores de Web más utilizados y que presenta garantías suficientes para el montaje de sitios Web confiables tanto a nivel de organizaciones independientes y para el ofrecimiento de servicios de *hosting* a otras organizaciones o en la misma organización a través de los servidores virtuales.

Una de las grandes ventajas de los servidores de Web es su capacidad de autenticación, de tal forma que controlen el acceso de usuarios y estaciones de trabajo a determinados sitios Web, y Apache no podía ser la excepción. De esta manera se mantiene una regulación en la Internet en lo que respecta a qué usuarios están en capacidad o impedidos de conocer la información de una organización.

A través de los servidores virtuales, Apache oculta la existencia de un solo servidor real y muestra la presencia de varios servidores, haciendo pensar a los usuarios que tienen una máquina servidora para cada uno de ellos cuando en realidad corresponde a la misma máquina físicamente.

Glosario

- *BSD* (Berkeley Software Distribution): Desarrollo de Unix llevado a cabo en la Universidad de California. Con estas tres letras se denota este estándar.
- *Dirección IP:* Dirección a través de la cual se puede hacer referencia a cada una de las computadoras conectadas a una red. Esta dirección consiste en un número compuesto por cuatro segmentos separados por puntos. Dependiendo del tipo de red, algunos de los primeros segmentos se utilizan para direcciones de red y algunos de los últimos segmentos para direcciones de *host.*
- *Dominio:* De manera simple podemos definir un «dominio» como cada una de las subredes en las cuales está dividida Internet, cada una con un nombre de dominio especificado.
- *DNS* (Domain Name Server, Servidor de Nombres de Dominio): Es un servicio de Internet que convierte nombres de dominio a sus correspondientes direcciones IP.
- *Host:* Utilizado a veces como sinónimo de *mainframe,* en realidad identifica al ordenador central en un sistema informático complejo.
- *Hosting:* Capacidad de un servidor para almacenar sitios Web.

Referencias

[1] MOHR, James. *Linux Recursos para el usuario.* México, Pearson, 1999. 825 p.

[2] PETERSEN, Richard. *Linux Manual de Referencia,* 2ª ed. Madrid, Osborne-McGraw-Hill, 2001. 1306 p.

[3] SCHENK, Thomas. *Administración de Red Hat Linux Al Descubierto.* Madrid, Pearson Educación, 2001. 1148 p.

[4] http://www.apache.org/dist. 15 de diciembre de 2001.

[5] http://www.netcraft.com/survey/. 23 de enero de 2002.

[6] http://www.linuxdoc.org. 23 de enero de 2002.

[7] http://linuxcol.uniandes.edu.co. 25 de enero de 2002.

[8] http://lucas.hispalinux.es. 16 de diciembre de 2001.

[9] http://www.geocities.com/linux2005. 16 de diciembre de 2001.